



UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

**CONFIRMACIÓN DE RESULTADOS PARA VOTACIONES CHILENAS
USANDO ALEATORIEDAD VERIFICABLE**

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN COMPUTACIÓN

DIEGO IGNACIO VARGAS JENSEN

PROFESOR GUÍA:
ALEJANDRO HEVIA ANGULO

MIEMBROS DE LA COMISIÓN:
TOMÁS BARROS ARANCIBIA
CLAUDIO GUTIÉRREZ GALLARDO

Este trabajo fue financiado por el proyecto NIST 60NANB18D215.

SANTIAGO DE CHILE
2020

RESUMEN

Las votaciones populares en Chile se utilizan para escoger a los representantes políticos, quienes dirigirán el país y las distintas partes de Estado. Estos procesos sumamente importantes son sometidos a verificaciones que intentan asegurar el correcto reporte de sus resultados, para lo cual interviene el Colegio Escrutador y el Tribunal Electoral. Las verificaciones tratan de reducir la probabilidad de asignar cargos a candidatos incorrectos, pero los métodos y condiciones con los cuales se llevan a cabo parecen no estar definidos de manera estándar, ya que dependen fuertemente del juicio de las personas encargadas. Es más, tampoco se anuncia la forma en la cual se han corregido estos posibles errores, una vez son encontrados, y solo pueden presenciar el procedimiento un grupo selecto de personas. Esto da espacio para equivocaciones, o incluso corrupción, lo cual no es visible al público por la naturaleza cerrada de las decisiones.

El trabajo desarrollado durante esta memoria consiste en el análisis e implementación de un sistema estadístico de confirmación de votaciones, basado en recuentos parciales aleatorizados, métodos previamente utilizados para votación electrónica. Para lograr esto se hace uso del Faro de Aleatoriedad Verificable del CLCERT, lo cual le otorga una transparencia total al sistema. Se construye un prototipo en forma de servicio web, a través del cual es posible verificar votaciones con los métodos desarrollados, y se analiza su rol crucial en una potencial transición hacia una votación electrónica para Chile.

El presente trabajo encuentra que los procesos propuestos de confirmación de votaciones tienen una alta efectividad, identificando los resultados incorrectos con una gran probabilidad. Estos recuentan una cantidad mínima de votos, lo cual genera evidencia a favor o en contra del resultado reportado, produciendo una probabilidad de error. Su uso permite la integración de sistemas electrónicos de votación, reduciendo los riesgos asociados, y disminuyendo el costo de entrada a éstos.

Agradecimientos

Quiero agradecerles a mi papá y mi mamá, por haberme cuidado y guiado incondicionalmente toda mi vida. A mi querida Fran, por apoyarme en todos mis proyectos y empujarme a ser una mejor persona. A mis abuelos, por haberme enseñado tantas cosas. Y a mis amigos, que siempre se han preocupado por mi y me han hecho feliz.

Tabla de Contenido

| | |
|---|-----------|
| 1. Introducción | 1 |
| 1.1. Contexto | 1 |
| 1.2. Objetivos | 2 |
| 1.2.1. Objetivo General | 2 |
| 1.2.2. Objetivos Específicos | 3 |
| 1.3. Descripción General de la Solución | 3 |
| 2. Marco Teórico | 6 |
| 2.1. Votaciones Chilenas | 6 |
| 2.2. Votación Electrónica | 7 |
| 2.3. Auditorías de Confirmación | 7 |
| 2.3.1. Características Generales | 7 |
| 2.3.2. Ballot-Polling Audits | 8 |
| 2.3.3. Comparison Audits | 8 |
| 2.3.4. Métricas de Certeza | 9 |
| 2.3.5. Variaciones Algorítmicas | 9 |
| 2.3.6. Aplicaciones | 10 |
| 2.4. Aleatoriedad Verificable | 11 |
| 2.4.1. Fuente de Aleatoriedad Verificable | 11 |
| 2.4.2. Selección de Muestras Aleatorias | 11 |
| 3. Problema | 12 |
| 3.1. Descripción del Problema | 12 |
| 3.2. Relevancia | 12 |
| 3.3. Requisitos de la Solución | 13 |
| 3.3.1. Características Generales | 13 |
| 3.3.2. Aleatoriedad | 14 |
| 3.3.3. Validación de Elecciones | 14 |
| 3.3.4. Transparencia | 15 |
| 4. Solución | 16 |
| 4.1. Selección de Variantes Algorítmicas | 16 |
| 4.2. Diseño de Algoritmos | 17 |
| 4.2.1. Suposiciones | 17 |
| 4.2.2. Definiciones | 18 |
| 4.2.3. Tamaño de las Muestras | 19 |
| 4.2.4. Selección de Muestras | 20 |

