A TOURNAMENT OF PARTY DECISION RULES:
MATERIAL FOR ELECTRONIC APPENDICES

James H. Fowler
University of California, San Diego

Michael Laver
New York University

APPENDIX A: TOURNAMENT RULES

OVERVIEW

In the spirit of Robert Axelrod's tournament for strategies in the repeat-play prisoner's dilemma game, we announce a tournament for party strategies in a dynamic agent-based spatial model of party competition. The winner will receive $1000. We herewith call for submissions of party strategies by 15 April 2006. Each strategy submitted will be pitted against each other strategy in a series of long-running simulations of a two-dimensional multiparty spatial model in which the number of parties is endogenously determined. All parties falling below a certain size threshold for two consecutive elections will "die". One new party will be "born" each election at a random spatial location, using a strategy randomly selected from the portfolio of submitted strategies. The strategy most successful at winning votes over the very long run will be declared the "winner".

THE ENVIRONMENT OF PARTY COMPETITION

The policy space

The policy space will be defined by 1000 voters with two-dimensional ideal points randomly drawn from a standard normal distribution with a mean of zero and a standard deviation of 1 in each dimension (a bivariate normal distribution). Voters never abstain and always vote sincerely for the party specifying the policy position closest to their ideal point.

Periods and elections

Each simulation in the series of simulations (see below) will run for 200,000 periods, after discarding results from 20,000 initial periods ("burn-in"). An "election" will be held every 20 periods. The period immediately after each election will begin with the "birth" of a new party, using a strategy randomly selected from the set of strategies submitted for the tournament. The initial policy position of each newborn party will be determined by a random draw from a normal distribution with a mean of zero and a standard deviation of 1 in both dimensions.

Each period

At the start of each period, each party strategy, subject to the information requirements set out below, must specify a policy position for the party. (The initial policy position in the first period after party birth will be random – see above). In the middle of each period, each voter supports the party specifying the policy position closest (in terms of Euclidean distance) to the voter's ideal point. At the end of each period, the policy positions of each party and the number of voters supporting each party are announced. As with an opinion poll, this announcement provides information to parties that they can use to adapt their position to the current competitive environment. Each party thus has 20 periods to adapt before each election.

Each election

At the end of every twentieth period, an election is held. The number of voters supporting each party is announced and recorded. If a party falls below a survival threshold of 10 percent of votes cast for two consecutive elections, it "dies" and disappears from the competition. Note that more than one party may die in an election, but only one new party is born immediately after each election. After votes have been recorded and party deaths have taken place, a new party is "born" at the beginning of each period following an election. The new party will use a strategy randomly selected, with equal probability, from the set of strategies submitted for the tournament, and will be given a random initial policy position (see above). (The only exception to this is that the first party selected for the first period of each simulation will be rotated between strategies - see below).

The series of simulations

To determine the winner of the tournament, we will analyze results from 5n simulations, where n is the number of strategies submitted. Each submitted strategy will be identified as the "starting" strategy for five simulations to be sure initial conditions are not influencing the tournament results. The simulation environment is programmed in R. Code for the simulation test-bed is available at http://jhfowler.ucsd.edu/tournament.htm. Each simulation in the series will run for 220,000 periods, with the first 20,000 periods (the "burn-in") discarded from the analysis. In preliminary testing of the simulation

environment, using the four "pre-submitted" party strategies (see below), this is about five times as long as was necessary to reach convergence as determined by standard statistical tests. Each submitted strategy will thus start "first" in simulations running for a total of 1,000,000 periods (50,000 elections) after burn-in.

CONSTRAINTS ON PARTY STRATEGIES

Party decision rules must specify a party position for the beginning of each period using only the following information:

1. Any previously-announced policy position and level of voter support for any party.
2. The mean or median position on each dimension of the ideal points of the party's own current supporters.
3. Note that party strategies cannot have access to the ideal point of any individual voter, nor can they interrogate voters about what they would do in hypothetical situations.
4. Note also that this implies that, on being born, a party using any strategy does not know its initial position in the space, and has no data about the state of the system in previous periods.

See http://jhfowler.ucsd.edu/tournament.htm for a table of variables that correspond to this information that have already been programmed in R.

