[[1]](#footnote-2)

Better Allocation to Reduce Voting Queue Length

Enas Mohamed, Hesham Naiem, and Mostafa Izz,

(em\_cmp\_eng , hesham.naiem)@yahoo.com),mostafa.3ez@gmail.com,

Faculty of Engineering, Cairo University

*Abstract*— Providing high turnout rate in elections is a desirable goal to reach democracy. In this paper we are extending some literature studies that were trying to provide equity between voters across all precincts which will result in reducing maximum waiting time between voters. In this paper we are extending the simulation model used in literature to study voting queues and in the same time we are comparing the results of the state of the art technique in distributing voting machines across precincts (namely GIA which is simulation based greedy algorithm) with our simple algorithm (RA which is Random Algorithm based on some heuristics for improvements) on a data conducted from 2004 US presidential election in one county.

*Index Terms*— Voting Queues, Simulation

# INTRODUCTION

Election administration is a service provided to the voting public. As with other service systems, waiting lines are non-value added activities and waiting time is viewed negatively. As discussed in Mebane (2005), waiting times to vote are directly correlated to voter turnout. As waiting times increase, more voters are forced to leave without having cast their ballots due to impatience and other time commitments. Thus, it is very important to design voting systems that result in voters waiting the least amount of time possible. Ideally, we would provide a sufficient quantity of voting machines such that voters would never have to wait to cast ballots. However, due to the expense of new voting machines such as direct-recording electronic

(DRE) machines (also known as “touch-screen” machines), election boards are limited in their ability to procure additional equipment.

Perhaps even more importantly, voting systems should provide equity, which means that we should not design systems that favor some voting groups (defined by geography, voting preference, etc.) by having shorter lines in some precincts than others. Such inequities have been a concern in recent elections. The Department of Justice investigated claims that the Board of Elections in Franklin County, Ohio, “systematically assigned fewer voting machines in polling places serving predominantly black communities as compared to its assignment of machines in predominantly white communities” during the 2004 election (Tanner 2005). While the Department of Justice ultimately found no evidence of systematic decisions to create voting inequities, their report does point to many factors, including specific measures of voter wait times and differences in voting patterns across precincts, which are typically not taken into account under current voting-machine allocation policies. Furthermore, Walter Mebane, Political Science Professor at the University of Michigan, has produced several studies criticizing the Department of Justice’s response to the issues in Franklin County. Professor Mebane concludes from his research, “The allocation of voting machines in Franklin County was clearly biased against voters in precincts with high proportions of African Americans” (Mebane 2006).

Determining an optimal voting-machine allocation is challenging and difficult to solve for several reasons. (1) Voters arrive randomly and according to non-stationary processes to polling locations. There are typically surges in voter arrivals during the morning, noon and evening times due to work schedules (Edelstein 2006). Moreover it is difficult to estimate voter turnout rates prior to Election Day because it depends on many uncontrollable variables (e.g., weather, composition of the voting ballot, etc.). (2) Voter queues may not reach steady state. Ohio law requires that the polls be open 13 hours, plus however much time is needed to accommodate voters waiting to vote at 7:30 pm. Given the limited amount of time that voting precincts are open, the voting queues may still be in a transient state. Such non-stationary arrivals and non-steady-state queues violate the fundamental assumptions of traditional queuing theory. (3) Actual voting scenarios involve considerable computational complexity. There are thousands of polling stations across Ohio and the input variables are stochastic. The result is a large-scale non-linear stochastic optimization problem. Thus, both building model and developing solution methods are challenging endeavors.

We model the voting process using a simulation model that allows us to employ non-stationary arrivals and non-steady state queues. We allocate voting machines to precincts using a greedy improvement heuristic. The objective in our machine allocation is to provide voter equity across precincts. The rest of the paper is arranged as follows. Section 2 provides a review of related literature. Section 3 introduces a performance metric for voter equity and discusses several analysis options for this problem. Section 4 describes the setup of the simulation model and the greedy improvement algorithm (GIA) implemented on the model. Section 5 systematically studies the performance of the simulation-based GIA through an experimental design. Section 6 presents conclusions and future work.