| | | |
|-----------|--|-----------|
| 4.2.5. | Confirmación de Resultados | 21 |
| 4.3. | Adaptación al Caso Chileno | 24 |
| 4.3.1. | ¿Por Qué Auditar? | 24 |
| 4.3.2. | Selección del Tipo de Auditoría | 25 |
| 4.3.3. | Selección de Parámetros | 25 |
| 4.3.4. | Condiciones para Auditar Votos | 26 |
| 4.3.5. | ¿Quién Puede Auditar? | 26 |
| 4.4. | Prototipo | 27 |
| 4.4.1. | Descripción General | 27 |
| 4.4.2. | Arquitectura General | 27 |
| 4.4.2.1. | Dependencias | 27 |
| 4.4.2.2. | Componentes | 28 |
| 4.4.2.3. | Django | 29 |
| 4.4.3. | Diseño de Base de Datos | 29 |
| 4.4.3.1. | Relaciones | 29 |
| 4.4.3.2. | Audit | 29 |
| 4.4.3.3. | RecountRegistry | 31 |
| 4.4.3.4. | SubAudit | 31 |
| 4.4.4. | Servicio Web | 32 |
| 4.4.4.1. | Vistas | 32 |
| 4.4.4.2. | Ejemplo de Uso | 32 |
| 4.4.5. | Limitaciones | 36 |
| 4.4.6. | Extensibilidad a un Producto Terminado | 37 |
| 4.5. | Aplicabilidad a Votación Electrónica | 37 |
| 5. | Validación | 39 |
| 5.1. | Resolución del Problema | 39 |
| 5.2. | Simulación de Casos | 40 |
| 5.2.1. | Configuración General | 40 |
| 5.2.2. | Mayoría Simple | 41 |
| 5.2.3. | Pluralidad | 42 |
| 5.2.4. | Mayoría Absoluta | 43 |
| 5.2.5. | D'Hondt | 44 |
| 5.3. | Optimización para Batch-Level Comparison | 45 |
| 6. | Conclusiones | 49 |
| 6.1. | Resumen | 49 |
| 6.2. | Revisión de Objetivos | 49 |
| 6.3. | Análisis de Resultados | 50 |
| 6.3.1. | Votos Auditados | 50 |
| 6.3.2. | Confirmaciones | 51 |
| 6.4. | Trabajo Futuro | 51 |
| | Bibliografía | 53 |

Índice de Tablas

| | | |
|------|--|----|
| 5.1. | Resultados Reportados Alcalde Maipú, 2016 | 41 |
| 5.2. | Resultados Auditoría Mayoría Simple | 41 |
| 5.3. | Resultados Reportados Presidenciales Primarias, 2017 | 43 |
| 5.4. | Resultados Auditoría Pluralidad | 43 |
| 5.5. | Resultados Reportados Presidenciales Secundarias, 2017 | 43 |
| 5.6. | Resultados Auditoría Mayoría Absoluta | 45 |
| 5.7. | Resultados Reportados Diputado, Primera Región, 2017 | 45 |
| 5.8. | Resultados Auditoría D'Hondt | 47 |

