Intelligent Power Allocation Strategies in an Unlicensed Spectrum

N. Clemens & C. Rose
WINLAB at Rutgers University, Piscataway, New Jersey 08854
Email: nevillec@winlab.rutgers.edu, crose@winlab.rutgers.edu

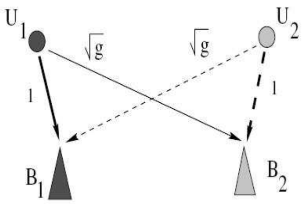

Fig. 1. Our channel model, with two user-base pairs and symmetric cross gains.

Abstract—
We consider power allocation strategies for radios operating in unlicensed bands. Since each radio's power allocation is a source of interference for other radios sharing the same spectrum, we seek to develop "intelligent" power allocation strategies that not only greedily optimize performance, but also tend toward socially optimal solutions. Radio interaction is modeled as a two-player repeated game where players develop "reputations" based on previous actions. Bad behavior can result in punishment meted out by other players while socially acceptable behavior could be rewarded. The space of possible radio strategies is explored using genetic algorithms and we find that there are identifiable strategy features shared by all good policies. We conclude with a brief discussion of the possible applicability of our strategies and algorithms, depending on the operating environment and computational complexity at hand.

I. Introduction

Unlicensed spectrum poses interesting problems in radio etiquette design. How should radios "behave" in a crowd of other radios when everyone is sharing a common band and no central server is available to provide advice or instruction? If radio "behavior" is defined by the manner in which it spreads its energy over the available signal space, then the behavior of each radio will influence the channel – and hence the channel capacity – of every other user in that signal space. Thus, the independent decisions taken by each radio affect the rest of the radio "population."

To gain insight, we consider the abstraction of a simple two-player repeated game. We assume two transmitter/receiver pairs and that each receiver can "hear" both transmitters, but is interested only in one of the transmissions as shown in FIGURE 1. The signal gain to the desired receiver is assumed to be unity while the interference gain is $\sqrt{g}$ as shown.

Again for simplicity, we also assume only two orthogonal channels are available to each user. If user 1 puts a fraction x of his power in channel 1 then the remaining $1-x$ fraction of power is in channel 2, and similarly for user 2's fractional power y. Assuming white Gaussian noise, Gaussian signaling by each user and each user treating the other as (colored) Gaussian noise, the capacities seen by each user are

$$C_1 = \frac{1}{2}log\left(1+\frac{x}{gy+N}\right)+\frac{1}{2}log\left(1+\frac{1-x}{g(1-y)+N}\right) \tag{1}$$

and

$$C_2 = \frac{1}{2}log\left(1+\frac{y}{gx+N}\right)+\frac{1}{2}log\left(1+\frac{1-y}{g(1-x)+N}\right) \tag{2}$$

We define the *collective capacity* [2] of the system by

$$C_c = C_1 + C_2 \tag{3}$$

and emphasize that C_c is *not* the information theoretic sum capacity of the implicit interference channel [1], [3], [4], [5]. That is, we assume no collaboration between users (as in shared codebooks, for example) so as to simplify the problem since the capacity region of the interference channel has eluded complete specification for over fifty years.

We choose to model our user-base pair interaction as a 2-player game, the model of which we explain in section II. Briefly put, the players are the user-base pairs and the actions available to them are their power allocation choices - namely the values they assign to x and y in equation (1) and equation (2). If we assume that the actions taken by the players are fixed over time, then each player's best strategy to maximize its expected payoff will take the game to a Nash Equilibrium, if it exists. However, the Nash Equilibrium is not always a desirable outcome for the game because it can lead to poor performance for both players [2].

Fortunately, with the advent of cognitive radios that can adapt their power allocation schemes, we can instead model radio interaction as a series of repeated games where each radio reacts to past outcomes. We ask the question of whether strategies can be devised which benefit both users in a distributed fashion without explicit and pre-defined cooperation (i.e., channel assignment).

Such dynamic/repeated game theory problems are notoriously difficult and generally there are no closed form expressions for strategies. We therefore represent strategies as a string of suitable symbols and use genetic algorithms to search the strategy space. The use of genetic algorithms as a tool for finding optimal strategies in games was inspired by previous work in this area by John Holland [10] as well as Axelrod's tournament [6] on the evolution of winning strategies in games.

