Testing the eSTV program for the Scottish Local Government Elections

Pieter Koopman Engelbert Hubbers Wolter Pieters Erik Poll
Rene de Vries
LaQuSO, Radboud University Nijmegen, The Netherlands
pieter@laquso.com
www.laquso.com

March 30, 2007

.

Management Summary

This report deals with the testing of eSTV software from ERS, Electoral Reform Services. The purpose of the test is to determine the conformance of eSTV to the WIG-rule [1]. In order to determine conformance we have created a new implementation of the rule in the functional programming language Clean [4]. Since this language uses a different programming paradigm and the implementation is made completely independent from the software to be tested it is extremely unlikely that both implementations contain the same error.

Testing is done by computing the election result for a given set of votes with both programs. The results of both programs are compared to the level of votes numbers after each selection and elimination step in the algorithm. This has shown to be a very sensitive test for very small variations in the algorithm used in the programs.

These tests are executed for 808 standard test cases. These test cases are partly handcrafted or the vote distributions in real elections for similar voting rules. Each of these sets is computed with 4 different elimination orders. In addition the results of 5000 generated data sets and a number of hand crafted data sets covering border situations in the algorithm are compared. Both test sets have shown to be very effective to identify even the smallest changes in the algorithm to compute the result of the election.

Testing of preliminary versions of eSTV showed some small inaccuracies in the program as well as some small space for different interpretation of the rules. The Scottish Executive provided the final interpretation of the rules (as detailed in section 7.1) and all test where repeated for eSTV version 2.0.16. In these final tests there was an exact match for each of the $4 \times 808 + 5000 = 8232$ test cases.

Contents

Chapter 1

Introduction

In the fall of 2006 the Radboud University Nijmegen was invited to tender for the contract to provide verification services for the STV calculation software by the Scottish Executive, the devolved government in Scotland. For the elections in May 2007 they will be introducing both e-counting and *Single Transferable Vote*, STV, for the local government elections for the first time. The eSTV program which will be used calculate the results of the Scottish Local Government elections in May 2007, works in accordance with the rules defined in the Local Government Election Order, which were approved by the Scottish Parliament on 25 January 2007. Whilst they have confidence in this software, they are conscious that for the sake of certainty and transparency there will be a need for an independent body to verify that the calculation they are asking the software to perform, in line with a set of defined election rules, is carried out accurately. The most appropriate form of testing was black box comparison testing where a set of supplied data is run through a reference program and the results are compared with those generated by the eSTV software supplied by ERS, Electoral Reform Services. The level of data to be put through the system is anticipated to be approximately 800 sets of preferential voting data, which will be supplied in a simple text format.

This question was redirected to LaQuSo, the laboratory for Quality Software, an initiative of Eindhoven University of Technology and Radboud University Nijmegen. See `www.laquso.com`.

LaQuSo and the Scottish Executive agreed that the LaQuSo department at the Radboud University Nijmegen will write software to test execute the STV algorithm according to the rules for the elections in may 2007, and run 800 standard test cases to this program and eSTV and compare the results.

Chapter 2

Specification

STV is a preferential voting system designed to minimize wasted votes and to provide proportional representation while votes are explicitly for candidates instead of party lists. When STV used in multi-seat constituencies as in this case, it is also called proportional representation through the single transferable vote (PR-STV). STV usually refers to PR-STV, and as it does here. In Australia STV is known as the Hare-Clark Proportional method, while in the United States it is sometimes called choice voting and preference voting.

The key idea is that each person gives a preference list of candidates as a vote. This list contains at least one candidate and at most all candidates in order of preference. In the first step of the election process the votes are given to the first candidate in the list. When a candidate is elected and has more votes than required, the surplus of votes is transferred proportionally to the next candidates on the ballots assigned to that candidate. Similarly, if a candidate is eliminated, their votes transfer to the next candidates on the ballots of that candidate.

The quota Q is computed as:

$$Q = \lfloor \frac{\text{number\_of\_votes}}{\text{number\_of\_seats} + 1} \rfloor + 1$$

The floor brackets $\lfloor x \rfloor$ indicate rounding down of the number x. The fraction of x, if any, is simply removed. This definition is called the *droop quota*, it ensures that at the number of candidates that reach the quota is at most equal to the number of seats.