Índice de Ilustraciones

| | | |
|-------|--|----|
| 4.1. | Diagrama de Componentes Principales del Servicio Web | 28 |
| 4.2. | Diagrama UML del Modelo de Datos | 30 |
| 4.3. | Vista Principal | 33 |
| 4.4. | Crear Auditoría | 33 |
| 4.5. | Resumen del Reporte | 34 |
| 4.6. | Ingreso del Recuento 1 | 34 |
| 4.7. | Resumen de la Auditoría 1 | 35 |
| 4.8. | Ingreso del Recuento 2 | 35 |
| 4.9. | Resumen de la Auditoría 2 | 35 |
| 4.10. | Detalle de la Auditoría | 36 |
| 5.1. | Histograma de Votos por Mesa para Mayoría Simple | 42 |
| 5.2. | Histograma de Votos por Mesa para Pluralidad | 44 |
| 5.3. | Histograma de Votos por Mesa para Mayoría Absoluta | 46 |
| 5.4. | Histograma de Votos por Mesa para D'Hondt | 48 |

Capítulo 1

Introducción

1.1. Contexto

Actualmente en Chile los representantes políticos son elegidos por medio de votaciones populares. En éstas, cada ciudadano con derecho a sufragio se dirige a una mesa de votación asignada y se le entrega un papel en donde marcará su preferencia, el cual depositará en una urna. Una vez terminado el periodo de votación, se cierran las mesas y empieza el proceso de conteo. Cada urna se cuenta públicamente y los totales se suman para obtener el resultado final, el cual es publicado por el SERVEL. Luego de esto se entrega las actas de los resultados (2 por cada mesa, hechas de manera independiente) a los distintos Colegios Escrutadores, quienes hacen una verificación simple y de manera privada de los resultados, al día siguiente de la votación.

La verificación del Colegio Escrutador consiste simplemente en comparar las actas y marcar una mesa si es que existe algún error. Esto lo deberían hacer todos los miembros de cada Colegio Escrutador juntos, mirando una misma pantalla, y llegando a un acuerdo sobre el estado de la mesa escrutada en ese momento.

Una vez terminado este proceso, se entrega los resultados al Tribunal Electoral Regional, quien está encargado de revisar y corregir todas las mesas de la región para las cuales se haya apelado (usualmente una municipalidad o un representante político). A esta corrección pueden asistir los candidatos y solo un representante, como se estipula en la ley, no el público en general.

Para realizar esta corrección se le otorga acceso a las actas revisadas por los Colegios Escrutadores, a las urnas con sus votos respectivos y a las apelaciones correspondientes. En caso de encontrar un error “sustancial” se puede llegar a recomtar una mesa completa, sin embargo esto no se hace frecuentemente. Lo más probable es que los miembros del Tribunal ignoren el problema a menos que este sea muy significativo, lo cual, hasta donde se pudo verificar, no está bien definido en la ley.

Una vez terminada esta etapa, el Tribunal Calificador de Elecciones (TRICEL) publica los resultados corregidos. Sin embargo, no publican el detalle que permite transparentar los medios o métodos utilizados para llegar a dicho resultado.

Esto claramente va en contra de la promesa de transparencia y acceso a la información que da el Estado de Chile. Es más, fácilmente se puede ver que un hipotético adversario (como lo podría ser una coordinación de los miembros del Colegio Escrutador o del Tribunal Electoral) podría estratégicamente insertar o ignorar “errores”, de tal forma que cambie el resultado final, eludiendo la verificación actual. Con ello, este ataque podría pasar totalmente desapercibido por la ciudadanía.

Por otro lado, el interés en utilizar dispositivos computacionales para realizar la votación, esto es “votación electrónica” (*electronic voting*), ha ido en aumento recientemente. Cada día hay más países que la adoptan para conducir sus propias votaciones. Sin embargo, la votación electrónica típicamente ha traído un conjunto de problemas, como posibles errores de registro o interpretación de votos, errores de implementación, vulnerabilidades a ataques, entre otros. Cada uno de estos puede ser tanto accidental como intencional, llegando a ser totalmente indetectables, lo que genera cuestionamientos en el sentido que no se debería confiar *a priori* en estas. Es por ello esencial que el resultado de una votación realizada con un sistema electrónico pase por un proceso de verificación o confirmación.

Para que un proceso de verificación sea robusto, debe poder contrastarse con la “intención real del voto” de los electores. Esta intención real típicamente toma la forma de una copia en papel del voto de cada ciudadano. Este debe ser verificable por el votante (*voter verifiable*) a simple vista, pues de otra forma no se puede garantizar que el voto que se está registrando es efectivamente el que se emitió. Además es necesario poder garantizar que este registro no haya sido manipulado de ninguna manera, o de otra forma el proceso de confirmación no tiene mucho sentido, por lo que se debería incluir también un *Compliance Audit* [2, 26].