Player 2's Action

		Ch. 1	Spread	Ch. 2
Player 1's Action	**Ch. 1**	(1.46,1.46)	(2,7.2)	(6.9,6.9)
	Spread	(7.2,2)	(2.6,2.6)	(7.2,2)
	Ch. 2	(6.9,6.9)	(2,7.2)	(1.46,1.46)

Fig. 2. This is the payoff matrix, calculated for $g = 0.3$ and $N = 10^{-3}$. The payoffs in the matrix are to be read as (Payoff for player 1, Payoff for player 2).

Often, the use of genetic algorithms can result in optimization without reification of structural features of good solutions. However, in this case we have been fortunate in being able to identify a handful of policy features which are common to good policies. We go on to demonstrate that these common "traits" of good policies can then be used as a skeletal framework to construct near-optimal policies. We elaborate on our results in sections IV, V and VI.

II. Game Model

In every game we have a set of players, a set of actions for each player and a payoff-tuple defined for every possible action-tuple played in a game. We consider the interaction of the user-base pairs as a two player game with each user-base pair represented as a player. The set of actions available to each player is the set of power distributions that we permit the transmitters to have. In our case, this simply corresponds to the values of x and y that we allow in equation (1) and equation (2). We define each player's payoff in a game as the channel capacity for a given choice of the action tuple (x,y), i.e. if the action-tuple is (x,y) for some permissible values of x and y, then the corresponding payoff-tuple is simply (C_1,C_2) as calculated in equation (1) and equation (2). We further simplify the game by imposing the condition that the actions for each player (values for x and y) can only be drawn from the set $0,0.5,1$. Thus, each player has three possible actions to choose from to play in a game.

The action $x = 0$ corresponds to player 1 putting all his power in channel 2. The action $x = 0.5$ corresponds to player 1 spreading his power equally over both channels and the action $x = 1$ corresponds to player 1 putting all his power in channel 1. Similarly for y and player 2. Thus each game has 9 possible outcomes corresponding to each ofthe 3^2 possible action-couplets. We represent these outcomes by a payoff matrix as shown in TABLE 2, for $g = 0.3$ and $N = 10^{-3}$. *In all our simulations, these are the values of g and N that are used.*

This restriction on player actions is imposed owing to the problem structure which dictates that the optimal solution will be either complete overlap or complete segregation depending on the value of the gain, g [14], [13], [15], [12], [2]. The threshold for g above which segregation is optimal was calculated as

$$g > \left(\frac{1}{\sqrt{2\rho+1}-1} - \frac{1}{\rho}\right) \tag{4}$$

where $\rho = 1/2N$ represents the raw SNR of each user. For the noise floor that we have used (N = 0.001), this threshold comes out to be 0.0306. Thus, the games and results presented here are most appropriate for *moderate* interference as opposed to weak interference where simple waterfilling is often adequate, or strong interference where users naturally segregate in the signal space.

It should be noted, however, that an analytic proof for the *optimality* of either complete segregation or complete overlap has not been given for the general symmetric channel. That is, [2] considered only mutually water filling solutions and not general power distributions. Nonetheless, for the two-channel, two player symmetric problem considered here, complete segregation or complete overlap *are* the optimal solutions which maximize the collective capacity, depending upon whether weak, moderate or strong interference is considered.

We model our radio interaction as a *series* of such games. In such a case, it is reasonable to expect each player to try and maximize its payoff over the course of the entire series of games. Since the Nash Equilibrium brings both players to a mutually inefficient payoff per game, it will not be the optimal operating point over a series of repeated games.

In our particular game, defined by the payoff patrix in TABLE 2, the Nash Equilibrium is the operating point where both players choose to 'spread' their power, resulting in each player scoring 2.9. Clearly, a socially better operating point would be the players stick to different channels so that each scores 6.9. In fact, it can be shown that for this payoff matrix this is the socially *optimal* operating point. However, at such an operating point each player is tempted to spread his power in the next game with the incentive of increasing his score (in the next game) to 7.2. But an intelligent opponent will immediately recognize this treachery and retaliate in the subsequent game by spreading his own power - thus bringing the game back to the inefficient Nash Equilibrium. Thus 'good' policies are those that can recognize the optimal operating point as channel segregation and resist the temptation to act with blind greed - knowing that it is in their own interest in the long term (over a series of repeated games with the same opponent).