# Motivation

A referendum was held in Egypt on March 19 following the 2011 Egyptian Revolution. For most Egyptians, this was the first genuinely free vote in their lives, so there was a large turnout for the referendum which reached 41.2 % of the 45 million eligible voters. As a result, the queues outside polling stations have continued to grow as Egyptians go out to cast their votes and voters waited patiently for hours in lines. This historical step toward democracy needs to be studied for improvement in the future.

# Problem Definition

Election administration is a service to the voting public. And the waiting queues are a non-value-added activities, besides the long queues affect negatively the turnout rate of the voters due to long waiting times. And we need to motivate people to contribute in the elections in Egypt because we are developing a new democracy life. The administration process mainly determine the parameters of the Election Day from the distribution of different precincts, to the total number of Machines used for voting and the distribution of these machines between different precincts. We have taken the path of searching for best allocation of the voting machines between different precincts. We are extending a simulation model used to show that the algorithm described in [4] – namely GIA - outperforms UEM (Utilization Equalization Method) which have been used in US 2004 presidential election in many counties. In this paper we are extending the simulation model used in ‎[4] and comparing the results of our algorithm RA with the algorithm described in ‎[4] GIA.

# Suggested Solution

## Description of Basic Polls Queuing Model

### Model Logic

Our simulation model provides the expected waiting time in each precinct for a given number of assigned voting machines. The numbers of DRE voting machines assigned to each precinct are our decision variables.

On Election Day, all polling stations in Ohio open at 6:30 am and close at 7:30 pm. Once a voter arrives to his or her designated precinct, the voter joins a single queue until there is a DRE voting machine available. There are multiple DRE voting machines per precinct. We assume that all DRE voting machines are identical and shared by all voters within a polling station.

### Input Distribution Assumptions and Data Sources

We use a data set based on statistics from the 2004 election in Franklin County, Ohio (available at <http://copperas.com/fcelection/>). Specifically, we fit a normal distribution with mean 1070 and standard deviation 319 to the number of registered voters in each precinct in the 2004 election. For a given number N of precincts (N will be set as a factor in our experimental design below), we generate the number of eligible voters for each precinct independently from this fitted normal distribution.

To get the voter turnout rate we used the data based on statistics from the 2004 election in Franklin County, Ohio.

We fit a Weibull distribution with Shape Parameter α=6.9514 and Scale Parameter β=60.884 to turnout percentage in each precinct in the 2004 election.

We assume that the voting service time in every precinct follows a gamma distribution with shape parameter of 5.71. One level for this factor is set using data from the 2006 Ohio gubernatorial election by setting the gamma distribution scale parameter to 1.05. Allen and Bernshteyn (2006a) use a mean service time of voting of 3.33 minutes. We use this as the mean for the other level of our Voting Time factor. We do not have data for the voting times with mean of 3.33 minutes, but we assume such voting times also follow a gamma distribution with shape parameter 5.71 and set the distribution’s scale parameter to 0.58 to match this mean voting time.

The turnout rate is applied to this, and then the non-stationary arrival pattern is used to distribute these voters’ appearance at the polling station, as described below. We assume that each precinct has the same voter turnout rate.

Ohio Revised Code (O.R.C.) §3501.32 states that on Election Day the polls shall be opened at 6:30 am and shall be closed at 7:30 pm “unless there are voters waiting in line to cast their ballots, in which case the poll shall be kept open until such waiting voters have voted.” Ohio law thus requires that the polls be open 13 hours, plus however much time is needed to accommodate voters waiting to vote at 7:30 pm. Therefore, we allow all queues to clear, but we do not allow any additional voter arrivals after 7:30 pm in our simulations.

On election day, there are “peaks and valleys” of usage by voters depending upon the time of day, the weather, traffic and other variables outside of the control of election staff. Voters do not arrive according to a stationary arrival stochastic process. There are typically surges in voter arrivals during the morning, noon, and evening due to work schedules (Edelstein 2006). Precinct poll workers at the Hamilton County, Ohio, Board of Elections reported an early morning voter rush and lines shorter by day’s end on Election Day 2008. The Voting Experience Survey (Feldman and Belcher 2005; Mebane 2005), which is based on a sample of voters throughout Ohio, provides the percentages of turnout voter arrivals by the time of day (see Table 1). We assume that in each time period the number of arriving voters follows a Poisson distribution. The timing and size of these surges may not be the same across all precincts due to differences in voters’ socioeconomic status, but here we assume that all precincts experience similar arrival patterns.