PRE-SUBMITTED STRATEGIES

There will be four "pre-submitted" strategies, not eligible for the prize. These will be the four strategies described in Michael Laver, "Policy and Dynamics of Political Competition", *American Political Science Review*, 99:2 (May 2005) 263-281, viz:

1. STICKER – the party does not move from its original position;
2. AGGREGATOR – the party sets policy at the mean position on each dimension of the ideal points of current party supporters;

3. HUNTER – If party support in the previous period was higher than in the period before that, then the party moves a step of 0.05 in the same direction it moved previously. Otherwise it randomly chooses a direction in the opposite 180 degree half space from its previous move and moves a step of 0.05 in that direction;
4. PREDATOR – the party observes the current sizes and policy positions of all parties. If it is the largest party it stands still, otherwise it moves a step of 0.05 towards the position of the largest party.

RULES FOR SUBMISSION

Each party strategy submitted must be fully specified so it can be programmed. Since the tournament will be run in an environment programmed in R, strategy submissions will be particularly welcome if they are specified in the form of program code snippets in R. However, any strategy submitted in the form of a clearly written and programmable algorithm, or programmable piece of pseudo-code, will be accepted. Please suggest a short and informative name for your party strategy. Only one submission is allowed per person. If several entrants independently submit an identical strategy, then this strategy will be entered only once in the tournament. If this strategy wins the prize, a single winner will be chosen by lot from among those who submitted the winning entry. All entries must be received by 15 April, 2006. Send them to jhfowler@ucsd.edu.

WINNERS

The prize of $1000 will be awarded to the strategy winning the most votes in all elections over the series of $5n$ simulations described above. That is, for every election in every simulation after burn-in, the total votes won by all parties using each strategy will be calculated. The winner of the tournament will be the strategy accumulating the most votes over the entire suite of simulations. Although these strategies will not win prizes, we will also calculate and report the strategy winning the most votes per party using each strategy over the suite of simulations, as well as the strategy used by parties surviving, on average, for the largest number of periods. The prize will be awarded at American Political Science Association's 2006 annual meeting in Philadelphia, PA.

APPENDIX B: METHODOLOGICAL ISSUES RELATING TO MODEL BURN-IN AND THE LONG-RUN CONVERGENCE OF RESULTS

Length of model "burn-in"

The length of our simulation runs was determined prior to the tournament. Using the four pre-entered rules, we conducted 20 simulations (in which each of the four rules was used five times by the first-born party) and let the simulation run to 100,000 periods (5,000 elections). We studied trace plots and used diagnostics to evaluate convergence in the party vote shares at each election. The Heidelberger convergence diagnostic, which uses the Cramer-von-Mises statistic to test the null hypothesis that the sampled values come from a stationary distribution, showed some nonstationary effects on the distribution up to 10,000 periods (500 elections) but none thereafter, so we chose to discard the first 20,000 periods of each chain. We also applied a Brooks-Gelman test to the set of simulations (Brooks and Gelman 1998). This indicates the point at which the output from all chains is indistinguishable, and thus at which initial values no longer influence the results. This test is based on a comparison of within-chain and between-chain variances, and is similar to a classical analysis of variance.[1] The 'potential scale reduction factors' for each variable in our simulations were reduced to less than 1.1 by the 40,000th period, indicating convergence. However, to be safe, we decided to extend the simulation length to 200,000 periods. Longer runs of up to one million periods yielded identical results.

Statistical convergence of simulation results on a single winning outcome

Particularly given the large number of different decision rules under review, it is clearly very important to be sure that we ran "enough" simulations, in the sense that a different statistical inference about the winner would have been very unlikely had we run more simulations. To investigate this we revisited the convergence criteria we established prior to the tournament and conducted Brooks-Gelman tests for each total vote outcome for each party for each set of five chains starting with each of the 29 rules in the tournament.

[1] Each chain starts at an over-dispersed starting point different from other chains. Between-chain variance should start high and *decrease* while within-chain variance starts low and *increases*. When the ratio of these variances is close to 1 then it suggests that the chains have reached their stationary distribution.

Recall that these tests indicate the point when output from all chains is indistinguishable. They are based on a comparison of within-chain and between-chain variances, and are similar to a classical analysis of variance. Brooks and Gelman (1997) recommend potential scale reduction factors less than 1.2 and every one of our 841 test statistics was significantly less than 1.1 at the 95% confidence level. We also applied Heidelberger convergence diagnostics, which indicated the chains had reached a stationary distribution.