The election rules, [1], give an operational description that guides a human to determine the result of the election by putting ballots on piles and transferring ballots to other piles. In more abstract terms this algorithm is:

```
assign ballots to the first candidate;
while (not all seats filled)
{
    declare any candidate with votes >= Q elected;
    if (number of candidates == number of seats)
        all candidates are elected;
    else if (there are candidates with more votes than Q)
        transfer the votes of the candidate with the most votes;
    else
        eliminate the candidate with the fewest votes and transfer the votes;
}
```

In the situation that there is more than one candidate with the highest number of votes, we look in the history. If one of the candidates had more votes on one of the previous iterations of the algorithm, the surplus of that candidate is transferred first. Otherwise there is a *tie*. In

a real election a human has to decide which candidate is treated first. During testing various fixed orders of elimination are used.

If there are several candidates with the least number of votes for elimination the same algorithm is used: look for a difference in the history, if that is not available use a tie.

Initially all votes have value one. In the transfer of votes they get a new value. The transfer value, tv is computed as

$$tv = \frac{(votes\_of\_candidate - Q) \times current\_value}{votes\_of\_candidate}$$

After the computation the tv is truncated to five decimal places.

The votes are transferred per ballot pile to the next candidate on that ballot that is neither elected nor eliminated. If there is not such a candidate available, the votes are added to the nontransferable votes.

An obvious condition to be satisfied is that no votes are lost in this algorithm. The sum of the votes of the candidates and the nontransferable votes should be equal to the initial number of valid votes after each iteration of the algorithm.

Chapter 3

Subject and goal of the tests

The goal of the tests is the verification of the eSTV software. This implies that we will run eSTV for a (large) number of test cases and determine whether it is a correct implementation of the algorithm specified by the WIG-rules [1], and outlined in chapter 2.

eSTV is a program to facilitate the counting of an STV election. Joe Otten, `joe@datator.co.uk`, has the copyright of this program. It is a product of ERS, Electoral Reform Services.

eSTV is able to handle various variants of STV elections. In the GUI-based interface the variant to be used can be determined by radio buttons. In the command line version, the variant to be used is determined by a command line argument of the program. In addition to eSTV there is a program called BulkTest. This is used to apply eSTV to a large number of election data. Each set of votes is stored in a separate file. The result of the election algorithm, intermediate numbers of votes for the candidates after each iteration of the algorithm, as well as the elected candidates are stored in a separate file for each set of votes.

We only have access to an executable version of eSTV, not to the Delphi source code. This implies that the verification will be done by a black box test. We supply vote distributions to eSTV and verify the result of the election as produced by eSTV.

The program eSTV tells the user that it is limited to 3000 candidates and 100000 ballot papers per contest. The Scottish Executive indicated that this is sufficient for the Scottish elections in May 2007 where the electorate in any one local government contest will not exceed 26,000. Testing showed that eSTV produces overflow errors for number much larger than these bounds. With numbers on or below these bounds we were not able to generate overflow errors. During the execution of tests described in chapter 6 overflow errors did not occur.

During the test described in this report various versions of eSTV are used. New versions were created based on autonomous development at ERS, issues found during the tests and our requests to allow better tests. The final version of eSTV used in the test is 2.0.16. The final verification verdict is based on this version.

Chapter 4

Test plan

The most convenient way to test a piece of software with a model based test tool like Gast [2] is to have a relation between input and output of a program that is easy to check. The construction of the output might be complicated, but checking the output is easy. A very simple example is a function that computes the square root of its input, multiplying the output by itself should yield the input for all positive numbers. A somewhat more sophisticated example is the construction of magic squares, a tiny example is depicted in the margin. It is not easy to generate large squares of numbers such that the sum of all rows and columns is equal, but it is very easy to check if a generated square is indeed a magic square (just compute the sum of all rows and columns and compare these sums).

8	3	4
1	5	9
6	7	2

Unfortunately there does not exist a property to check if the result of an election is correctly determined based on the vote distribution and the election result. Hence we decided to build our own implementation of the election algorithm and verify the correctness of eSTV by comparing the results to our own implementation. In order to reduce the risk that both programs contain the same error, our implementation was developed completely independently of eSTV. Our implementation is based only on the provided election rules and some additional information provided by, or on behalf of, the Scottish Executive. To further reduce the occurrence of identical errors in both programs, or the compiler used to construct the executable, we used the high level functional programming language Clean to construct our program.

