REACTIVE GRASP WITH PATH RELINKING FOR BROADCAST SCHEDULING

Clayton W. Commander
96 Communications Group,
Eglin Air Force Base, FL

Sergiy I. Butenko
Department of Industrial Engineering,
Texas A&M University

Panos M. Pardalos, Faculty Advisor and Carlos A.S. Oliveira
Department of Industrial and Systems Engineering,
University of Florida

ABSTRACT

The Broadcast Scheduling Problem (BSP) is a well known NP-complete problem that arises in the study of wireless networks. In the BSP, a finite set of stations are to be scheduled in a time division multiple access (TDMA) frame. The objective is a collision free transmission schedule with the minimum number of TDMA slots and maximal slot utilization. Such a schedule will minimize the total system delay.

We present variations of a Greedy Randomized Adaptive Search Procedure (GRASP) for the BSP. Path-relinking, a post-optimization strategy is applied. Also, a reactivity method is used to balance GRASP parameters. Numerical results of our research are reported and compared with other heuristics from the literature.

KEYWORDS

Broadcast Scheduling Problem, Ad-hoc Networks, Combinatorial Optimization, GRASP, Heuristics

INTRODUCTION

In recent years, research in the area of wireless communications has increased dramatically. This is a result of improved technology and increasing demands. One particular focus has been in the area of so-called ad-hoc networks. Along with this research has been the introduction of many exciting and challenging problems [2, 10]. The problem of interest for this article is known as the Broadcast Scheduling Problem (BSP) [4]. Before describing the problem in detail, we will first give a brief review of ad-hoc networks.

Ad-hoc networks provide high speed communication between potentially mobile receivers by the application of a packet switching technique over a shared radio channel. In such networks, each station can act as both a transmitter and a receiver. Thus, it is often necessary to use intermediate stations as relays to forward messages over the network to the intended recipient. Applications of ad-hoc networks can be seen in military battlefield scenarios and mobile commerce [2].

Since every station in the network shares the same channel, it is crucial that precautions are taken when messages are scheduled to be transmitted. That is, stations should be scheduled in such a way that there is no destructive interference, or message collision [22]. There are two types of collision in ad-hoc networks. *Direct collision* is a result of two adjacent stations broadcasting during the same time slot. *Hidden collision* occurs when two non-neighboring stations transmit simultaneously to a station that can receive messages from both senders. The desired result is a broadcast schedule which guarantees collision free transmissions and minimizes the overall delay of the system [4].

PROBLEM DESCRIPTION

We can model an ad-hoc TDMA network as follows. Consider a graph $G = (V, E)$ where the vertex set $V = \{1, 2, \ldots, n\}$ represents the stations in the network. Then we can successfully model the network by letting the edge set E represent the set of transmission links between adjacent stations in the network. We say that stations i and j are *one-hop neighboring stations* iff there exists an undirected edge $(i, j) \in E$. An equivalent interpretation of V is that it represents the set of direct collisions. If $(i, j) \notin E$ but there exists an intermediate node $k \in V$ such that $(i, k) \in E$ and $(k, j) \in E$, then stations i and j are referred to as *two-hop neighboring stations.* A hidden collision is a result of two-hop neighbors transmitting in the same slot [3].

Let C be an $N \times N$ symmetric binary matrix, where $N = \mid V \mid$. Then we can represent the set of one-hop neighbors in the *connectivity matrix* $C = \{c_{ij}\}$ as follows:

$$c_{ij} = \begin{cases} 1, \text{if } (i, j) \in E \text{ and } i \neq j, \\ 0, \text{otherwise.} \end{cases}$$

There are some assumptions that must first be made before we formally define the problem statement. We assume that there are M time slots per TDMA frame, and that each slot length is equal to the amount of time required to transmit one packet of data. We assume also that packets are received in the same slot they are transmitted and packets are sent at the beginning of each time slot. Now we represent the broadcast schedule as a $M \times N$ binary matrix $S = \{s_{mn}\}$ defined as follows:

$$s_{mn} = \begin{cases} 1, \text{if station } n \text{ is to broadcast in slot } m, \\ 0, \text{otherwise.} \end{cases}$$