Se ha propuesto múltiples métodos de auditoría y recuento, que intentan disminuir la probabilidad de que un adversario pueda cambiar su resultado. En particular, en Estados Unidos se ha aplicado varios de estos métodos, pues el *electronic voting* es bastante popular allá. Estos métodos de auditoría parecen tener buenos resultados, aunque cada distrito electoral elige qué método específico usará, dado que no se ha definido un estándar nacional. Además, todo el proceso se realiza manualmente (escoger números aleatorios usando dados, hacer cálculos con lápiz, papel y una calculadora, entre otras cosas).

Este trabajo de título consiste en la implementación de un sistema de Auditorías de Confirmación de Resultados orientado a las votaciones chilenas en papel, más un análisis sobre cómo podría esto ayudar a la transición hacia votación electrónica de manera segura. Para lograrlo se hace uso de un sistema de aleatoriedad pública verificable, el Faro de Aleatoriedad Verificable del CLCERT, y se modifican esquemas de auditorías diseñado para votación electrónica, de tal forma que funcione en votaciones clásicas, como las empleadas en Chile.

1.2. Objetivos

1.2.1. Objetivo General

El objetivo general de esta memoria es diseñar, analizar e implementar un prototipo que permita reducir el error y aumentar la fiabilidad en los resultados electorales chilenos

actuales, usando técnicas de *Risk-Limiting Audits* y aleatoriedad verificable. Dicho sistema debe ser aplicable a todos los tipos de votación usados, sin tener que cambiar la forma actual de verificación. En particular, este proceso no deberá alterar mayormente la operación de los Vocales de Mesa, el Colegio Escrutador ni el Tribunal Electoral durante una elección. El trabajo incluye también un análisis sobre cómo facilitar el cambio de votaciones clásicas a electrónicas, usando *Auditorías de Confirmación de Resultados*, evitando significativamente los problemas de credibilidad de los resultados.

1.2.2. Objetivos Específicos

1. Analizar las posibles variantes estadísticas y algorítmicas de las Auditorías de Confirmación de Resultados y su relación con el sistema de votación chileno.
2. Escoger alguna de las propuestas de métodos para las Auditorías de Confirmación de Resultados para ser adaptada al sistema de votación chileno.
3. Construir un prototipo diseñado para las personas que conducirán estas auditorías.
4. Hacer una selección de votos usando aleatoriedad verificable con un método a elegir en base a los datos pertinentes a la votación que se está auditando.
5. Diseñar una métrica de certeza en base a los resultados de la auditoría que permita determinar si el conteo ha sido verificado o si es necesario seguir auditando, dependiendo del riesgo máximo permitido y el cálculo de la métrica de certeza.
6. Mantener transparencia durante toda la operación, de tal forma que cualquier persona externa pueda verificar que el proceso se hace de manera correcta.
7. Validar el funcionamiento del sistema usando simulaciones de gran escala con datos reales y una prueba piloto a menor escala para comprobar el resultado de alguna de las votaciones dentro de la misma FCFM.
8. Evaluar la aplicabilidad práctica a un sistema de votación electrónico, ya sea presencial o remoto.

1.3. Descripción General de la Solución

La solución consiste principalmente en un sistema que monitorea una Auditoría de Confirmación de Resultados, utilizando fuentes de aleatoriedad verificables para garantizar la integridad de esta (en particular, que la aleatoriedad utilizada en la auditoría estadística no

sea manipulada). Este se diseñó para ser utilizado por alguna entidad del estado que estuviera a cargo de dirigir dicha auditoría, como lo podría ser el Colegio Escrutador o el Tribunal Electoral. Desafortunadamente pareciera ser que con la ley chilena actual ninguna entidad puede tomar ese rol (el Tribunal Electoral puede recontar todos los votos que sean necesarios durante la corrección, sin embargo debe haber alguna justificación, como una inconsistencia en las actas o apelaciones de candidatos), por lo que sería necesario una reforma legal primero.

