

2008

PACIFIC NW SOFTWARE
QUALITY
CONFERENCE



COLLABORATIVE
QUALITY

OCTOBER 13-15, 2008

CONFERENCE PAPER EXCERPT
FROM THE

CONFERENCE
PROCEEDINGS

Permission to copy, without fee, all or part of this material, except copyrighted material as noted, is granted provided that the copies are not made or distributed for commercial use.

Collaborative Techniques for the Determination of a Best Alternative in a Software Quality Environment

Dr. James McCaffrey
Volt Information Sciences, Inc. / Microsoft Corp.

presented at the Pacific Northwest Software Quality Conference
Portland, Oregon
October 13-15, 2008

== Introduction

Consider the general problem of determining one best option from a list of alternatives, where the decision making process is collaborative (performed by two or more people) rather than by a single person or by some purely quantitative technique. Examples of this type of activity in a software quality environment include a group of beta users choosing the best user interface from a set of prototypes, and the members of a company or team voting for some policy alternative. Situations where exactly one of a set of alternatives must be selected by a group of people occur in a wide range of problem domains, and different problem domains tend to use different terminology. In sociology, the study of this type of problem is usually called social choice theory, and typically uses terms like options and evaluators. In political science, analysis of these problems is often called voting theory and often uses the terms candidates and voters. In mathematics, group determination of a best alternative is frequently considered a sub-branch of decision theory. In a software development and testing environment, a mixture of terminologies is typically used.

Based on my experience, practical techniques for performing a group determination of a best alternative are not widely known in the software quality and development communities. Dozens of group analysis techniques have been studied. This paper presents five of the most common methods used for group determination of a best alternative in a software testing environment:

- The pure plurality technique
- The majority runoff technique
- The Borda count system
- The Condorcet method
- The Schulze method

Each technique is explained with an emphasis on their application to software quality. The techniques described in this paper apply to a fairly narrow range of problems and do not apply to strictly quantitative scenarios where a best alternative is clearly defined, non-collaborative scenarios where the decision is made by a single person, or scenarios where a group of people use discussion and negotiation to arrive at a consensus decision.

== The Pure Plurality Technique

The pure plurality technique is the simplest technique and one of the most common ways to perform a collaborative determination of a best alternative -- but one which is not the best approach in many software quality situations. Imagine you are developing a Web based software application for your external use and you create four significantly different user interface designs. Although you could ask a single person to choose the best design, in most situations a better approach is to ask a group of people to rank your four different prototypes. The process of performing a collaborative evaluation, and interpreting the results of the evaluation, are a bit trickier than you might expect. The pure plurality technique is simply to submit the alternatives to a group of evaluators and allow each evaluator to select, or vote for, just one alternative. The alternative that receives the most votes is declared the winner. Suppose for example, that you submit the four different prototypes (call them A, B, C, D) to 10 evaluators,

and ask each person to choose the best prototype. Now imagine the results are: Prototype A is best according to 4 people, prototype D is ranked best by 3 people, prototype C is chosen by 2 people, and prototype B is selected best by 1 person. Therefore, you select prototype A for further development. There are several problems with this approach. Suppose that while your ten evaluators are looking at the four prototypes, they are mentally ranking each alternative in this way:

A > B > C > D according to 4 people
D > C > B > A according to 3 people
C > B > D > A according to 2 people
B > C > D > A according to 1 person

Notice that even though prototype A is selected as best by the pure plurality rule, that prototype is evaluated as the worst design by 60% of your evaluation group. Furthermore, in this situation, suppose that prototype A was compared against each of the other prototypes in a head-to-head fashion. Prototype A would lose each head-to-head comparison by a 60% to 40% margin. Additionally, in some sense the non-winning opinions do not contribute directly to the final result.