APPENDIX C: TOURNAMENT ENTRIES

ID #	Name	Author	Description
5	Average	Grynaviski, Jeff	Adopt the weighted average of all parties' locations, where the weights are given by the number of votes given to the party in the most recent round.
6	Avoider	Thomas Plümper Martin, Christian	Given the calculated weighted mean of party distributions $\overline{x}, \overline{y}$, let our party take the following position $x = \overline{x} + a \cdot b$ $y = \overline{y} + \sqrt{(1 - a)} \cdot c$ where *a* is drawn from a uniform distribution of the interval [-1,1] and *b* and *c* take the value -1 or +1 with equal probability.
7	Bigtent	Curran, John P Daryl Posnett	The algorithm consists in two parts. The first forms an estimate of voter distributions, and the second searches for a party position that is optimal subject to certain constraints. To estimate the voter distribution, a bounding box is determined that contains all current and historical party positions, including those of extinct parties. The box is partitioned into a coarse grid, and the current position for each active party is mapped to the nearest grid point. Thus, each grid point is assigned a fraction of total votes based on the most recent poll. Each grid point maintains a finite-horizon moving average of the vote fraction assigned to it. These historical averages are used as the estimate of the true voter distribution in the search stage. In the search stage, the algorithm tries to find a position within the bounding box that will give its party the largest fraction of votes, assuming (i) that the voter distribution is given by the estimate calculated in the first part of the algorithm, and (ii) the other parties do not move from their most recently revealed positions. Given a candidate position for its party, the algorithm computes what fraction of votes the party will receive if (i) and (ii) hold. The candidate position is then revised using a standard optimization routine. (The submitted version uses R's optim routine with the BFGS option, which is a modified Newton's method.) Ideally, several initial points are tried within the bounding box, due to the possibility of local maxima. Because finding

			the optimum is computationally expensive in the context of the competition, however, the length of the search has an upper limit that probably stops short of finding a local maximum.
8	Center-mass	Zimmerman, Matt	Move one-twentieth of the distance towards the weighted average of the parties each period.
9	**Fisher****	Mayer, Alex	A party using the fisher strategy fishes by taking steps at near right angles to its last step (random angles within 45 degrees of a right angle). At each position the level of voter support is recorded. Overview: When a party is born with the fisher strategy, it stores its position and level of voter support, then moves to a policy position calculated to be approximately the mean voter position. From this position, it fishes throughout the policy space taking large steps. During the 11th period (just before 3/5 of the total number of periods in an election cycle - election frequency [efreq] = 20 periods), the party identifies the 4 (efreq/5) policy positions that yielded the most voter support during the initial fishing periods. During this period and the next 3 periods (total of efreq/5), the party revisits those positions and records voter support at each. During the 15th period (just before 4/5 of efreq), the party returns to the top position among the 4 re-visited positions. If that position yields voter support below the threshold for survival, the party returns to the second-ranked position from among the 4 re-visited positions. Once the party has chosen the first or second ranked position, it fishes with very small steps (5% of the step size at birth), starting from that position. During the 19th period (the period immediately before the election), the party moves to the position that produced the most voter support during the last 4 periods (these include at least 2 fishing periods). If the party's vote total in an election satisfies the threshold for survival, then the party reduces the size of steps it takes to test the policy space to 20% of the size it used at birth, staying relatively near the successful position. If the party's voter support at an election drops below the threshold, it instead uses the original step size used at birth until it finds a position with support that satisfies the threshold.
10	Follow-the-Leader	Kasdin, Stuart Glasgow, Garrett	The first period after being born (the second period in the game), calculate the shortest Euclidian distance between the two leading vote gainers from the previous period. Move OWNPARTY to the point midway along the shortest line in Euclidian space between these two leading vote gainers. After each period, move OWNPARTY 0.05 toward the party with