In order to analyze the efficiency of a broadcast schedule, we need to calculate the percentage of the available slots being assigned in a transmission frame [3]. Let ρ_n be the slot utilization for station n. Then,

$$\begin{aligned} \rho_n &= \frac{\text{the number of slots assigned to station } n}{\text{frame length}} \\ &= \frac{\sum_{m=1}^{M} s_{mn}}{M}. \end{aligned}$$

Hence, the total slot utilization of the entire network, ρ, is given by

$$\begin{aligned} \rho &= \frac{\sum_{n=1}^{N} \rho_n}{N} \\ &= \frac{\sum_{m=1}^{M} \sum_{n=1}^{N} s_{mn}}{NM}. \end{aligned}$$

With these tools, we can now represent the Broadcast Scheduling Problem as follows:

$$\text{Minimize } M \text{ and Maximize } \rho$$

subject to:

$$\sum_{m=1}^{M} s_{mn} \geq 1, \qquad \forall n, \tag{1}$$

$$c_{ij} + s_{mi} + s_{mj} \leq 2, \qquad \forall i, \forall j, \text{and } \forall m, i \neq j, \tag{2}$$

$$c_{ik}s_{mi} + c_{kj}s_{mj} \leq 1, \qquad \forall i, \forall j, \forall k, \text{and } \forall m, i \neq j, j \neq k, k \neq i. \tag{3}$$

Contraint (1) ensures that each station transmits at least once per frame. Constraint (2) ensures that one-hop neighbors do not transmit in the same slot. Finally, the last constraint prevents two-hop neighbors from transmitting during the same slot [22]. The BSP was shown to be NP-complete in [4].

GRASP FOR THE BSP

Greedy Randomized Adaptive Search Procedure (GRASP) [5, 7, 17] is a multi-start metaheuristic for combinatorial optimization problems first introduced by Feo & Resende in [6]. GRASP is a two-phase procedure. The first phase is known as the construction phase in which an initial greedy randomized solution is formed. Since a construction phase solution is not guaranteed to be locally optimal, phase two implements a local search procedure to improve the initial feasible solution. The best solution produced from all GRASP iterations is returned as the output. GRASP is easily adaptable and has been successfully applied to such problems as broadcast scheduling [3], quadratic assignment [13, 14], and most recently to the p-median problem [18] and the uncapacitated facility location problem [19].

Construction Phase: In [3], Commander, et. al, applied GRASP to the BSP with satisfying results. Here, we will implement the same construction phase originally employed in [3]. Initially, the stations are sorted in descending order of the number of one-hop and two-hop neighbors. Then, the station with the most neighbors is assigned. After this greedy choice, the restricted candidate list (RCL) is created and consists of the best $\alpha\%$ of stations which may simultaneously transmit with the greedy assigned station. In our case, we use $\alpha = 20$. A station is then selected at random from the RCL and assigned in the current slot. A new RCL is then created, and another station randomly selected and assigned. This process continues until RCL $= \emptyset$, at which point the slot number is incremented and the process restarts with another greedy choice. The selection of the greedy choice is biased towards those stations which have not been previously assigned. Though it is desired to have multiple broadcasts by each station, this bias is to help minimize the frame length.

Local Search: Again, the local search used here is taken from [3]. Using the schedule produced in the construction phase, the slots are sorted in descending order of the number of bursts. The two slots with the fewest transmissions are combined, and the number of slots is now $k = m - 1$. Call this modified schedule $s_{m',n}$ and define $E(m'_i)$ to be the set of collisions in slot m'_i. If we sum along all the slots, we have the function to minimize as $f(s) = \sum_{i=1}^{k} E(m'_i)$. We apply the following local search for this minimization.

```
procedure PathRelinking(Guide, Current)
    $slot \leftarrow 1$;
    do while $slot \neq \sigma_M \rightarrow$
        if $|guide(slot)| > |current(slot)| \rightarrow$
            $current(slot) \leftarrow guide(slot)$;
        fi
        if all stations assigned at least once $\rightarrow$
            EXIT;
        fi
        $slot \leftarrow slot + 1$;
    od elihw
end PathRelinking
```

Figure 1. Pseudocode for the Path Relinking subroutine

A colliding station from the combined slot is randomly selected and every attempt is made to swap this station with another from the remaining $k-1$ slots. After each swap, $f(s)$ is re-evaluated. If the new value of $f(s)$ is less than the value before the swap, then this swap is deemed a success and another colliding station from the combined slot is selected and the swap exchange repeated. However, if $f(s)$ is not improved, the swap is undone and another is attempted.