El sistema le permite al usuario ingresar los datos relevantes de una votación y sus resultados preliminares, elegir qué tipo de auditoría quiere realizar, definir el *risk limit*, cantidad máxima de votos a auditar y *timestamp* exacto del cual se obtendrá el pulso aleatorio del Faro de Aleatoriedad. Luego, se inicia la auditoría y se le indica al usuario qué votos o mesas debe recontar de manera iterativa, hasta que se confirme la votación o se alcance la cantidad máxima de votos. En este último caso, se le recomienda al usuario iniciar un recuento manual completo, ya que no se pudo recolectar suficiente evidencia a favor del resultado reportado.

Las auditorías se registran en una base de datos. Con eso se construye una vista de resumen de la auditoría, para que cualquier persona pueda ver su estado y todos sus pasos, con los datos de entrada y salida. Dado esto, es posible reconstruir la auditoría en su totalidad, alcanzando una transparencia completa. Adicionalmente, se desarrolló un programa de verificación de auditorías para verificar que estas se hagan adecuadamente. Toma los resultados reportados y los recontados, y determina si la conclusión fue correcta.

Todo esto se integra en una página web, lo que le da una interfaz gráfica que facilita su uso. Si bien esta no es particularmente atractiva visualmente, presenta los elementos básicos que debería tener un producto final. Se maneja con un *backend* de Django en Python3.6 y una base de datos PostgreSQL.

Para generar las muestras de votos que se debe recontar durante la auditoría, se utiliza un Generador de Números Pseudo Aleatorios basado en ChaCha20, con su semilla siendo el pulso obtenido del Faro de Aleatoriedad Verificable. Este permite definir una secuencia aleatoria con reposición de votos o mesas, tomando en cuenta cierta probabilidad de elección para cada una.

El funcionamiento del sistema fue verificado usando datos reales de elecciones pasadas, obtenidos del Serval [21]. Esto corresponde al resultado preliminar, el cual se publica inmediatamente luego de terminado el conteo de votos. El TRICEL publica los resultados finales [28], con todas las correcciones hechas por el Tribunal Electoral. Estos están detallados a nivel de mesa, por lo que es posible conocer qué tan diferentes son ambos resultados, sin embargo, no se indica justificaciones para los cambios. Estos no se conocían al momento de correr las simulaciones, por lo que se utilizó las primeras.

En particular, se usó los datos de las elecciones presidenciales primarias y finales, y la elección de diputados de la primera región, del 2017, además de la elección de alcalde para la comuna de Maipú, del año 2016. Con esto se puede cubrir los casos de pluralidad, mayoría absoluta, *D'Hondt* y mayoría simple, respectivamente. Dado que se implementó ambos modos de auditoría (*Ballot-Polling* y *Comparison*), en total se tiene 6 posibles casos diferentes.

Inicialmente se había propuesto conducir una auditoría a menor escala en una votación real de la FCFM. Sin embargo, dadas las circunstancias sociales derivadas de la pandemia, esto no fue posible.

Capítulo 2

Marco Teórico

2.1. Votaciones Chilenas

Las votaciones, también llamadas *funciones de elección social*, en Chile se dividen en tres categorías:

- **Pluralidad:** Se quiere elegir k ganadores dentro de un total de n . Para esto, se escoge aquellos k candidatos con la mayor cantidad de votos. Si $k = 1$, se denomina *Mayoría Simple*. Aplica para elecciones municipales, en el caso de alcaldes [9].
- **Mayoría Absoluta:** Se escoge 1 ganador dentro de un total de n , quien tiene más del 50 % del universo de votos. Si esto no sucede, la votación se convierte en una *Pluralidad* con 2 ganadores. Aplica para elecciones presidenciales [11].
- **Escrutinio Proporcional Plurinominal:** Más conocido como el método *D'Hondt*. Compiten n partidos, de 1 o más integrantes, por k puestos. Iterativamente se le asigna un puesto al partido con mayor cantidad de votos, y se divide los votos totales del partido por la cantidad de asientos asignados + 1. En Chile además se vota por los miembros del partido, lo cual define el orden en el que se asignará los puestos dentro del mismo. Aplica para elecciones parlamentarias [10] y municipales, en el caso de concejales [9].