A characteristic of a repeated game is the memory associated with a player. If a player is memoryless, then each game is independent of all previous games and this will reduce to a series of independent one-off games which will converge to the Nash Equilibrium at each game. However, when a player has a memory, he bases his next move on the outcome of the previous games. *In our games, we fix the depth of the players' memories at two games*, i.e. they can base their actions on the outcomes of the previous two games. It is now our objective to devise strategies for players that will attempt to reach these optimal operating points.

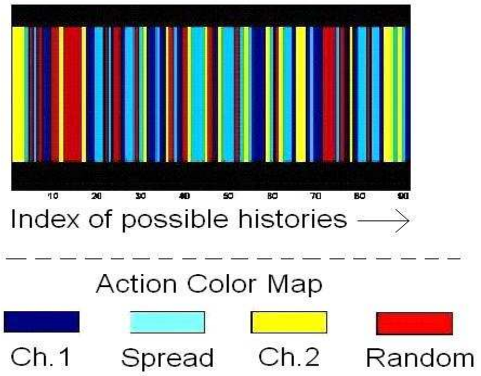

Fig. 3. A player's strategy can be completely represented by a 'genome', which is a string of actions. Here each color represents an action, and each position in the string represents one of the possible histories. Thus, in order to make his move, a player locates the history of the last two games on his genome and plays the corresponding action (color).

III. The Genome and the Genetic Algorithm

We model each player's strategy as a genome - a string of symbols (genetic 'letters') that will completely define a player's strategy over all time. The basic idea behind the genome is to map every possible history in a player's memory (the outcomes of a *finite* number of games in the past - two, in our case) to a particular action which will be taken in response to that history. Thus, a player's strategy consists of choosing a response to every possible outcome that could have occurred in his (finite) past. With 9 possible outcomes per game, the number of possible outcomes over two games will be $9x9 = 81$. Thus, a player with a memory of 2 games (which is all that our players can remember) will have a strategy 'booklet' that will tell him what action to take for each of these 81 possibilities. This 'booklet' will be our genome, represented by a string of colors (each color representing an action). Each *position* in that genome will correspond to one of the 81 possible outcomes. Thus, each possible outcome in a players memory is indexed (from 1 to 81) and when outcome i occurs in the past, the player looks up the i^{th} position in his genome and plays whatever action appears at that position of his genome.

Note: There are 81 possibilities to cover when a player has had 2 games in the past. However, at start-up, a player needs to be told what to play in the very first game as well as what to play in the second game. That requires another 1 + 9 positions in the genome. Hence the size of a player's genome is actually 1 + 9 + 81 = 91 letters long, to define a player's strategy from start-up.

The actions are represented by colors in the genome, one color representing each action. We use four colors - one for each of the three possible actions and another to represent a random action chosen from one of the previous three. Randomness is often useful in preventing "lockup" which can result in mutually destructive behavior between similar strategies.

We see that for a genome with 91 positions and 4 possible actions (colors) at each position, the number of possible strategies is 4^{91}. With a lack of analytical tools to arrive at an optimal genome (or strategy), we have used genetic algorithms to search this large space for 'good' strategies. Our method of using genetic algorithms is inspired by John Holland's work on identifying robust strategies for games [10] and Robert Axelrod's tournament [6]. The following steps outline our genetic algorithm:

1. Construct an *Evaluator Set* E, which is a collection of strategies. In our case, this was comprised of a mixture of random strategy strings (genomes) and some hand-crafted strategies - players that blindly stick to one channel, players that always spread their power, players that follow the opponent in signal space, or avoid the opponent, hop from one channel to the other, etc. This is the set of strategies against which intermediate populations will be evaluated.
2. Generate a random population S of strategies, i.e. a random collection of genomes. This is the population that will evolve in our genetic algorithm.
3. Each member of S is then played off against each member of E in a series of games - a match. Each match consists of a fixed number of games. Thus, at the end of this 'tournament', every member of S has played a match with every member of E.
4. For every game that a member of S plays it receives a score depending on the outcome of that game, as per the payoff matrix shown in TABLE 2. So at the end of the tournament, every member of S has an average score over *all* the games of *all* its matches. This average score is our metric for the fitness of that particular strategy genome.
5. The genomes in S are then ordered according to their fitnesses (which are their average scores as calculated in the preceeding step). The next generation's population must now be constructed. The two fittest genomes are copied into the new population as they are. Then 'mating' of the genomes is done between successive pairs in the ordered set of our genomes S, for all but the weakest two genomes (i.e. 'reaping' is used in our genetic algorithm). No mutation is used. Once done, we have a new population of the same size as our previous population and this becomes our new population set S.
6. Steps 3 to 5 are repeated for a fixed number of generations (80 in our simulations). The final population S is then an optimized set of strategies for performance against the evaluator set E.
7. The fittest member from this final population S is added to the evaluator set E. Thus, our evaluator set is incremented with a new evolved strategy.
8. Steps 2 to 7 are repeated a fixed number of times. Note that at each iteration of this process, a complete genetic algorithm is run to produce a final population that is tuned to perform optimally against the evaluator set E. At each iteration we are incrementing the evaluator set with an intelligent strategy and then running a genetic algorithm to evolve a population against this smarter evaluator set.
9. The final population S at the end of all this is a collection of our final 'intelligent' strategies.

IV. Strategy Performance

Our iterative genetic algorithms yield a different final population of genomes at each run. We call these as

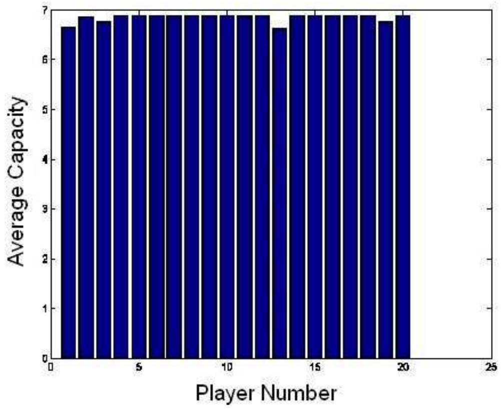

Fig. 4. The performance of 20 evolved strategies when played off against each other in a round robin tournament, with 1000 games per match.

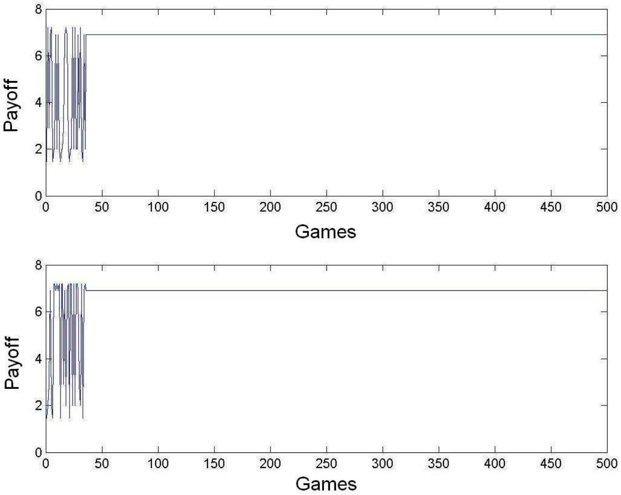

Fig. 5. The payoffs of two different winning strategies, when played off against each other in a series of 500 games. The initial stage of varying payoffs is where each 'probes' the other strategy and eventually they decide to cooperate mutually and stick to different channels, giving each a payoff of 6.9.

evolved strategies. However, although ostensibly different in structure, they seem to share some properties that make them perform well - as we shall see in section V. We begin by playing 20 of these various evolved strategies against each other in a round-robin tournament, where each strategy plays a match of 1000 games with every other strategy in the population. We then calculate each player's average score per game. The results of this simulation are shown in FIGURE 4. The important point to note is that the average scores of the players (calculated as channel capacities) approach 6.9, which is precisely the limit that we hoped to achieve (refer TABLE 2). This limit is achieved when the two players work out a 'deal' to stay in separate channels.

FIGURE 5 shows the variation of scores for two players in a series of games. In the initial few games players seem to gauge the other, engaging in a negotiating process before coming to the conclusion that segregation in signal space is the best option which is then maintained.