If after every attempt no successful swap is made, a new colliding station is randomly chosen and again the swap procedure is attempted. This process continues until either a successful swap is made, or until some iteration limit is reached. If this combined solution $s_{m'n}$ is improved such that $f(s) = 0$, then the frame length has been successfully decreased by one slot. The two slots with the fewest broadcasts are combined, the value of k is once again decremented, and the process repeats. In the end if $f(s) > 0$ then no improved solution was found and the original construction phase solution is returned as best.

Path Relinking: First introduced by Glover in [9], path relinking (PR) was used as an enhancement for tabu searches. PR was first combined with GRASP by Laguna and Martí in [12]. When applied to GRASP, path relinking introduces a memory to the heuristic which usually results in improvements in solution quality. In the standard GRASP, the multi-start nature of the heuristic doesn't include any mechanism for remembering traits about solutions generated in each iteration. Thus nothing can be recalled about why or why not a certain solution was more favorable than another. Path relinking allows GRASP to remember these traits and favor them in successive iterations. GRASP with PR was successfully applied to problems such as job shop scheduling [1], quadratic assignment [14], and originally for line crossing minimization [12], to name a few instances. In this paper, we propose a variant of path relinking for BSP.

Path relinking works by using a set of elite solutions as guides and examines point to point trajectories in search of an optimal solution. With the GRASP for the BSP, the set of elite solutions is stored in memory. Our elite set contains the ten best solutions up to the current iteration. After a normal GRASP iteration, one of the elite solutions is chosen at random to be a guiding solution. There is then a slot-wise comparison between the guiding solution and the current solution. If in the first slot, the guiding solution has more scheduled stations than does the current solution, then that slot in the current solution is replaced by the slot from

Stations	LB	Frame Length				Channel Utilization			
		G	RG+PR	SVC	MFA	G	RG+PR	SVC	MFA
15	8	8	8	8	8	0.167	0.167	0.15	0.15
30	10	10	10	11	12	0.120	0.120	0.112	0.108
40	8	8	8	8	9	0.203	0.206	0.188	0.197

Figure 2. Comparison of frame length and utilization all tested heuristics. The lower bounds (LB) were determined by calculating the clique number as described in [11].

the guiding solution. If however the current slot has a greater throughput than the guiding solution, then it is kept and the next slot examined. At this point, the procedure checks to see if all stations have been assigned. If they have, then the procedure stops and the solution is returned. If all stations have not been assigned, then the slot-wise comparison continues and after each slot another "all-broadcast" check is performed. The process is allowed to continue for at most σ_M slots, where $\sigma_M = \min\{M_{guide}, M_{current}\}$. If at this point, all stations have still not been scheduled, then the program exits and the solution with σ_M slots is returned as the best solution from the current iteration. Pseudocode for the PR subroutine can be seen in Figure 1. Here $guide(i)$ refers to the set of vertices in the i^{th} slot of the guiding solution.

Note that unlike the standard path relinking schemes, our procedure does not necessarily "relink" the starting and guiding solutions, *i.e.*, it aims to introduce certain attributes of the guiding solution into the current solution, while the generated path may not reach the guiding solution in the end.

Reactive GRASP: The idea of reactive GRASP (RG) [15] is a quite helpful addition when implementing the GRASP to solve a problem. The RG method automatically determines the value of α parameter. Recall that α determines the size of the RCL. Without Reactive GRASP, the practitioner must determine the best value of α by brute-force testing.