== The Majority Runoff Technique

The problems with pure plurality led to the development of another group selection technique which is usually called the majority runoff system. There are many possible variations on the idea, but the winning alternative is required to receive more than 50% of the first place votes of all votes cast. This can be accomplished either by performing multiple rounds of voting, or by performing just a single round of voting but requiring the evaluators to rank all options from best to worst. For example, if evaluators vote for the best prototype design according to the data above, after the first round of voting, no prototype would have a majority. So, the list of candidates would be reduced (typically to the top two options.) In this case the top two alternatives are prototype A with 4 votes and prototype D with 3 votes. A second round of voting takes place. If evaluators vote according to their original ranking preferences above, after the second round of voting, prototype A would still receive 4 votes as best but prototype D would now receive 6 votes and be declared the winner. Using multiple rounds of voting in a software development and quality environment often has significant disadvantages compared to using a single round of voting. In some software development scenarios, multiple rounds of voting simply aren't feasible. Furthermore, because you are dealing with human beings, multiple rounds of voting can become tedious. Also, situations can arise where a group of people are voting for losing alternatives over and over until they are forced to choose an option they don't really like. Because voters are most often stakeholders in the sense that they will be affected by the outcome of the voting, you can potentially create disenfranchised voters at the end of a physical runoff scheme. So instead of performing multiple actual rounds of voting, you can just perform one round of voting but require evaluators to rank all candidates from best to worst. This allows you to determine voters' preferences in virtual rounds of voting. Requiring ranking of all alternatives has disadvantages too. The technique often isn't practical when the number of alternatives is large (typically more than eight) simply because humans have difficulty comparing large numbers of alternatives. Also, research has shown that evaluators tend to get careless with their evaluations of options that are low on their list of preferences. And, when performing multiple, physical rounds of voting, evaluators may not rank a subset of alternatives in the same order in which they rank the entire set of alternatives.

== The Borda Count System

The problems with the pure plurality and majority runoff techniques led to the development of an analysis technique called the Borda count system. The Borda count approach is very simple and one that you have almost certainly used before. Suppose there are k alternatives. Each evaluator ranks the alternatives from best to worst. A top-ranked alternative is assigned a value of k-1 points, a second-ranked alternative is assigned k-2 points, and so on, down to the last ranked alternative which receives 0 points. The Borda count is the sum of the point values for each alternative. For example, suppose there are four alternatives, A, B, C, and D, and seven evaluators. Therefore a first place ranking is worth 3 points, second place is worth 2, third place is worth 1, and last place is worth 0 points. If voting results are:

B > A > C > D according to 4 people
A > C > D > B according to 2 people
D > C > B > A according to 1 person

then the Borda count for each alternative is:

$$\begin{aligned}A &= (4 * 2) + (2 * 3) + (1 * 0) = 14 \\B &= (4 * 3) + (2 * 0) + (1 * 1) = 13 \\C &= (4 * 1) + (2 * 2) + (1 * 2) = 10 \\D &= (4 * 0) + (2 * 1) + (1 * 3) = 5\end{aligned}$$

and so alternative A is selected as the best option. The Borda count technique is widely used in many problem domains, especially in sports competitions. Advantages of the Borda count technique when used in a software development and testing environment are that the technique seems to make sense intuitively, and all evaluators' opinions are taken into account. However the Borda count system has several technical problems. Suppose that you decide to remove option D, which was a clear loser. Common sense suggests that removing an (apparently) irrelevant alternative should not affect the final result of voting. However the voting data becomes:

B > A > C according to 4 people
A > C > B according to 2 people
C > B > A according to 1 person

and so the Borda counts are now:

$$\begin{aligned}A &= (4 * 1) + (2 * 2) + (1 * 0) = 8 \\B &= (4 * 2) + (2 * 0) + (1 * 1) = 9 \\C &= (4 * 0) + (2 * 1) + (1 * 2) = 4\end{aligned}$$

and now option B becomes the best alternative instead of option A. In other words, with this particular set of data, removing an irrelevant alternative changes the outcome. It is even possible to construct Borda count examples where the best alternative actually becomes the worst alternative. This effect is sometimes called the Borda count winner-becomes-loser paradox. There are two other closely-related technical problems with the Borda count technique. The first is the pair-wise comparison problem. It is possible when using the Borda count technique to run into situations where the Borda winner would lose against a non-winner in a head-to-head comparison. For example, suppose there are four alternatives and 12 evaluators. If the ranking preferences are:

B > A > C > D according to 7 people
A > C > D > B according to 1 person
C > A > D > B according to 2 people
D > A > C > B according to 2 people

then using the Borda count method, alternative A is selected as best with 25 points. However if you examine head-to-head comparisons you will notice that alternative B is preferred to all other alternatives (B is preferred to A by a score of 7 people to 5, B is preferred to C also by 7 to 5, and B is preferred to D again by 7 to 5.) The other problem with the Borda count technique is that it is theoretically vulnerable to evaluator manipulation. In some situations it is possible to affect the outcome of a Borda count evaluation by adding a spurious alternative that is very close (in terms of evaluators' preferences) to an existing alternative. This can have the effect of diluting close alternatives and changing the winner of the analysis. The references at the end of this paper contain examples of this effect. Despite these technical flaws with the Borda count technique, the system is quite practical often works well for collaborative policy determination by small groups in a software testing environment. The Borda count technique is generally seen by voters as fair in a subjective way, and therefore if voters are stakeholders, the people who voted for non-winning alternatives generally accept the result of the voting and do not come away with a

psychological bias against the winning alternative. The point is that if you do use the Borda count technique, check your data for the effect of removing an irrelevant alternative, the result of pair-wise comparisons between the Borda count winner and non-winning alternatives, and mid-process addition of a new alternative.

=== The Condorcet Technique

The Condorcet technique for collaboratively determining the best alternative from a set of options, was developed primarily as a reaction to the head-to-head pairing problem of the Borda count method. The Condorcet method is very simple. It requires evaluators to rank all alternatives, and then a comparison of the results between each possible pair of alternatives is performed. If one alternative beats every other alternative in a head-to-head comparison then that one alternative is declared the Condorcet winner. If there is no Condorcet winner then a separate, tie-breaking technique is employed. The tie-breaking technique can be any other technique (such as Borda count or pure plurality.) The main principle of the Condorcet system is that a unanimous head-to-head winner should automatically win over any alternatives chosen by any other criteria. In practice, the Condorcet system is not often used by itself as a group evaluation technique. Instead, evaluation techniques such as pure plurality, Borda count, and others are mathematically analyzed to see if they satisfy the Condorcet principle in all cases.

== The Schulze Method

One of the most interesting voting systems which satisfies the Condorcet principle has a variety of names including the Schulze method, the clone-proof Schwartz sequential dropping technique, and the beat-path method. The Schulze method is somewhat more complicated than the other techniques presented in this paper. The Schulze method is similar to the Condorcet system in the sense that the Schulze algorithm checks to see if one option dominates all other options. However, instead of directly using raw ranking data like the Condorcet system does, the Schulze method performs a head-to-head comparison using an indirect measure of the strength between alternatives. This indirect measure is called the Schulze path strength. The technique is best explained by example. Suppose you have four alternatives, A, B, C, and D, and 11 evaluators. And suppose the voting data is:

A > B > C > D according to 4 evaluators
B > C > D > A according to 2 evaluators
C > D > A > B according to 3 evaluators
D > C > B > A according to 1 evaluator
C > A > B > D according to 1 evaluator

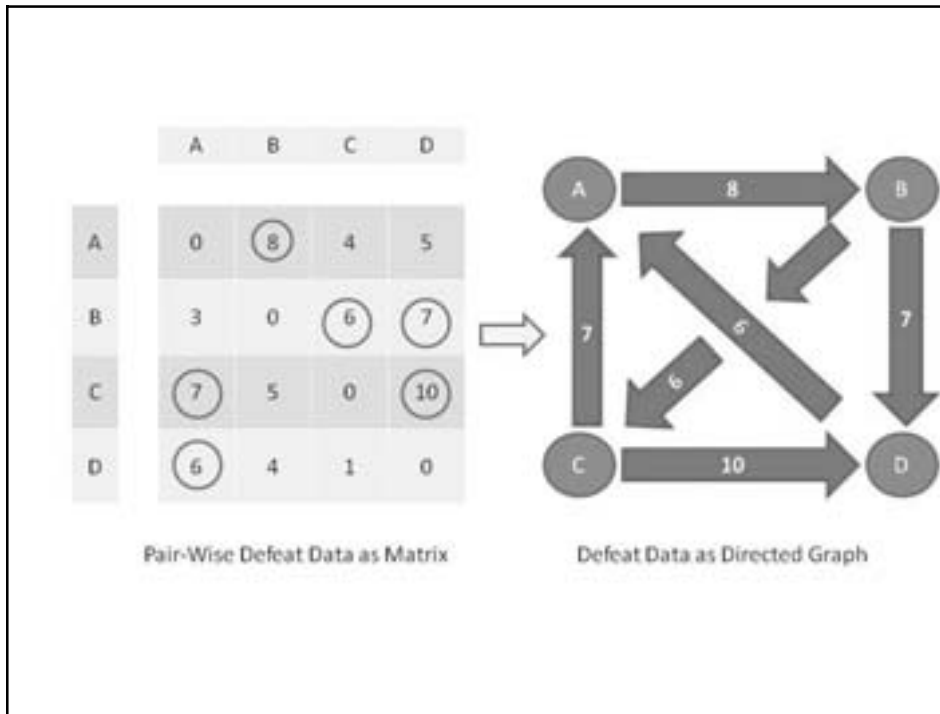
The Schulze method allows evaluators to rank options equal to each other, and also allows evaluators to leave out alternatives altogether. The first step in the Schulze method is to construct a matrix of pair-wise defeats. For the raw voting data above, the matrix of pair-wise defeats is:

```
0 8 4 5
3 0 6 7
7 5 0 10
6 4 1 0
```