			the highest level of support in the most recent period parallel to the axis on which the distance between it and OWNPARTY is greatest. If this increases vote share from the previous period, move again next period 0.05 along the same axis in the same direction. Do not move on the other dimension. When OWNPARTY no longer gains electoral support after a period ends, in the next period move 0.05 along the other axis, so move on x if OWNPARTY's last move was on y, or y if OWNPARTY's last move was on x. If OWNPARTY is the strongest party do not move.
11	Fool-proof	Adams, Jim	Shift to the weighted mean position of all the parties' positions. At each period t, employ the following two-step procedure to determine the policy position: First, compute the average policy shift of all rival parties at the previous period, compared to the period before that, with these shifts not weighted by the parties' vote shares. Thus at time t, for instance, this calculation might yield the conclusion that between periods t-2 and t-1 the rival parties shifted on average .05 units to the left along the horizontal axis and .15 units downward along the vertical axis, or whatever. Second, the party's decision rule at time t is to shift its position in the same direction as the rival parties shifted between periods t-2 and t-1, but the focal party's shift will be of only one fifth the magnitude of the average shift of the rival parties. So for instance in the above example, where the rival parties shifted on average .05 units to the left along the horizontal axis and .15 units downward along the vertical axis between period t-2 and t-1, the focal party at time t would shift .01 units to the left on the horizontal axis and .03 units downward along the vertical axis.
12	**Genety****	Larson, Eric Cook, Tyson	Weighs three factors to choose where to move: 1) location relative to the origin, 2) aggregation, and 3) hunting. Party calculates the vector toward the origin, the vector toward the mean position of party supporters, and the vector emerging from applying the pre-submitted hunting algorithm, and moves according to a weighted sum of these vectors. The static weights applied to these vectors were determined by applying a genetic algorithm, and were optimized in competition with: 1-4) The four pre-submitted strategies 5) A brownian motion strategy 6) A random bivariate normal draw strategy 7) The hole-finding algorithm (developed by another

			Kalamazoo College student, Joel Haas) 8-11) Four different genetic algorithm parties 12) A circling algorithm that stayed outside the radius of the farthest party from the origin 13) A party that oscillated inside the r=1 circle but outside the r=0.8 circle Each generation of the genetic algorithm had approximately 100 candidate strategies, run against each other in randomized order in sets of four as above.
13	Half-aggregator	Williams, Daniel	Arbitrarily select one dimension, call it X. With respect to dimension X, the party selects a position that is at the mean ideal point of its current party supporters. After the first round, dimension X remains fixed. Call the other dimension, dimension Y. With respect to dimension Y, if party support in the previous period was higher than in the period before that, then the party moves a step of 0.05 in the same direction it moved previously, otherwise it moves 0.05 steps away.
14	Insatiable-predator	Nyblade, Ben	The basic strategy is identical to that of Predator, but Insatiable Predator never stays in one spot. The only change to the code is what Predator does when it is the largest party. Instead of staying put like Predator, the Insatiable Predator moves .05 towards the nearest other party when it is the largest party.
15	Jumper	Smirnov, Oleg	Party policy is located on the straight line formed by the locations of the two parties with the highest vote totals on the previous round. Regardless of whether or not it is the largest party, Jumper occupies a point on this line such that the former largest party is in between the jumper and the 2nd place party. The distance between the former largest party and the jumper is 0.5 of the distance between the winner and the 2nd place party.
16	**KQ-strat****	Quinn, Kevin	If alive for 3 iterations and votes < pthresh for 3 iterations or more and haven't moved more than 1.46e-4 units in 3 iterations or more then set new position equal to p + e, where p is the position of a randomly chosen party that did not previously move exactly 1.46e-4 units and whose current vote total is greater than pthresh and e is a N(0, .0625^2) random variable. Otherwise move 1.46e-4 units in random direction.
17	Median-voter-seeker	Gartner, Scott	The party locates the dimension-by-dimension median of the policy space, defined by the location and vote share of the parties in the previous period. It moves a step of 0.05 in that