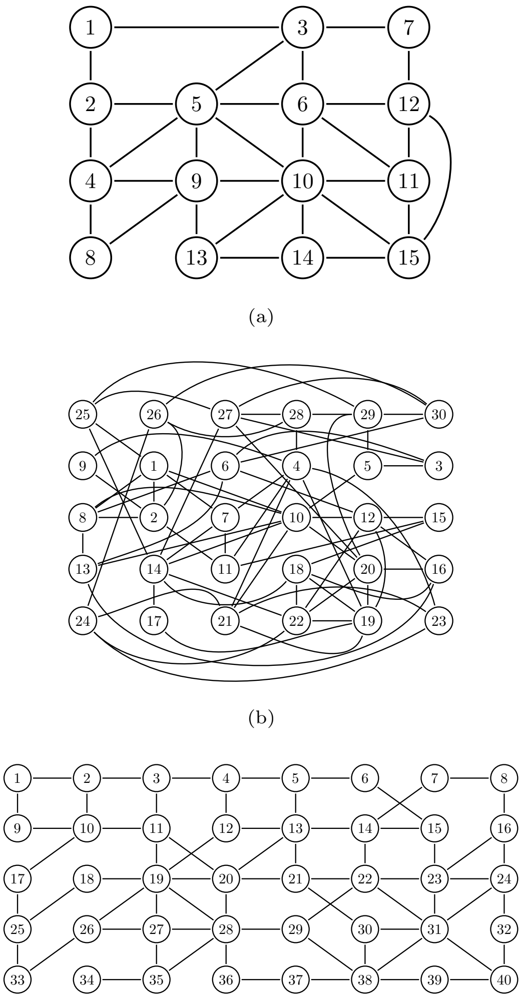
For our application of RG, we will use a common approach first introduced in [15] and again in [10]. The method works as follows. We first start with a set $\aleph = \{\alpha_1, \alpha_2, \ldots, \alpha_k\}$ of potential α values. We follow the same convention as Gomes, et. al, and begin with $k = 10$ and $\aleph = \{.1, .2, \ldots, 1.0\}$. From each iteration, an α_i parameter is selected from $\aleph$ with some probability, say p_i. Initially, there are no favored choices so p_i is uniform for all i.

As the iterations progress, certain values of α will produce better results than others. Therefore, we have another set $A = \{a_1, a_2, \ldots, a_n\}$ where each element $a_i \in A$ stores the average solution value found using parameter $\alpha_i \in \aleph$. We then determine the values for $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$, where $\lambda_i = \frac{f(s^*)}{a_i}$ and s^* represents the current best solution. For our problem, we define $f(s)$ to be the minimum frame length found. Lastly, the probabilities p_i are updated such that $p_i = \frac{\lambda_i}{\sum_{k=1}^{n} \lambda_k}$.

It turns out that reactive GRASP can provide benefits over traditional GRASP while adding little additional overhead. Since more options are available, the program can tailor itself to produce improved solutions by varying the size of Restricted Candidate List [10].

NUMERICAL RESULTS

GRASP was tested on three networks which were first introduced by Wang & Ansari [21], and have since become the default test cases for broadcast scheduling heuristics. These examples



(a)

(b)

(c)

Figure 3. (a) 15 station network. (b) 30 station network. (c) 40 station network.

include a 15, 30, and 40 station network having varying densities. These networks can be seen in Figure 3.

Figure 4 is a comparison of network utilization versus time for the two GRASP heuristics for the 40 station network. It can be seen that reactive GRASP with path relinking (RG+PR) converges much faster than the standard GRASP. Also, the RG+PR heuristic results in a better solution than standard GRASP. With the additional steps of reactivity and path relinking there is the common exchange of higher computation time yielding a better quality solution. However, this trade-off is minimal since higher quality solutions are found in the early iterations of the enhanced procedure.

Comparative results for the test cases are given in Figure 2. The algorithms tested are GRASP (G) and reactive GRASP with path relinking (RG+PR). Results from these heuristics are compared with the sequential vertex coloring (SVC) algorithm from [22] and the mean field annealing (MFA) heuristic from [21].