The first row of the matrix means option A is preferred by 8 voters over option B, preferred by 4 voters over option C, and by 5 voters over option D. The second row means option B is preferred by 3 voters over option A, by 6 voters over option C, and 7 voters over option D. The third and fourth rows, for options C and D, are interpreted similarly. The pair-wise defeats matrix has 0 values on the main diagonal because these values are comparisons between an option and itself. At this point the Condorcet system would check this direct pair-wise data to see if there is a unanimous head-to-head winner but the Schulze method derives an indirect measure of strength between alternatives called the path strengths. For this example the path strengths are:

0 8 6 7
 6 0 6 7
 7 7 0 10
 6 6 6 0

This is the key idea behind the Schulze method and is a very clever concept. The first row of this matrix means the strength of the path from option A to option B is 8, the strength from A to C is 6, and the strength from A to D is 7. Explaining the idea of path strength is best done visually. We can convert the raw ranking data to a matrix of pair-wise defeats. Next we can conceptualize the pair-wise defeats matrix as a directed graph as shown in Figure 1.

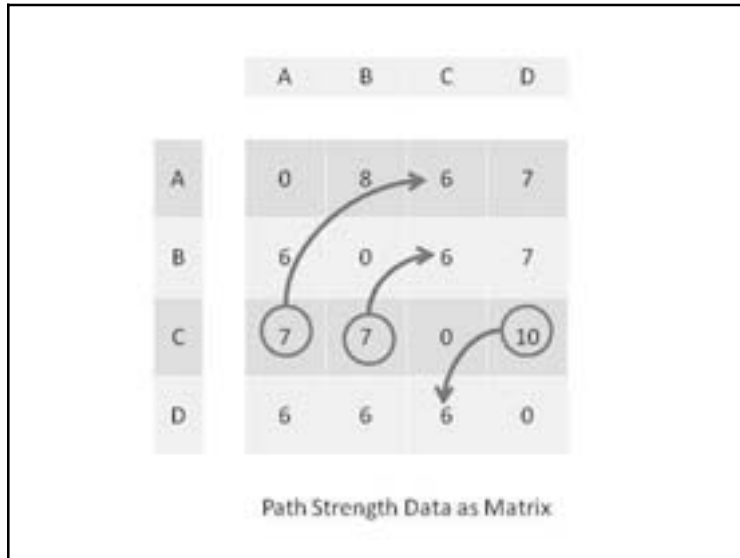


<Figure 1 - Pair-wise Defeats as Matrix and Directed Graph>

The arrow from node A to node B means that option A is preferred by 8 evaluators over option B. We do not put an arrow from node B to node A because option B is preferred by only 3 evaluators over option A, so B dominates A. In this example, all the arrows are unidirectional but if two nodes are preferred by equal numbers of evaluators then you can draw a bidirectional arrow. Now consider the path between node/option A and node/option D. A "beats" B by 8 which in turn beats D by 7. We choose 7, the smaller of these two values, to represent the overall strength of path between A and D. The idea is that the overall strength between two nodes which have intermediate nodes is best represented by the smallest direct node-to-node strength. Notice that we can also take a different path from A to D and say that A beats B by 8, and B beats C by 6, and C beats D by 10, and select the smallest number, 6, as the path strength. In situations where there are multiple paths between nodes, you select the largest path strength, so in this case the final path strength between A and D is 7.