Table 1: Voter Arrivals by Time of Day

|  |  |
| --- | --- |
| Period of Time | Percentage of Turnout Voters |
| Before 8 a.m. | 20.61 |
| 8 a.m. – 11 a.m. | 27.34 |
| 11 a.m. – 3 p.m. | 24.05 |
| 3 p.m. – 5 p.m. | 13.26 |
| After 5 p.m. | 13.87 |

Ohio law states that voters are allowed up to five minutes to place their vote (Anthony et al. 2004). However, anecdotal evidence suggests that this law is rarely, if ever, enforced. Actual voter service times will depend on the length of the ballot - in particular, the number of issues on the ballot, which generally require the most time for voters to read and on which to make a choice. To determine approximate voter service times we use data from the 2006 Ohio gubernatorial election with six issues on the ballot. We fit a distribution to the data of voting times read from a sample of Election Systems & Software machines in this election and found that a gamma distribution with the scale parameter of 1.05 and the shape parameter of 5.71 fits the data acceptably. We assume that voting times in every precinct follow this same gamma distribution.

For now, we assume that voting machines are perfectly reliable (i.e., there are no voting-machine failures). We also

assume that all available voting machines must be allocated to precincts as this is often the policy used in practice, and will reduce the system-wide voter wait time.

## The Greedy Improvement Algorithm

We use a heuristic solution method to allocate voting machines to precincts. Our method combines a simple greedy heuristic to reduce expected voter wait times with a local improvement search where we use (1) as our objective function. We refer to this solution method as the greedy improvement algorithm (GIA), which consists of two phases: Phase I is a simple greedy heuristic and Phase II is a local improvement search.

## The Random Algorithm

We create 4 combinations of machine distribution in precincts by using the random method, then we choose the best distribution that result in minimizing the max waiting time in all precincts, so the equity After that we combine this random method with some local search to improve the results.

Phase I: Iterative random method to find the best allocations in the precincts that reduce the max waiting time.

Step 1. Assign random values to for each except for (minimum value = 1, maximum value =)

Step 2. =

Step 3. If , then

Step 4. Run the simulation and calculate the max waiting time

Step 5. Go to step 1 and repeat 4 times, save each combination of

Step 6. Choose the combination with the min waiting time

Phase II: Iterative improvement that works as follow

Step 1. Add Machine to the precinct with the maximum waiting time and remove one from the precinct with the minimum waiting time

Step 2. Run the simulation and calculate the equity (new equity)

Step 3. Repeat while (new equity < old equity); otherwise stop.

## Implementation

It is not difficult to implement the GIA in a standard programming language (e.g., C, VBA, and C++) once the expected wait times in any precinct () are obtained. However, extensive effort is required to build simulation models to obtain the expected wait times using standard programming languages. On the other hand, it is easy to obtain () in simulation modeling software (e.g., Arena), but difficult to implement the GIA, because simulation-modeling software is not designed as a programming language.

We preferred to use a general purpose programming language to implement the simulation model (namely C# .Net) with the help of open source .Net simulation library called React..NET [10].

# [Example]

# Performance Measurement and Analysis

## Performance Metric

Some performance metric used in simulating elections is minimizing the total expected waiting time across all precincts, but it leads to long voter waiting times in some precincts and short voter waiting time in other precincts. This is not preferred since we need to make all voters are experiencing same situations across all precincts.

A better performance metric which provides equity to all voters was proposed in [4].

The proposed metric is the average absolute differences of expected waiting times among precincts which is defined in equation (1):

Where N is defined as the total number of voting precincts,, is the number of voting machines allocated to precinct *i*, and is the expected waiting time for voters at precinct *i*.

To find the allocation that provides the best equity we need to solve the following optimization problem:

Where *l* is the set of feasible solutions, and | *l* | is finite.