En Chile, según la ley 18.700 [8], el proceso de votación completo se compone por: conteo de votos por los Vocales de Mesa, verificación del Colegio Escrutador, y corrección del Tribunal Electoral. El conteo se hace de manera pública y solo una vez (a diferencia de otros países, como Dinamarca que divide el conteo en un *rough count* y un *fine count*). Luego el Colegio Escrutador hace una verificación mínima, en la cual solo se revisa las diferencias entre las actas de cada mesa, y cada miembro no puede escrutar más de 200 mesas ni resolver cuestiones de validez. Finalmente el Tribunal Electoral hace las correcciones que considere pertinentes, tomando en cuenta los hallazgos del Colegio Escrutador.

Los detalles sobre las correcciones del Tribunal Electoral no son muy conocidos. En general no se informa al público de estos, y no está claro qué sucede en caso de que se encuentre un error muy grande. Tampoco se considera la agregación de todos los errores a lo largo de las mesas de Chile, el cual podría llegar a ser una magnitud significativa.

2.2. Votación Electrónica

El *electronic voting* se refiere a un sistema o mecanismo de votación apoyado en un formato electrónico para contar o mantener registro de los votos. Votación por Internet es un caso especial de votación electrónica, donde la emisión del voto (*vote casting* en inglés) es realizada de forma remota, usando Internet. En este trabajo, por *electronic voting* se referirá a votación electrónica presencial; esto es, no remota. La votación no remota (o presencial) puede hacerse directamente por máquinas de votación, con las cuales cada votante emite su voto por una interfaz digital. Otra forma es llenar una papeleta, las cuales se guardan en una urna y luego son escaneadas por un lector óptico [3].

La primera forma pareciera ser mucho más conveniente. Sin embargo, para poder hacer un proceso de verificación es necesario dejar una traza en papel la cual se pueda revisar más adelante. Una forma automática de generar esta traza es imprimir un registro del voto recién emitido. Imprimir la traza también tiene sus problemas (como que se agote el papel o la tinta, que se atasque la impresora). Una segunda manera de obtener una traza es dejar que cada persona llene además una papeleta a mano. Para este último, es posible que las personas llenen mal las papeletas o emitan un voto distinto al electrónico, por lo que la manera de verificar los resultados debería tomar en consideración estas cosas.

2.3. Auditorías de Confirmación

2.3.1. Características Generales

Para confirmar los resultados de las votaciones existen las *Risk-Limiting Audits* (RLA) [14], o *Auditorías de Confirmación de Resultados*. Estas son un tipo de recuento diseñado para votación electrónica, que intentan limitar la probabilidad de asignar un resultado incorrecto a una votación a cierto valor escogido. Esto es, asegurar con una probabilidad alta que el resultado del conteo preliminar es efectivamente correcto, o bien, demostrar que existe un error en los resultados.

Notar que si no hay errores en el conteo, la auditoría no puede cambiar el resultado. Sin embargo, si es que hay errores, existe una probabilidad alta de que serán encontrados. Por ejemplo, si se elige una cota de riesgo (o *risk limit*) de 5%, la probabilidad de corregir la votación en caso de haber un problema es al menos de un 95%. Dicho de otro modo, al menos el 95% de las votaciones incorrectamente reportadas serán corregidas.

Para conducir la auditoría es necesario haber hecho un conteo preliminar de todos los votos, cuyo resultado se intenta verificar. El proceso de confirmación extrae y cuenta cierto porcentaje de votos al azar (proceso denominado “auditar votos”). Se intenta minimizar la cantidad de votos auditados, pero esto depende de los márgenes reportados de la votación, del *risk limit*, y de la cantidad total de votos, entre otros factores.

Finalmente se calcula alguna métrica de certeza (que depende del tipo de votación, tipo de auditoría, entre otras cosas), con la que se puede decidir detenerse o continuar auditando

votos. Esto continua hasta que se haya obtenido la certeza suficiente para confirmar la votación, se haya auditado todos los votos, o se haya alcanzado cierto límite de votos predefinido después del cual se inicia un conteo manual completo.