Computing the strength of each pair of nodes can be performed by a variation of the Floyd–Warshall algorithm. The standard Floyd–Warshall algorithm finds the shortest paths in a weighted, directed graph. Once the path strengths have been determined, you can determine the Schulze method winners by checking to see if there is any option where its path strength is greater than or equal to all other options' corresponding path strengths. In this case option C is the only Schulze winner because option C's path strength to option A is 7 (but option A's path strength to B is just 6), option C's path strength to option B is 7 (but C's path strength to C is just 6), and option C's path strength to option D is 10 (but D's path

strength to C is just 6). On the other hand, option A is not a Schulze winner because option A's path strength is better than option B (8 to 6) but option A's strength is worse than option C (6 to 7). Figure 2 shows the matrix of path strengths that corresponds to the data in Figure 1 and demonstrates that option C is a Schulze method winner. Visually, an option is a Schulze winner if every path strength value on that option's row is greater than or equal to the corresponding value across the matrix's main diagonal.



<Figure 2 - Path Strength Data>

Although the Schulze method can be easily performed by hand, the process can be error-prone. Therefore, in practice the Schulze method is usually performed using a software tool. The Schulze method has strengths and weaknesses. One weakness of the Schulze method in some scenarios is that Schulze is not immediately intuitive. In situations where the evaluators are stakeholders, the Schulze method can possibly be viewed as a black box technique that produces a magic result. The primary advantage of the Schulze method is that it has been extensively analyzed and has been shown to meet many favorable criteria. Recall that the Borda count system can sometimes be affected by the removal of an irrelevant alternative. There are many voting system criteria. For example the monotonicity criterion can be loosely stated as the principle that a winning option cannot become a non-winner by one or more evaluators ranking that option higher. There are many proposed voting system criteria, and the Schulze method has been shown to meet most of these criteria. An interesting exception is that the Schulze method violates what is called the participation criterion. In informal terms, it is possible to add to an existing system, new voters who prefer the current winner to some other alternative, which results in the less preferred alternative becoming a Schulze winner. In spite of its shortcomings, based on my experience, the Schulze method is often very effective, especially in situations with a large, educated group of evaluators (meaning they have knowledge of the underlying Schulze method and therefore do not view the algorithm as mysterious) who are determining a policy alternative (as opposed to a product decision) and who are stakeholders in the final result. For example, the Schulze method is used by several Open Source groups to determine general policy decisions and to elect officers. In software development and quality scenarios, the Schulze method is generally an excellent technique to employ.

== Conclusions

When performing a collaborative determination of a best alternative, the interpretation of exactly what *best alternative* means is entirely defined by the approach used. Each technique has pros and cons and there is no single best approach in all situations. Some of the key factors you should consider when using collaborative evaluation techniques include the number of alternatives, the number of evaluators, whether the alternatives are policy decisions or product decisions, and the extent to which evaluators are affected

by the final result of the analysis. In general, you should use more than one collaborative analysis technique when possible in order to use multiple results to cross-validate your analysis. If multiple techniques yield the same conclusion, you gain some measure of confidence in your results. If multiple techniques do not yield the same results then you are well advised to reexamine your assumptions and try and determine why you are getting different results. The situations where the collaborative techniques described in this paper are used are often quite subjective so your experience and intuition should play a big part in interpreting your results. In other words, you should not blindly accept the results of any collaborative technique.

As a general rule of thumb, the pure plurality technique is only effective in situations where ranking all alternatives is simply not feasible. Because the majority runoff technique has the potential to alienate up to 49% of your evaluators, you should generally use this technique primarily to supplement other techniques. The Borda count system, in spite of technical flaws, works quite well especially when the evaluators are stakeholders in the analysis. The Condorcet principle is very simple, and so can often be used to validate the results of other techniques. The Schulze method is an excellent general purpose technique in many software quality situations except when you are dealing with a relatively small group of evaluators who are influenced by the results, or evaluators who do not understand the Schulze algorithm.

Acknowledgment: I am grateful to Joshua Eisenberg (McAfee, Inc.) and Moss Drake (Dentists Management Corp.) who reviewed this paper and made significant improvements to its quality.

== References

Gaetner, Wulf. "A Primer in Social Choice Theory", Oxford University Press, New York, 2006.

McCaffrey, James. "Group Determination of a Best Alternative in Software Testing", MSDN Magazine, November 2008 (in press).

Nurmi, Hannu. "Comparing Voting Systems", Springer-Verlag, 1987.

Saari, D. G. "Basic Geometry of Voting", Springer-Verlag, New York, 1995.

<end of document>