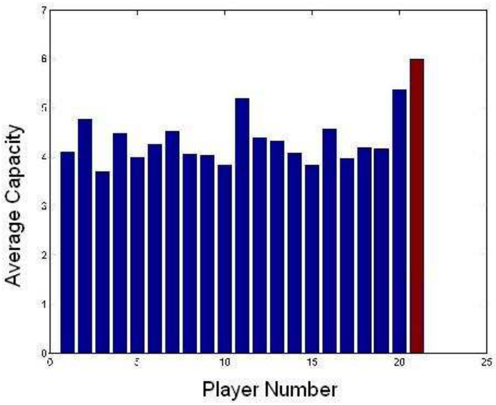

Fig. 6. The results of a round robin tournament. The first 20 bars are the average payoffs of arbitrary players - players with genomes that are random combinations from $\{0,1,2,3\}$. The last bar is a player that resulted from our genetic algorithm. Note that it scores the best, but not as high as if it were playing against a fellow intelligent winner.

Thus we see that when played among themselves, our evolved strategies do very well and approach the theoretical limit to maximize the collective capacity. However, in practice we cannot assume that our strategies will be interacting with other "like-minded" strategies. It is then natural to inquire about how one of our evolved players will perform in a population of arbitrary strategies - players with genomes that are random strings of the genetic alphabet. The results of these simulations are illustrated in FIGURE 6. In these results, we see that our evolved player outscores the rest of the arbitrary players but ends up with an average score of a little below 6 (instead of around 6.9, as in FIGURE 4. The reason for this is that in order for two players to realize that segregation is the optimal solution, *both* the players must have some degree of intelligence. In this case we have one intelligent player matched against arbitrary strategies. These arbitrary players do not recognize the optimality of segregation and hence do not encourage it, resulting in a lower score throughout the population. However, the fact that our evolved player scores the highest in an arbitrary population is evidence to show that it is a robust strategy which is not easily exploited and scores well against arbitrary strategies. This robustness was seen in every simulated round-robin tournament against arbitratily chosen strategies.

V. Policy Features

Each run of the genetic algorithm ends up with a final 'intelligent' population, in which all players do approximately equally well when played off in a round robin tournament against each other. For our analysis we group 4 sets of such winners, each set being a final intelligent population of 20 players from a genetic algorithm. On these 80 winners, we conduct some simple statistical investigation to identify certain defining features of all these policies. The objective in mind is to isolate dominant characteristics and then check if these dominant characteristics are sufficient to define a 'good' strategy.

A histogram was plotted showing the relative frequency of the various actions (represented by colors) at each posi-

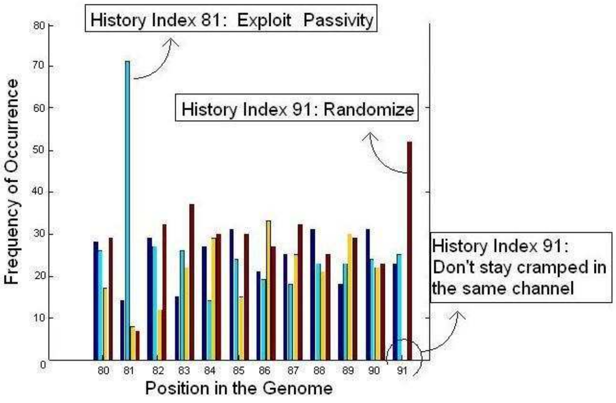

Fig. 7. A bar-graph showing the relative frequencies of action-choices for each position in the genome. A distinct spike or a distinct absence of a color indicates a characteristic that runs common to all 'intelligent' genomes.

tion in the genomes of our intelligent strategies. At certain positions of the genomes, it was found that certain colors were the clearly dominant choice among our 'good' genomes. This would mean that for that particular history (corresponding to that position in the genome), almost all our intelligent strategies choose to respond with the same action.