			direction.
18	Satisficing Explorer	Zick, Ken	"First employ a satisficing rule: if my vote total was at or above the election threshold (e.g. 10%) on either of the last two periods (periods, not elections), then maintain the current position. Otherwise, explore the space, as follows: 1) Create a list of all parties that are at or above the election threshold on the current period 2) Randomly choose one party location from this list 3) Jump to a location that is 0.3 units away from the chosen party location. The angle between my new location and the chosen party location is random." **NOTE: (9/1/2007) This strategy was not programmed correctly in the tournament, due to a mistake James Fowler made when translating code sent by Ken Zick. Fowler inadvertently left out the first 'satisficing' stage of the author's strategy. Zick emailed Fowler about the problem, but his email did not reach him because Fowler had changed institutions. Given the success of other satisficing strategies, we think the correct implementation of Zick's stratgy would have made the strategy perform much better.**
19	Niche-hunter	Money, Jeannette	If party support in the previous period was higher than in the period before that, then the party moves a step of 0.05 in the same direction it moved previously. Otherwise, it examines the party support of each ADJACENT party in the two prior periods (that is, it compares the electoral support at t - 1 with the electoral support at t - 2). It moves a step of 0.05 in the direction of the ADJACENT party with the largest gains (smallest losses) in party support. To identify the adjacent party in a two-dimensional space, generate a four quadrant grid centered on the NICHE HUNTER party's policy position. Unless a quadrant contains no parties, there will be one adjacent party in each quadrant, which is the party closest to the NICHE HUNTER in Euclidian distance. Using this definition, the NICHE HUNTER may have 1-4 "adjacent" parties. If the "adjacent" party with the largest electoral gains is less than 0.05 units away from the NICHE HUNTER, then the NICHE HUNTER moves one half of the distance between itself and the "adjacent" party.
20	Niche-predator	Andrews, Jo	The party observes the current sizes and policy positions of all ADJACENT parties. If it is the largest party it stands still; otherwise, it moves a step of 0.05 toward the position of the

			largest ADJACENT party. If the largest "adjacent" party is less than 0.05 units away from the NICHE PREDATOR, then the NICHE PREDATOR moves one half of the distance between itself and the "adjacent" party. To identify the adjacent party in a two-dimensional space, generate a four quadrant grid centered on the NICHE PREDATOR party's policy position. Unless a quadrant contains no parties, there will be one adjacent party in each quadrant, which is the party closest to the NICHE PREDATOR in Euclidian distance. Using this definition, the NICHE PREDATOR may have 1-4 "adjacent" parties.
21	Over-under	Lo, James	Go to (x_i+10, y_i+10) until election period, where x_i and y_i are the coordinates of the position of the highest vote getter in the previous election. In an election period given random perturbation e~U(0, 0.05), go to (x_i+e, y_i+e) or (x_i-e,y_i-e), alternating between going just "above" and just "below".
22	Parasite	Schleicher, David	For 19 periods, the policy position reflects my initial position. For the 20th period (the election period), I observe the largest party and move to a position that is .01 away from the largest party's position in a random direction.
23	Patchwork	Guarnieri, Fernando	As predator, identifies largest party and moves towards it, then becomes an aggregator if it is the largest party.
24	**Pick-and-stick****	Schulz, Evan	In the periods before the first election, locate party at random points in the space. Then return to the point at which it received most votes and stay there for subsequent elections.
25	**Pragmatist**	Calvo, Ernesto	Combination of the vote-weighted mean location of all parties and the party's own median voter. A normally distributed noise term prevents parties with the same strategy from overlapping with each other. $ploc_i = \alpha\left(\sum_j v_j ploc_j + e\right) + (1-\alpha)* pmloc_i$ where α=.2, 0≥v_j≥1, and e~N(0,.2).
26	Raptor	Jones, David Hugh	The strategy is based on the pre-submitted Hunter rule but with two differences. 1. Each turn, the party moves a distance of 0.178 rather than 0.05 2. Each turn, the party calculates the distance all other parties have moved. If any of these other parties has also moved a distance of 0.178, is now less than 0.3 away from the first

			party, and is weakly to the right of the first party, then the party behaves as if its votes had decreased, ie moves randomly backwards in the opposite 180 degree half space to its previous direction.
27	**Shuffle****	Takeuchi, Kan	This behaves like Aggregator if it won 11.5% or more votes in previous round, Hunter if won 11.5%-8.0%, or it relocates itself into the least populated area as follows: Divide the space into 4 quadrants around the center of the gravity, which is derived from ploc and ptotals, and find one in which the votes per party is the highest; then jump to any point there randomly using rnorm function.
28	**Sticky-hunter-median-finder****	Bramson, Aaron	If party receives at least 10% last election, then use Sticky Hunter, else Median Finder. The Sticky Hunter Strategy: If the party collected less than 10% of the votes in both of the last two periods then it steps according to the following rule. If its last step improved its percentage of votes then step forward a distance of 0.135. If the last step did not improve its vote percentage then pick a new direction from the first and last third of the opposite 180 degrees (i.e. cutting out the middle third that includes a total U-turn) and make a 0.135 step. If the agent did collect at least 10% of the votes last turn then do nothing (i.e. stick). The Median Finder: Each period, determine the median position of my supporters, and move to that location.
29	Zeno	McGovern, Geoff	Like PREDATOR, "the party observes the current sizes and policy positions of all parties. If it is the largest party it stands still." Otherwise, ZENO will move half the distance between itself and the largest party in the direction of the largest party.