After computing an election result eSTV produces not only the direct election result, in which candidates are elected, but also a table giving the number of votes for all candidates after each iteration of the algorithm, as well as a log of the vote transfer. This table is produced as a comma separated file. This csv-format is commonly used as text format for spreadsheets and similar programs. We decided to compare the textual representation of this table in the csv-format to verify the equality of the results of both programs. Comparing the tables will show any slight difference in number of votes during the various iterations of the algorithm, while the decision to elect or eliminate candidates based on different vote distributions can still be equal. Hence, comparing the tables is a much more sensitive for way of testing than just checking whether the same candidates are declared elected by both programs.

Apart for the votes in a text file in dat-format [3], eSTV can take user input to break ties. For testing large number of test cases automatically such a user input is impractical. Fortunately eSTV is able to break ties automatically, either in a fixed order or pseudo randomly. Our first attempt was to let our own software generate all possible election results and check whether the result generated by eSTV is part of this collection of results. This approach has the danger that the set of election results to be generated is impractically large. In real elections ties are rare, and even after four ties between 2 persons there are only $2^4 = 16$ solutions. In the standard test cases however, there are several examples where the number of ties is large. Test case T100.dat

for example contains a tie between 23 persons with the same number of votes. Even if only two of these persons have to be eliminated, this yields 506 different elimination orders. If all of these candidates have to be eliminated this yields 23! = 25852016738884976640000 different elimination orders. Indeed an impractical large value. The generation of possible solutions was interrupted after the generation of more than 9 GByte of transfer tables.

On our request eSTV and BulkTest were changed such that four different systematic orders can be used: for vote transfer and elimination both from front to back and from back to front. All standard test cases were run in these four directions. For a small number of test cases also other elimination orders were tested semi automatically.

Chapter 5

Tooling

In order to handle the numbers with five digit precision used by the algorithm easily we constructed a special data type `Fixed` for these numbers. A number in this data type is just the number multiplied by 10^5. This yields a whole (integer) number that is stored as a `BigInt`, an infinite size integer, to prevent overflow problems. For this type conversion from and to strings is defined as well a set of arithmetic operators (like comparison, addition, subtraction and multiplication). These operators are tested with Gast in the usual way. It is no surprise that elementary arithmetic laws do not hold for these finite precision numbers. For example $(a \times b) \times c \neq a \times (b \times c)$ for $a = 0.33333$, $b = 0.50000$ and $c = 2.00000$. This just tells us that we have to be careful with the order of mathematical operations in the formulation of the transfer factor. Another way to limit the effects of using finite precision numbers is by delaying the rounding of values. A value is rounded only at to the end of the entire computation, instead of rounding the value of each sub-computation. This is used in the election algorithm by rounding the value of tv as defined in chapter 2, only at the end end of the computation of a new transfer value. This rounded value of tv is then multiplied by the current value of the pile of ballots and rounded again.

The further implementation of the election algorithm is rather straightforward. We use lists of candidates, ballot piles and votes on a ballot to prevent any danger of problems with bounded arrays. For very large numbers of votes or candidates this might be less efficient, but the possibilities to make errors using lists are much smaller than with arrays. Moreover, eSTV is limited to relatively small numbers. It appears that our implementation using lists is significantly faster than eSTV using arrays.

Chapter 6

Test sets

As test case we used the set of 808 standard test cases provided along with eSTV, as well as a set of 5000 test cases generated using the Gast technology and some hand crafted test cases.

6.1 Format of the test cases

Each test case is stored in a separate text file. The names of these files end in `.dat`. On the first line such a file contains the number of candidates and the number of seats. Then there is optionally a line indicating which candidates are withdrawn, indicated by a sequence of negative numbers. Then there is a series of lines indicating the values on the ballot papers. This sequence is terminated by a line beginning with 0. Each line starts with the number of ballot papers with this vote distribution, followed by the numbers of the candidates and terminated with 0.