## Confidence Interval

A confidence interval (CI) is: “a particular kind of interval estimate of a population parameter and is used to indicate the reliability of an estimate.”

In our case we need to calculate the equity performance parameter and we get a sample of its value from different replications. This sample has unknown mean µ and unknown standard deviation, so the confidence interval for the population mean, based on sample of size n is defined in equation (2):

Where is the sample mean and t is the critical value for the *t* distribution with *n-1* degrees of freedom and s is the estimated standard deviation known as standard error.

We implemented the calculation of confidence interval for the results of the equity metric in Microsoft Excel.

## Experimental Design

In order to examine the performance of the simulation-based RA and compare it with the greedy improvement algorithm (GIA), we set up experimental design using four factors: Voting Time, Number of Precincts, Ratio of the number of machines to the number of precincts, and Allocation Strategy (see Table 2). The result of this experimental design is the “equity” metric given in (1).

Table 2: Factors and Values for Experimental Design

|  |  |
| --- | --- |
| Factors | Possible Values |
| Number of Precincts | 20 – 30 - 50 Precincts |
| Voting Time  (Scale Parameter of Gamma Distribution) | 0.58 - 1.05 |
| #Machines/#Precincts | 2 - 3.6 |
| Allocation Strategy | RA - GIA |

To obtain the allocations for each combination of design factors, we run the RA and GIA using the React.Net Library [10]. We use 50 replications for each scenario so that the 95% confidence-interval half width will be less than 10% of the average waiting time in a precinct.

There are 12 design points in total (see Table 3). We run 50 replications for each design point. Different random numbers are used for different design points to ensure that the design points are independent.

Table 3: Design Points

|  |  |  |  |
| --- | --- | --- | --- |
| **Design Point** | **Voting Time** | **No. of Precincts** | **No. of Machines** |
| 1 | 0.583 | 20 | 40 |
| 2 | 1.05 | 20 | 40 |
| 3 | 0.583 | 30 | 60 |
| 4 | 1.05 | 30 | 60 |
| 5 | 0.583 | 50 | 100 |
| 6 | 1.05 | 50 | 100 |
| 7 | 0.583 | 20 | 72 |
| 8 | 1.05 | 20 | 72 |
| 9 | 0.583 | 30 | 108 |
| 10 | 1.05 | 30 | 108 |
| 11 | 0.583 | 50 | 180 |
| 12 | 1.05 | 50 | 180 |

## Results

The running of simulation for the RA and GIA methods shows that the GIA method takes very long time to run all iterations compared to RA, this is because Phase 1 of the GIA runs the simulation number of times equivalent to the number of DRE machines to be allocated in one replication, and the RA method allocates them random at Phase 1 of it. For example if we have 100 DRE Machines and 50 replications, the simulation will run for about 100\*50 times in Phase 1 only which is a lot of time. So the RA outperforms the GIA in the speed of simulation.

The results of the experimental design are shown inside Table 4 which contains the equity and confidence interval (CI) of the RA vs. GIA method through 50 replications of the simulation.

From the results we can say that RA method is significantly better than GIA at large numbers of DRE Machines which is 6 out of 12 combinations of design points, and in small numbers of DRE machines the GIA is slightly better than RA in 6 out of 12 combinations of design points and at the best result the equity is better with about 5 minutes less than RA equity result (First Design Point).

Table 4: Results of Experimental Design

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DP** | **RA- Equity** | **RA - CI** | **GIA -Equity** | **GIA - CI** |
| 1 | 34.949 | 27.88 to 42.02 | 30.027 | 22.25 to 37.80 |
| 2 | 67.874 | 56.08 to 79.66 | 65.738 | 53.43 to 78.05 |
| 3 | 27.857 | 19.06 to 36.66 | 26.675 | 19.07 to 34.27 |
| 4 | 55.567 | 46.26 to 64.88 | 65.880 | 52.18 to 79.58 |
| 5 | 31.651 | 19.86 to 43.44 | 29.149 | 22.17 to 36.13 |
| 6 | 28.626 | 16.49 to 40.77 | 69.653 | 57.90 to 81.41 |
| 7 | 13.354 | 7.29 to 19.41 | 12.0472 | 8.56 to 15.54 |
| 8 | 32.961 | 21.26 to 44.66 | 36.377 | 27.52 to 45.24 |
| 9 | 14.867 | 7.19 to 22.55 | 16.031 | 9.74 to 22.32 |
| 10 | 21.002 | 10.98 to 31.02 | 45.641 | 35.45 to 55.83 |
| 11 | 6.689 | 0.00 to 13.47 | 23.236 | 16.31 to 30.17 |
| 12 | 9.106 | 2.57 to 15.65 | 41.936 | 32.85 to 51.03 |

