Project Phoenix: VOTE in the 21st Century

William Edward Bailey, III

January 18th, 2016

Advisor: Stephen Slade
Dept. of Computer Science

I. Background

During the late 80s early 90s, Professor Slade wrote the automated decision system known as VOTE. The goal of VOTE was to apply the concepts of qualitative goal-based decision-making to the domain of politics. Members of Congress make decisions on bills on a weekly if not daily basis. What if there were a way to predict how each member of Congress would vote and then use this data to predict how likely the bill would pass? What if a rationale could be given for these decisions so that others would know what factors could sway policymakers in one way or another? Vote was designed to tackle these problems.

At its core, VOTE is an object oriented database equip with a decision-making strategy system. The database contains member objects representing an individual Congressperson. Members have relations with various groups (such as Republicans and Democrats). They make stances on various issues (like civil rights or economics). Each of these stances and relations have varying degrees of importance. This importance is not a hard score but a qualitative measure denoting what the member values. The database contains numerous bills relating to these issues which members must vote on. Members wish to achieve various goals, which are represented as stances on issues. Members may adopt goals of groups they are associated with
when making a decision. VOTE uses various strategies to analyze stances and relations to
determine what decision will achieve the most important goals for the member.

As a final project for CPSC 558 Automated Decision Systems, Krishnan Srinivasan and I
worked on translating VOTE from Lisp into Python. Our project was divided into three main
phases. First, we translated the core code of VOTE stored in the /pol folder. Second, we
configured a Mongo database to store our objects. Third, we proofed our code to produce a
rudimentary working version. At the end of the project, VOTE possessed three strategies with
which it could make decisions for a few members of Congress on a few bills.

The goal of this project is to continue the work from last semester and refine the VOTE
system. The sections below cover the objectives in more detail. The first section discusses the
deliverables – features to be completed by the end of the semester. The second highlights other
areas of improvement that will be addressed as time permits.

II. Deliverables

A. Refining of Core Code

For the first version of VOTE, we proofed three out of twenty-one strategies. The first
stage of this project will be to refine the remaining eighteen. If called upon, each strategy should
produce a decision object or return null if the strategy were unable to produce a decision. In
particular, we will translate the Deeper Analysis strategy, which slowly expands the stances
examined by inferring new stances that are commonly held based upon older stances.

When this stage is complete, all strategies should be proofed and working. Any code
necessary to make a strategy work or to explain why a strategy was applied should also be
translated and proofed.

For more information on the code to be refined, please see here.

B. Flesh out the Database

For the first version of VOTE, we only had a handful of representatives and bills. The Sunlight Foundation has API's which provide information about current members of Congress and current bills. During the second stage of this project, I will write scripts to call the API and then use this information to populate the VOTE database. The database has several objects that will need to be constructed: members, bills, groups, and issues. While APIs provide some useful data on these areas, not all fields needed by VOTE are readily available. Furthermore, much of this data is not directly stated but must be inferred (for example, what issues a representative cares about and how strongly, what groups a representative has relationships with, what is the normative view on issues today). Rather than creating a comprehensive database, the goal is to construct the pipework to receive current data about Congress and feed it into VOTE.

At the conclusion of this stage, VOTE should have all strategies at its disposal to make decisions on how current members of Congress would vote on current bills.

III. Additional Improvements [As Time Permits]

A. VOTE Website

The VOTE engine and database can be stored online. A website can then be created to enable users to query the database or call the engine to see how a given member of Congress will vote for a specific bill.
B. Test Suite

A comprehensive test suite is needed to ensure that VOTE works as expected. Given that functions often have a decent branching factor, multiple unit tests are needed. Functional tests are also needed to ensure that the functions properly interact with each other.

C. Analytics

As a part of the translation from Lisp to Python, we translated several files that produce analytics and statistics about decisions. This code needs to be proofed so that it works properly in Python.

D. A More Complete Database

VOTE will need scripts to more fully populate the database. This could potentially require using linguistic techniques to extract what issues a bill addresses, the stances on the issues implied by the bill, how strongly a member holds to a given issue, etc. The more columns filled per object, the better.

E. Explanation Generation

The original VOTE contained a language generation facility to produce a rationale that a member might use to explain a decision to others. For example, “This bill is unconstitutional and goes against the principles I stand for”. This code will need to be translated from Lisp, proofed, and updated accordingly.
F. A Generic Vote Engine

VOTE's strategies can be applied to many domains. The Congress-specific portions can be generalized to apply to any domain in which an Agent (Member) faces Problems (Bills) that reflect Values (Issues) which the Agent uses to form Goals (Stances) that he/she wishes to fulfill. Scripts or a GUI would also be created to make inputing data easier.

G. Neural Networks

A neural network can be created to predict how members of Congress will vote. The network and VOTE can then be compared to determine which one is better. The comparison could be done on a variety of levels such as accuracy, ability to explain itself, etc.

IV. Conclusion

The next version of VOTE will have all strategies operational and a database containing current data. There are several other areas I would like to address as time permits. In particular, I would like to create the website and invest heavily in a test suite once I am confident of the final structure of the code. The Python version of VOTE has its own Github page, which can be checked for further details about the code: https://github.com/WEB3-GForce/VOTE. I have enjoyed working on VOTE this past semester and look forward to refining it in the weeks to come.