After the votes, there are some lines containing the names of the candidates between quotes and the name of the election. Optionally this data is followed by some comments. A typical example is:

```
5 3
-3
1102 1 2 4 0
1101 2 5 4 0
398 4 0
399 3 5 4 0
0
"Alice"
"Bob"
"Clay"
"Desiree"
"Ed"
"Fake election"
```

Since Clay is withdrawn, the pile of 399 votes is immediately assigned to Ed. Since there are 3000 votes (all valid) and 3 seats, the quota Q equals $\frac{3000}{3+1} + 1 = 751$. This implies that Alice and Bob are elected immediately. Since Alice has more votes than Bob her votes are transferred first. Since Bob is already elected, he does not receive votes from Alice. In the next iteration the votes of Bob are transferred. Next Desiree or Ed has to be eliminated. In the current round they have an equal amount of votes, but in the previous round Desiree had more votes than Ed. So, Ed is eliminated. His 399 votes are transferred to Desiree. Now the number of remaining candidates is equal to the number of remaining seats: 1. The remaining candidate, Desiree, is now deemed to be elected. Note that Desiree has also exceeded the quota. This ends the

election algorithm. Due to rounding all numbers to five digits there are some rounding errors. This explains the small number of non-transferable votes.

The lines of the corresponding .csv file that contain the transition table and are compared in the tests are [1]:

```
"Alice",1102,-351.00000,751.00000, ,751.00000, ,751.00000,"Elected"
"Bob",1101, ,1101.00000,-350.00000,751.00000, ,751.00000,"Elected"
"Clay","Withdrawn", , , , , , ,
"Desiree",398,+350.99802,748.99802, ,748.99802,+748.99689,1497.99491,"Elected"
"Ed",399, ,399.00000,+349.99689,748.99689,-748.99689,"-",
"Non-transferable", ,+0.00198,0.00198,+0.00311,0.00509, ,0.00509,
"Totals",3000,,3000.00000,,3000.00000,,3000.00000
```

In a spreadsheet, like MS Excel or Calc from Openoffice, this wil be displayed like:

Alice	1102	-351.00000	751.00000		751.00000		751.00000	Elected
Bob	1101		1101.00000	-350.00000	751.00000		751.00000	Elected
Clay	Withdrawn							
Desiree	398	+350.99802	748.99802		748.99802	+748.99689	1497.99491	Elected
Ed	399		399.00000	+349.99689	748.99689	-748.99689	-	
Non-transferable		+0.00198	0.00198	+0.00311	0.00509		0.00509	
Totals	3000		3000.00000		3000.00000		3000.00000	

Due to the automated test execution, displaying these tables in a spreadsheet and human inspection is only needed to find the cause of issues spotted by the test software.

The election rules for the Scottish local government elections state that candidates cannot withdraw after the close of nominations. However, for the purposes of testing, the Executive indicated that withdrawn candidates should be treated as eliminated candidates. This is relevant since some of the provided test cases contain withdrawn candidates. For this reason we include a withdrawn candidate in the example.

The closest match of the example above for the Scottish elections is the situation where Clay does not get any votes at all:

```
5 3
1102 1 2 4 0
1101 2 5 4 0
398 4 0
399 5 4 0
0
"Alice"
"Bob"
"Clay"
"Desiree"
"Ed"
"Fake election 2"
```

The corresponding transfer table is:

Alice	1102	-351.00000	751.00000		751.00000	751.00000		751.00000	Elected
Bob	1101		1101.00000	-350.00000	751.00000	751.00000		751.00000	Elected
Clay	0		0.00000		0.00000	-		-	
Desiree	398	+350.99802	748.99802		748.99802	748.99802	+748.99689	1497.99491	Elected
Ed	399		399.00000	+349.99689	748.99689	748.99689	-748.99689	-	
Non-transferable		+0.00198	0.00198	+0.00311	0.00509	0.00509		0.00509	
Totals	3000		3000.00000		3000.00000	3000.00000		3000.00000	

[1] Note that for a table with an equal number of columns for all rows, the last line is missing one comma (and hence one column). Any spreadsheet will add this additional column without complaining.

In addition some empty fields are really empty (in the row totals) while others contain a space. This is not an issue for a spreadsheet, but does matter for the textual comparison used in the tests.

Note that Clay is now eliminated after the transfer of the surplus of Alice and Bob. Since Clay has less votes than Desiree and Ed, he is eliminated first. There is an additional empty column indicating the transfer of zero votes and a column indicating the new (unchanged) totals for all candidates but Clay. In this column (and all columns to the right of it) the elimination of Clay is indicated with a $-$, instead of zero votes.