Figure 1 and Figure 2 displays 95% confidence intervals and the range of the equity values against the allocations set by the RA and the GIA.

Figure 1 show the scenario for design point 10, where RA is better than GIA and Figure 2 show the scenario for design point 1, where GIA is better than RA, but the difference between the confidence interval is very small.

Figure 1: GIA vs. RA Allocation Strategies Confidence Interval of Design Point 10

Figure 2: GIA vs. RA Allocation Strategies Confidence Interval of Design Point 1

# Related Work

There are a few papers that deal with the voting machine allocation problem in order to avoid long lines for voters and provide equity.

The first one that we developed our research using it is [4], where they proposed a simulation-based Greedy Improvement Algorithm (GIA) to generate machine allocations to provide equitable voting experiences to all voters so that no one particular group of voters is disadvantaged or disenfranchised, they used the average absolute differences of waiting times across all precincts as a performance metric for equity, so did we.

One of the problems in simulating the voting system is that voters do not arrive according to a stationary arrival process, so they used data set based on statistics from the 2004 election in Franklin County, Ohio and they fit a normal distribution and generated a turnout rate, they also found that the election day is divided into periods and assumed that each time period the number of arriving voters follows a Poisson distribution; as for the voting time a gamma distribution fits the data acceptably.

Then there are two papers that used simulation to their models Edelstein (2006) and Allen, Bernshteyn (2006b), and this allow them to consider some realistic complications in the model like voting-machine failures and uncoordinated voter arrivals.

Edelstein (2006) used a simple method of allocating voting is to allocate machines in proportion to the expected number of voters at each precinct.

# Conclusions

The occurrence of long lines in elections depends on many unpredictable factors and is difficult to control. One way to explore this problem is through simulation which can be useful in the election process and can result in good recommendations to the allocation of DRE machines across precincts.

In our simulation model we used non-stationary voter arrivals, transient queues, and different turnout rates across all precincts.

The machine allocation RA method is proposed and illustrated using an example from the 2004 election in Franklin County, Ohio. Then it is compared to the GIA method [4].

The RA method is shown to offer potential advantages, both through the reduction of equity metric (the average absolute differences of waiting times across all precincts) and it takes smaller time in simulation compared to the GIA.

The GIA is better slightly in the case of small number of DRE machines, but the RA is significantly better than GIA in large numbers of DRE machines, which indicate that the number of machines affects the performance of machine allocation policies.

# Limitations and Future Work

An incomplete list of additional topics for future study is as follows:

1. Include more realistic cases such as heterogeneous precincts to the simulation model, which have different voters-arrival patterns, and different distributions of voting times due to ballot length and nature of voter. Also considering the voting-machine failures in the model which occurs in real life.
2. Explore the elections in the developing countries such as Egypt, where the elections don’t have a technological infrastructure for voting because it is not practical or present. It uses manual technique for voting, tallying, or verification of paper-ballots. These factors lead to the lack of necessary data which can be used to simulate elections in developing countries. Studying the elections there, find a way to simulate it, and applying the methods mentioned in the literature could help in strengthening the democratic process in developing countries.
3. Commercial software could be developed based on the RA or any other method on the literature. This software will help governments take decisions about election allocations that result in voters waiting the least amount of time possible and provide savings to the money spent on election process.
4. Compare proposed method to other methods in the literature such as Allocating machines to minimize the maximum expected voter waiting time across all precincts.