FIGURE 7 shows a section of the histogram in which we can see the dominance of the color Cyan at the 81^{st} position in the genome. Recall that each position in the genome corresponds to a particular possible outcome in the (finite) past of a player, and each color in the genome corresponds to the action that a player would take in response to that outcome in the past. According to the indexing scheme that we used to index all possible outcomes over the past two games, the 81^{st} position in the genome corresponds to the following:

- **One game back**: Our player spreads, Opponent plays in channel 2
- **Two games back**: Our player spreads, Opponent plays in channel 2

The histogram shows us that the action taken by almost all our intelligent strategies in response to such a history is to spread their energy over both channels - an action that we represent by the color Cyan. Thus, if an opponent does not retaliate then our intelligent strategies will continue to exploit that passivity. This observation identifies a dominant characteristic of an intelligent strategy. Similarly other such spikes in the histogram are identified, corresponding to other dominant response characteristics of intelligent strategies. We call these dominant characteristics 'schema'.

In the same way, at certain positions of the genomes of our intelligent strategies, certain actions are conspicuous by their absence, indicating that these actions are to be avoided as a response to that particular history. In FIGURE 7 we see that the color yellow is completely absent as a choice in the 91^{st} position of our intelligent genomes. The 91^{st} position corresponds to the following outcomes in the past:

- **One game back**: Our player plays in channel 2, Opponent plays in channel 2
- **Two games back**: Our player plays in channel 2, Opponent plays in channel 2

The histogram shows us that our intelligent strategies completely avoid playing in channel 2 (Yellow) as a response to this history. This shows that they try to avoid staying cramped in the same channel as the opponent. Similarly, other such schema are identified by their 'conspicuous absence'. For instance, position 11 which is the mirror image of 91 with channel 1 substituted for channel 2 shows similar characteristics.

Our sets of schema comprise the basic traits that make up an intelligent genome. Each of them contributes a certain aspect to the character of an intelligent player - understood by looking at what the position in the genome corresponds to (in terms of past outcomes) and the actions taken in response. Some of these characteristics, when abstracted in colloquial terms, are

- **Segregation** - Stay on your side of the fence.
- **Robustness to exploitation** - Push me, I push you back
- **Exploit passive opponents** - No mercy for the meek
- **Occasionally forgive to foster a spirit of cooperation**
- **Randomize** - occasionally, to avoid repeated collisions in signal space

VI. The Schema Skeleton

We now turn to the driving question behind the analysis of the preceeding section - can we use these handful of schema to construct strategies that perform like the intelligent strategies from our genetic algorithms? To find out, we construct a schema 'skeleton' for our genome - a genome in which the schema positions were fixed according to the observations of section V. The remaining positions in the genome were randomly filled up with colors from the genetic alphabet. Thus we constructed a genome that had all the schema we had observed, with the remaining parts of the genome being chosen randomly. We then proceed to test the performance of this constructed strategy by playing it in a round-robin tournament against other strategies that evolved out of our genetic algorithms. In this round robin tournament, each player plays a match of 1000 games with every other player. The average scores of the players are shown in the bar chart of FIGURE 8, with the last bar (shown in red) indicating the average score of our constructed strategy.

We see that the performance of the constructed strategy is about the same as that of the evolved strategies, reaching the socially optimal score of 6.9. The implication of this result is that *a handful of schema suffices to make a strategy intelligent* - leaving the other parts of the genome to be chosen freely.

VII. Conclusion

Our experiments and observations show that in our simple two-player symmetric game model, it is possible to identify salient features of strategies that are robust and perform well. Moreover, these salient features are a sufficient set of features to ensure a high performance level in the repeated game. Looking forward to possible implementation ideas, there could be two broad ways to use the ideas in this paper: the first would be to hardcode radios

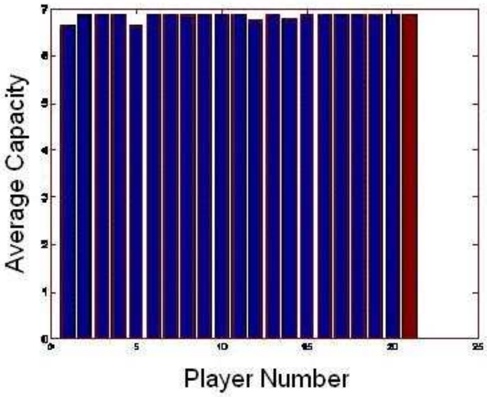

Fig. 8. The average scores of players in a round-robin tournament of 1000 games per match. All the players in this simulation evolved from our genetic algorithm *except* the last bar (in red), which was constructed based on the observed schema of our intelligent strategies.