Notice that both GRASP heuristics attain the optimal frame lengths for all test cases. Also, standard GRASP achieves solutions at least as good as the MFA and SVC algorithms. The enhanced GRASP routine outperforms the other heuristics for all test cases. It is not known whether these solutions are optimal for these relatively small test cases or not. It would be interesting to formulate the BSP as an integer programming problem and use a commercial IP solver to determine the optimal solution.

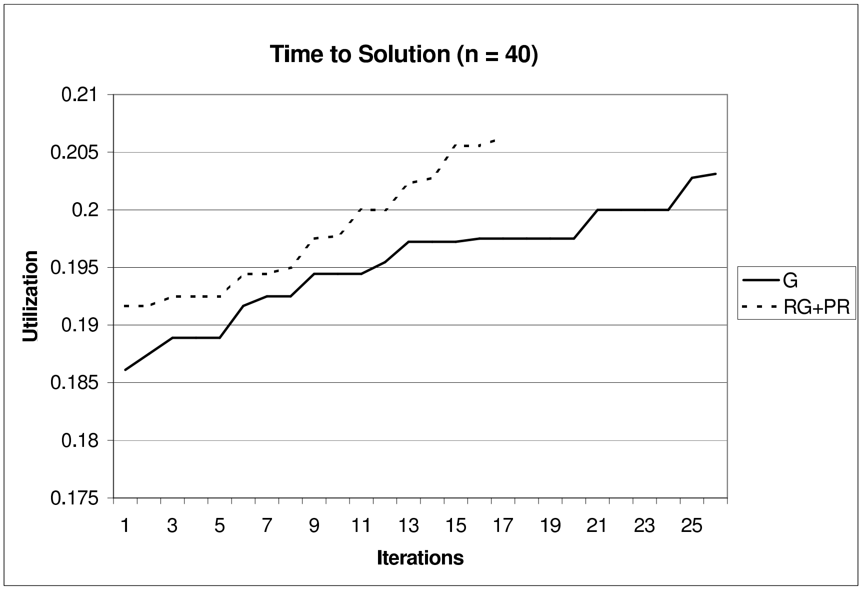


Figure 4. Time to solution comparison for standard GRASP and Reactive GRASP with Path Relinking for 40 station example.

CONCLUSIONS

In this article, we introduced the Broadcast Scheduling Problem as an important NP-complete problem which arises in the implementation of wireless ad-hoc networks and plays a crucial role in telemetry systems. We presented several variants of the Greedy Randomized Adaptive Search Procedure (GRASP) and analyzed the results on some well-known test cases. When compared to other heuristics in the literature, the standard GRASP is the better performer. The solutions are then increased by the application of reactive GRASP and the post-optimization step of path relinking. For further research, it would be interesting to see the performance of the enhanced GRASP metaheuristics on some larger graphs, such as those provided by Commander, et. al in [4].

REFERENCES

1. Aiex, R.M., S. Binato, and M.G.C. Resende, “Parallel GRASP with Path Relinking for Job Shop Scheduling,” Parallel Computing, vol. 29, pp. 393-430, 2003.
2. Butenko, S.I., X. Cheng, C.A.S. Oliveira, and P.M. Pardalos, “A New Algorithm for Connected Dominating Sets in Ad Hoc Networks,” in S. Butenko, R. Murphey, and P. Pardalos, editors, Recent Developments in Cooperative Control and Optimization, pp 61-73, Kluwer Academic Publishers, 2003.
3. Commander, C.W., S.I. Butenko, and P.M. Pardalos, “A Greedy Randomized Adaptive Search Procedure for the Broadcast Scheduling Problem,” submitted to Journal of Combinatorial Optimization, 2004.