APPENDIX D: SUUPLEMENTARY FIGURES (TOURNAMENT RESULTS WITH SECRET HANDSHAKES ENABLED)

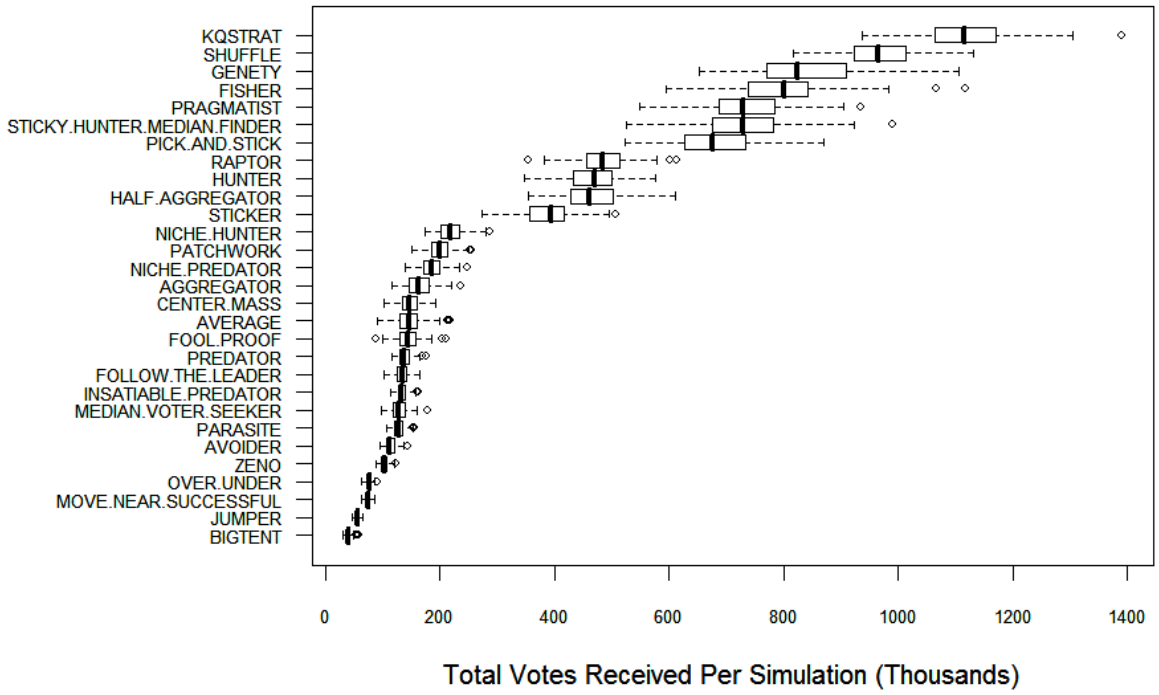

Figure D1: Total votes received, By rule