6.2 Manual test cases

A small set of hand crafted test cases was used initially to test our own software. Later it was also used to compare the results of eSTV and our software.

These test cases cover the basic behavior of the algorithm including special situations. The special situations tested are:

1. No transfer of votes needed, the desired number of seats is filled directly.
2. Only one seat to be filled.
3. Numbers of seats one less than the number of candidates.
4. Number of seats equal to the number of candidates, no election needed.
5. Right elimination order.
6. Right transfer order.
7. Right transfer order when a candidate that is elected later has more votes to transfer than a candidate elected earlier.
8. Filling last vacancies by the special clause in the algorithm.
9. Very large number of votes. Not exceeding the bounds mentioned above.
10. Very large number of candidates. Not exceeding the bounds mentioned above.
11. Many withdrawals, such that there are fewer candidates than seats. As noted above this can happen in the format for data provided, but not in the real elections.
12. Votes for nonexistent candidates. Any ballot containing votes for a nonexistent candidate should be classified as invalid.
13. Same candidate multiple times on one ballot. All these ballots should become invalid.
14. Withdraw one candidate twice. This is harmless and should be allowed.
15. Votes for withdrawn candidates. Since withdrawn candidates have to be treated as eliminated, this is allowed.
16. Ties for transfer.
17. Ties for elimination.

Most likely these cases are also included in the 808 provided test cases. Since there was no documentation on these test cases, it was easier and faster to design these test cases than to find them in the provided test suite.

6.3 The standard test cases

The exact source of the standard test cases is not known to us. By looking at the test data and the comments it is clear that there are manually generated test in order to cover special cases in the election algorithm as well as data from real elections using some variant of STV.

6.4 Generated test cases

The generated test sets are constructed using the abilities of Gast to generate instances of data types systematically. In principle all we have to do is to define a type corresponding directly to the data format outlined above, let Gast generate instances of this type, and write each instance in a separate file.

In this situation we have added some guidance in order to guarantee that special cases in the algorithm (like ties and rounding of numbers) occur frequently. For testing the final version of eSTV we generated a set of 5000 test cases.

6.5 Test execution

Generating the result of a test set of this size takes a considerable amount of time: more than one hour for one of the four elimination options. Generating the reference results by our software and comparing the results is about one order of magnitude faster. In the provided test set there are 162 cases that contain a tie. The generated test suite contains 1849 cases with a tie.

Both test sets appear to be effective. If we make a change in the program that influences the result in some situation, this situation occurs in both test sets. As usual with black box testing, this holds only for general changes in the program. If we make a very specific case in the program, for instance a candidate with exactly 1234.56789 votes (or a specific name) gets always precedence in a tie, it is unlikely that this is discovered by any kind of black box testing.

Chapter 7

Issues found

During program construction and testing a number of issues were found. Here we group them by source.

7.1 Specification

The rules we had did not cover all aspects of the election. The issues are:

1. Withdrawal of candidates is covered by the data format and occurs in the test cases, but is not mentioned in the rules. The election rules state that candidates cannot withdraw after the close of nominations. However, for the purposes of testing the standard test suite, the Executive indicated that withdrawn candidates should be treated as eliminated candidates.

2. The rules do not specify how to treat blank ballots. In some election systems a blank vote is valid. In this system they are invalid. This is done since blank votes have an influence on the quota Q if they were treated as valid.

3. Also other kind of invalid ballots are not covered by the provided rules. A ballot containing a nonexisting candidate is invalid. The entire ballot, not only the invalid candidate, is ignored by the election algorithm. The Scottish Executive has advised that this occurrence would not be an issue in the May elections as any non-existing candidates on the ballot paper would simply be ignored.

4. Also forms containing a candidate twice are considered to be invalid and are ignored in the election process. However, such a form will be harmless in the election algorithm. When the second occurrence of the candidate is considered by the algorithm the candidate has either been elected or eliminated. In both situations the candidate number will be ignored by the algorithm.

5. During the tests we discovered that the rounding to five digits as prescribed by rule 48 can be interpreted in two different ways. One way to read this is that all calculations should be done with numbers containing (at most) five digits. Another way of reading this tell to do the calculations in arbitrary precision and round to five digits after all computations. The Scottish Executive confirmed that the latter interpretation was more in line with the intention of the local Government Election Rules.