Acknowledgment

References

1. Allen, T. T., and M. B. Bernshteyn. 2006a. Mitigating voter waiting times. Chance 19 (4): 25–36.
2. Allen, T. T., and M. B. Bernshteyn. 2006b. optimal voting machine allocation analysis. In Analysis of May 2006 Primary Cuyahoga County, Ohio, ed. S. Hertzberg, 71–89. Election Science Institute. Available via <http://moritzlaw.osu.edu/electionlaw/news/documents/DREAnalysisforMay2006Primary.pdf>
3. Anthony, A. A., M. F. Colley, K. E. Marinello, and C. C. Petree. 2004. Election 2004: A report to the community. Franklin County Board of Elections Report. Available via <http://centralohiovoters.home.att.net/elect2004.htm>
4. Muer Yang, Michael J. Fry, W. David Kelton. ARE ALL VOTING QUEUES CREATED EQUAL? in Proceedings of the 2009 Winter Simulation Conference
5. Edelstein, W. A. 2006. New voting systems for New York - long lines and high cost. Technical report, New Yorkers for Verified Voting Report. Available via [http://www.nyvv.org/ doc/voterlines.pdf](http://www.nyvv.org/%20doc/voterlines.pdf)
6. Feldman, D., and C. Belcher. 2005. Voting experience survey. In Democracy at Risk: The 2004 Election in Ohio. Democratic National Committee. Available via <http://www.democrats.org/pdfs/ohvrireport/section03.pdf>
7. Houdt, B. V., and C. Blondia. 2005. Approximated transient queue length and waiting time distributions via steady state analysis. Stochastic Models 21:725–744.
8. Confidence Interval, Wikipedia Entry: <http://en.wikipedia.org/wiki/Confidence_interval>
9. Egyptian Constitutional Referendum, 2011, Wikipedia Entry: <http://en.wikipedia.org/wiki/Egyptian_constitutional_referendum,_2011>
10. React.NET Discrete Event Simulation Framework, Avilable via: <http://reactnet.sourceforge.net/>

# Work Division

|  |  |
| --- | --- |
| Team Member | Activites |
| Enas Mohamed,  em\_cmp\_eng@yahoo.com | 1. Search for ideas and conferences 2. Draft of Project Proposal (State of the art, and References) 3. Search for Simulation Library 4. Understand the React.Net Library used. 5. Building the main components of the simulation model. 6. Implementation of the RA method. 7. Run different iterations of the simulation and collect output 8. Write the Final Paper (Related Work, Suggested Solution) 9. Write Poster Presentation |
| Hesham Naiem Mamoun,  Hesham.naiem@yahoo.com | 1. Search for ideas and conferences 2. Draft of Project Proposal (Target Conference, Motivation, and Potential Contribution) . 3. Search for Simulation Library 4. Understand the React.Net Library 5. Search for Data to be used in simulation. 6. Implementation of the GIA method. 7. Run different iterations of the simulation and collect output 8. Write the Final Paper (Performance Measurement and Analysis, Conclusions, Limitations and Future Work) 9. Write Poster Presentation |
| Mostafa Mohamed Izz,  Mostafa.3ez@gmail.com | 1. Search for ideas and conferences 2. Draft of Project Proposal (Problem Statement, and Proposed work and plan) 3. Study Arena (Simulation modeling software) 4. Understand the React.Net Library 5. Responsible for using svn through google code. 6. Responsible for creating data using distributions by the library. 7. Run different iterations of the simulation and collect output 8. Preparing the document template of the conference. 9. Write the Final Paper (Abstract, Introduction,Motivation, Problem Definition) 10. Present Poster Presentation |

1. Manuscript received June XX, 20XX; revised July XX, 20XX. (Write the date on which you submitted your paper for review.) This work was supported in part by the U.S. Depart­ment of Com­merce under Grant BS123456 (sponsor and financial support acknowledgment goes here). Paper titles should be written in uppercase and lowercase letters, not all uppercase. Avoid writing long formulas with subscripts in the title; short formulas that identify the elements are fine (e.g., "Nd–Fe–B"). Do not write “(Invited)” in the title. Full names of authors are preferred in the author field, but are not required. Put a space between authors’ initials. [↑](#footnote-ref-2)