4. Commander, C.W., S.I. Butenko, and P.M. Pardalos, “On the Performance of Heuristics for Broadcast Scheduling,” to appear in D. Grundel, R. Murphey, and P. Pardalos, editors, Theory and Algorithms for Cooperative Systems, Kluwer Academic Publishers, 2004.
5. Feo, T.A. and M.G.C. Resende, “Greedy Randomized Adaptive Search Procedures,” Journal of Global Optimization, vol. 6, pp. 109-133, 1995.
6. Feo. T.A. and M.G.C. Resende, “A Probabilistic Heuristic for a Computationally Difficult Set Covering Problem,” Operations Research Letters, vol. 8, pp.67-71, 1989.
7. Festa, P. and M.G.C. Resende, “GRASP: An Annotated Bibliography,” in C.C. Ribeiro and P. Hansen, editors, Essays and Surveys on Metaheuristics, pp 325-367, Kluwer Academic Publishers, 2001.
8. Garey, M.R. and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman, New York, NY, 1979.
9. Glover, F., “Tabu Search and Adaptive Memory Programming - Advances, Applications, and Challenges,” in R.S. Barr, R.V. Helgason, and J.L. Kennington, editors, Interfaces in Computer Science and Operations Research, pp 1-75, Kluwer Academic Publishers, 1996.
10. Gomes, F.C., P.M. Pardalos, C.A.S. Oliveira, and M.G.C. Resende, “Reactive GRASP with Path Relinking for Channel Assignment in Mobile Phone Networks,” Proceedings of the 5th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, pp 60-67, 2001.
11. D. Jungnickel, Graphs, Networks and Algorithms. Berlin, Germany: Springer-Verlag, 1999.
12. Laguna M. and R. Martí, “GRASP and Path Relinking for 2-Layer Straight Line Crossing Minimization,” INFORMS Journal on Computing, vol. 11, pp. 44-52, 1999.
13. Li. Y, P.M. Pardalos, and M.G.C. Resende, “A Greedy Randomized Adaptive Search Procedure for the Quadratic Assignment Problem,” in P.M. Pardalos and H. Wolkowicz, editors, Quadratic Assignment and Related Problems part of DIMACS Series on Discrete Mathematics and Theoretical Computer Science, vol. 16, pp. 237-261, 1994.
14. Oliveira C.A.S., P.M. Pardalos, and M.G.C. Resende, “GRASP with Path Relinking for the QAP,” 5th Metaheuristics International Conference pp. 57.1-57.6, 2003.
15. Prais M. and C.C. Ribeiro, “Reactive GRASP: An Application to a Matrix Decomposition Problem in TDMA traffic Assignment,” INFORMS Journal on Computing vol. 12, no. 3, pp. 164-176, 2000.
16. Resende, M.G.C. and C.C. Ribeiro, “GRASP with Path-relinking: Recent Advances and Applications”, in T. Ibaraki, K. Nonobe, and M. Yagiura, editors, Metaheuristics: Progress as Real Problem Solvers, Kluwer Academic Publishers, 2005.
17. Resende, M.G.C. and C.C. Ribeiro, “Greedy Randomized Adaptive Search Procedures”, in F. Glover and G. Kochenberger, editors, Handbook of Metaheuristics, pp 219-249, Kluwer Academic Publishers, 2003.
18. Resende, M.G.C. and R.F. Wernece, “A Hybrid Heuristic for the P-Median Problem,” accepted to Journal of Heuristics, 2003.
19. Resende, M.G.C. and R.F. Werneck, “A Hybrid Multi-Start Heuristic for the Uncapacitated Facility Location Problem,” AT&T Labs Research Technical Report TD5-RELRR, Florham Park, NJ, Sept. 2003.
20. Salcedo-Sanz, S., C. Busoño-Calzón, and A.R. Figueiral-Vidal, “A Mixed Neural-Genetic Algorithm for the Broadcast Scheduling Problem,” IEEE Transactions on Wireless Communications, vol. 2, no. 2, 2003.

21. Wang G. and N. Ansari, “Optimal Broadcast Scheduling in Packet Radio Networks Using Mean Field Annealing,” IEEE Journal on Selected Areas in Communications, vol. 2, no. 2, 2003.
22. Yeo, J., H. Lee, and S. Kim, “An Efficient Broadcast Scheduling Algorithm for TDMA Ad-hoc Networks,” Computers and Operations Research, vol. 29, pp 1793-1806, 2002.