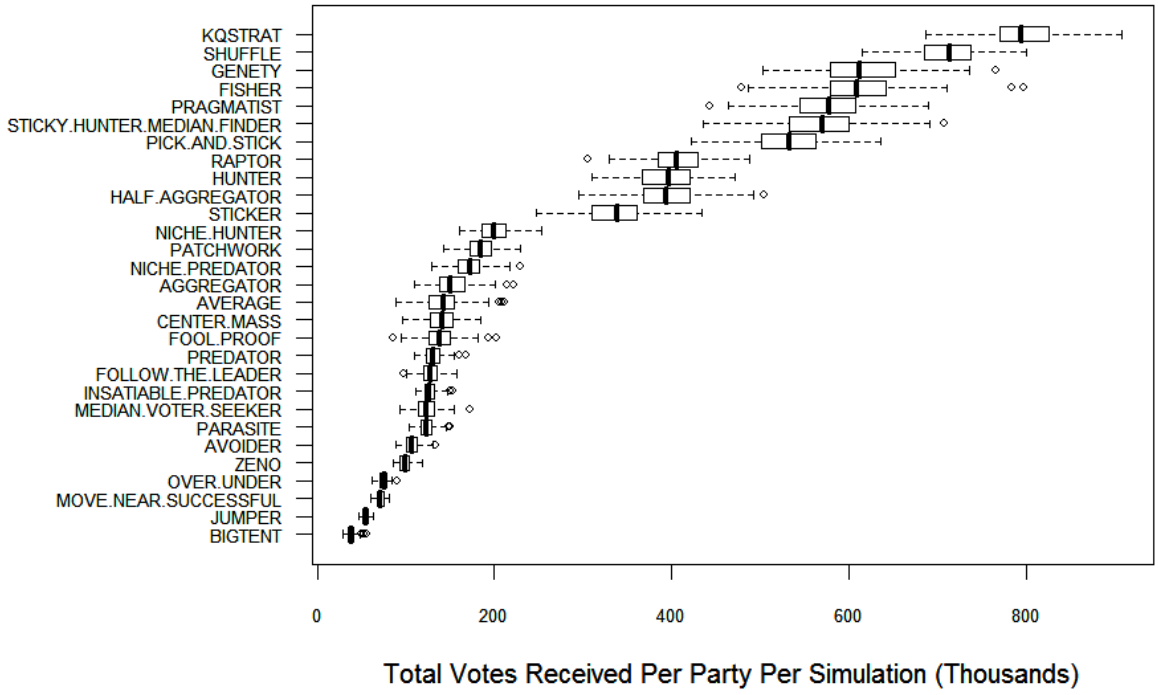

Figure D2: Total voted received, per party-using-rule

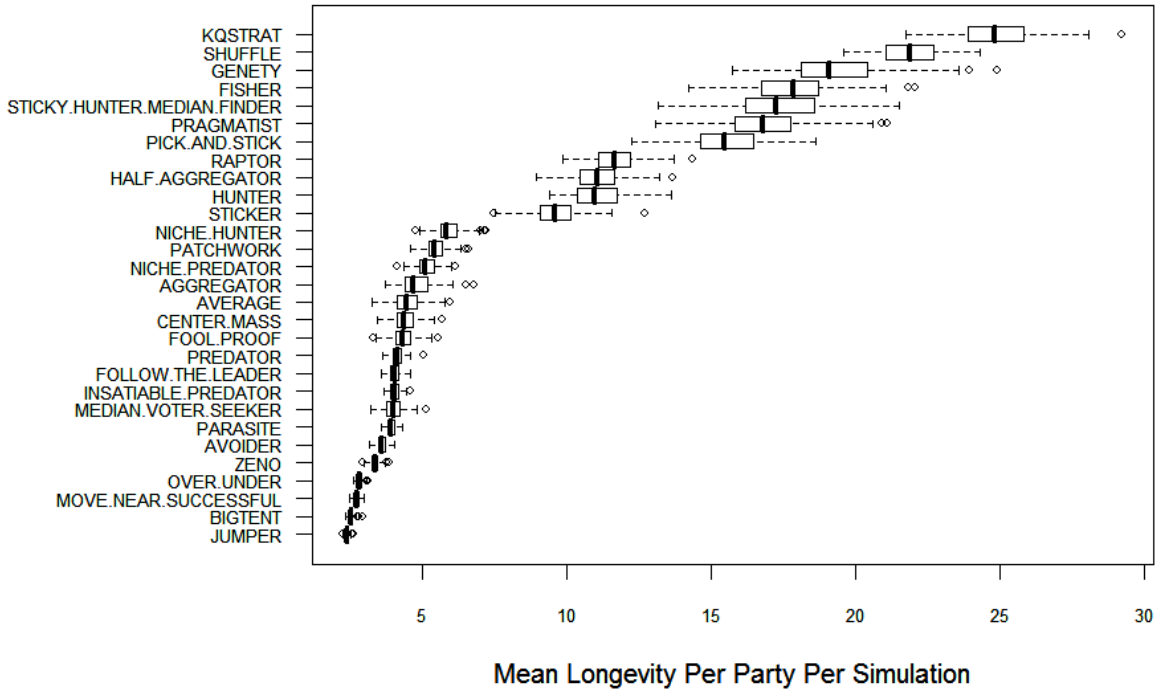

Figure D3: Mean number of elections survived by each rule