7.2 Transfer table

The specification [5] does not prescribe exactly how to make a transfer table. A lot of different choices appear to be possible. In all but one of these situations we have adopted our transfer table to the table generated by eSTV.

In the first versions more than 75% of the generated transfer tables indicated an issue. Fortunately a small number of differences in layout of the tables were responsible for the majority of these issues.

7.3 Data format of test suite

The provided data format gives the name of each candidate on a separate line. There are a few test cases that contain more than one name on a line. We have adapted our software such that these case are handled correctly

7.4 Real differences

One real problem was found in an earlier β-version of eSTV. For a fully populated ballot paper (e.g. in F002, the voter marked a full 9 preferences), the final preference is ignored in eSTV. In most other STV rules this makes no difference - a nine-candidate ballot paper marked `1 2 3 4 5 6 7 8 9` is logically equivalent to `1 2 3 4 5 6 7 8`. The lack of formal exclusions in Scottish rules is obviously an exception. This was fixed in version 2.0.12 and still correct in the final version 2.0.16.

A minor difference between our software and eSTV is the treatment of ballots containing only votes for withdrawn candidates, like `100 3 0` in the example in section 6.1 above. eSTV filters the ballots by removing withdrawn candidates before considering the ballots in the remaining algorithm. This implies that these ballots become empty and hence invalid. Our software considers this as a valid ballot since it is nonempty and contains only valid candidate numbers. Since the candidate is withdrawn, the votes are added to the non-transferable votes. These different choices result in a different number of valid votes, and hence a different quota. This causes another transfer table. In effect eSTV treats withdrawn candidates and their votes as nonexistent, while our software treats them as dynamically eliminated. Since the Scottish Executive confirmed that a candidate cannot withdraw during the voting period in this election, we have not elaborated on this.

Other really different transfer tables occurred in the tests. It appeared that they were cause by a different interpretation of the rounding to five decimals. After agreeing on the rule that the rounding should only be done at the end of the calculation (and implementing that rule), all these differences disappeared.

7.5 Windows platform

eSTV does not run on all Windows XP distributions. Especially on Dutch systems there appears to be a problem with kernel32.dll which prevents eSTV from starting properly. This problem is still existent, but not considered as problem with the correctness of eSTV with respect to the election rules.

Chapter 8

Conclusion

Testing eSTV was a useful activity. A number of issues were found and solved.

Testing revealed a spot in the specification, the Scottish local government election rules, where two different interpretations are possible. These different interpretations of the rounding rule give different transfer tables, and hence there exists vote distributions with a different result for both interpretations of the rules.

The subject of test, eSTV, produced a strange transfer table for some eliminated candidates in some situations. The elimination was correct, but shown in an unexpected way.

Preliminary versions of eSTV ignored the final candidate on a fully populated ballot. This was corrected.

Testing with the standard test suite for this kind of elections and the generated test suite indicated the same issues after roughly the same number of tests. This indicates that both test suites are equally effective.

No issues were found in the final version of eSTV. Since testing has been shown to be very sensitive to small differences in the programs, it is very likely that they will produce the same results for all inputs. Since both programs have been created independently and in a different programming paradigm, it is very unlikely that they contain the same error. Hence, one can have great confidence in the correctness of eSTV version 2.0.16 in calculating election results in complete conformance with the rules defined in the Local Government Election Order approved by the Scottish Parliament on 25 January 2007.

Bibliography

[1] The Scottish Ministers, Scottish Local Government Elections Order 2007, Rule 45–52, December 2006.

[2] Koopman, P. and Alimarine, A. and Tretmans, J. and Plasmeijer, R., Gast: Generic Automated Software Testing, In Peña, Ricardo and Arts, Thomas: The 14th International Workshop on the Implementation of Functional Languages, IFL'02, Selected Papers, Springer LNCS 2670, pp. 84–100, 2003.

[3] Document `votes_dat format.txt` as mailed on 25 Oct 2006 by the Scottish Executive.

[4] Rinus Plasmeijer and Marko van Eekelen, Concurrent CLEAN Language Report (version 2.0), Nijmegen Institute for Computing Science and Information Science, December 2001, `www.cs.ru.nl/~clean`

[5] Brian Wichmann, CSV format for Result Sheets, 16/1/99, version 4.