with strategies that were evolved beforehand with all the salient features of robustness and intelligence. This would not require the radio to deal with any of the genetic algorithms computation, since it is fitted with a ready-to-go strategy. The *other* option, however, is to let the radio run its own genetic algorithm using the radios around it (its competitors, so to speak) as an evaluator set. The point is that our genetic algorithm evolves strategies that perform optimally against the evaluator set. What we did in our simulations was to make our own evaluator set with a fairly eclectic selection of strategies, so that the evolved strategies would be robust in any operating environment. But if the evaluator set were to be finely tuned to reflect the actual operating environment, then the strategies so evolved would be best suited to perform in *that* operating environment. For instance, in trials where the evaluator was a policy that behaved randomly, independent of history, optimal strategies always spread their energy equally over the channels.

The overhead with such a scheme, of course, is the computational complexity and the time involved in generating strategies - not to mention the need of re-running the evolution process everytime the operating environment changes. This begs the question of whether strategy evolution is a good operating principle for mutually interfering radios “in the wild” or whether an “offline teaching” strategy is best. In addition, as deeper history and more players are added, strategy complexity – and hence the size of the genetic description of the strategy – grows rapidly. We are currently studying efficient representation of strategies with deeper history and more than two players.

References

[1] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 1991.

[2] O. Popescu. *Interference Avoidance for Wireless Systems with Multiple Receivers*. PhD thesis, Rutgers University, Department of Electrical and Computer Engineering, 2004. Thesis Director: Prof. C. Rose. In progress.

[3] A.B. Carleial. Interference Channels. *IEEE Transaction on Information Theory*, 24(1):60–70, January 1978.

[4] A.B. Carleial. Outer Bounds on the Capacity of Interference Channels. *IEEE Transaction on Information Theory*, 29(4):602–60, July 1983.

[5] A.B. Carleial. A Case Where Interference Does Not Reduce Capacity. *IEEE Transaction on Information Theory*, 21:569–570, September 1975.

[6] R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1985.

[7] M. A. Nowak and K. Sigmund. Tit for tat in heterogeneous populations. *”Nature”*, (355):250–253, 1992. (also available at http://www.ped.fas.harvard.edu/pdf_files/Nature92b.pdf).

[8] M. A. Nowak and K. Sigmund. A strategy of win-stay, lose-shift that outperforms tit-for-tat in the prisoners dilemma game. *”Nature”*, 364(6432):56–58, 1993.

[9] M.A. Nowak, R.M. May, and K. Sigmund. The arithmetics of mutual help. *”Scientific American”*, pages 76–81, June 1995.

[10] Holland J.H. Genetic algorithms. *”Scientific American”*, pages 66–72, July 1992. (also available at http://www.econ.iastate.edu/tesfatsi/holland.GAIntro.htm).

[11] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Kluwer Academic Publishers, Boston, MA, 1989.

[12] O. Popescu and C. Rose. Waterfilling May Not Good Neighbors Make. In *Proceedings of IEEE Globecom ’03*, pages 1766–1770, San Francisco, CA, December 2003.

[13] O. Popescu, C. Rose, and D.C. Popescu. Signal Space Partitioning vs. Simultaneous Water Filling for Mutually Interfering Systems. In *Proceedings of IEEE Globecom ’04*, pages 1766–1770, Dallas, TX, November 2004.

[14] O. Popescu, D.C. Popescu, and C. Rose. Greed Interference Avoidance in Non-Collaborative Multi-Base Wireless Systems. In *39th Conference on Information Sciences and Systems – CISS’05*, Baltimore, MD, March 2005.

[15] O. Popescu, C. Rose, and D.C. Popescu. Strong Interference and Spectrum Warfare. In *38th Conference on Information Sciences and Systems – CISS’04*, Baltimore, MD, March 2005.

[16] I. J. Mitola. Software radios: Survey, critical evaluation and future directions. *”IEEE Aerosp. Electron. Syst. Mag”*, 8:25–36, April 1993.

[17] Federal Communications Commission. Et docket no. 03-322, 2003.