2.3.2. Ballot-Polling Audits

Estas son las RLA más simples. Solo requieren poder auditar votos al azar de manera uniforme dentro del universo de votos. Con esto se construye un total de recuento, el cual se usa para calcular una métrica de certeza de la votación.

Para auditar votos en esta forma, se ha propuesto dos métodos: muestreo de tamaño fijo y de probabilidades independientes. En el primero, se elige cierta cantidad n de votos y se auditan de manera *aleatoria*. En el segundo se escoge una probabilidad p de auditar cada voto de manera independiente. Ambos producen una distribución uniforme, incluso luego de múltiples rondas, pudiendo haber cambiado los valores de n o p .

2.3.3. Comparison Audits

Otro tipo de RLA son las *Comparison Audits*, que pueden llegar a requerir una preparación considerablemente mayor a las *Ballot-Polling Audits*. Necesitan poder dividir el universo de votos en conjuntos o *batches* (no necesariamente del mismo tamaño), conociendo el resultado parcial reportado para cada uno. Luego se audita aleatoriamente algunos conjuntos, comparando el recuento de cada uno con su resultado reportado.

Si los *batches* son de tamaño mayor a 1 voto, como lo podría ser una mesa de votación, se denomina *Batch-Level Comparison*. De lo contrario se le llama *Ballot-Level Comparison*. Esta última suele ser mucho más eficientes en términos de cantidad de votos recontados que *Batch-Level Comparison* (que llega a ser menos eficientes incluso que las auditorías de tipo *Ballot-Polling*), sin embargo no son posibles en el contexto de votaciones clásicas. Esto es porque se necesitaría un registro de la interpretación y ubicación de cada voto particular, lo que significaría una inversión en infraestructura bastante significativa, posiblemente entrando al territorio de votación electrónica.

Una propiedad interesante de las *Comparison Audits* es que su resultado no depende del contenido de los votos recontados, solo de la diferencia con los resultados reportados. Es decir, este tipo de auditorías solo se preocupan de medir la diferencia, o error, entre el *batch* reportado y el recontado, verificando la votación si son suficientemente parecidos. Se podría dar la casualidad de que los *batches* recontados no concuerden con el resultado final de la votación, dándole más votos a un candidato perdedor, sin embargo, esto no tiene repercusión en el proceso de verificación.

2.3.4. Métricas de Certeza

Para calcular la certeza del resultado de una elección, las RLA usualmente definen el experimento “Auditar votos/conjuntos aleatoriamente”, con la hipótesis nula siendo el suceso “Uno de los perdedores reportados obtuvo la misma cantidad de votos o más que uno de los ganadores reportados”. Es decir, uno de los perdedores es en realidad un ganador, o está en empate con uno. Luego, la certeza está dada por $1 - p\text{-value}$, en donde $p\text{-value}$ es la probabilidad de que la hipótesis nula sea correcta, dado los votos auditados. Dicho de otro modo, la elección será confirmada si y solo si en algún momento se encuentra que $p\text{-value} \leq \text{risk limit}$.

Algunas funciones de elección social requieren conducir múltiples auditorías simultáneamente (estas se llamarán sub-auditorías, mientras que la confirmación completa de denominará auditoría), usando la misma muestra aleatoria. Para estos casos, se toma el $p\text{-value}$ del experimento completo como el máximo entre los $p\text{-values}$ de cada una de las subauditorías simultáneas. Similarmente, las subauditorías se suelen dividir en competencias, o *contests*, entre cada ganador y cada perdedor de la votación, con las cuales se intenta verificar de manera independiente que cada ganador obtuvo más votos que cada perdedor. Luego el $p\text{-value}$ de la sub-auditoría es el máximo entre los $p\text{-values}$ de cada *contest*.

Para estimar el $p\text{-value}$ típicamente se usa el *Sequential Probability Ratio Test* (SPRT) de Abraham Wald [31]. Este calcula un factor llamado *likelihood ratio*, que es independiente para cada muestra y se agrega usando el producto (es decir, se puede actualizar el *likelihood ratio* total, simplemente multiplicándolo por el *likelihood ratio* de la nueva muestra). Wald demuestra que, para cualquier $\alpha \in (0, 1)$, si $\text{likelihood ratio} \geq 1/\alpha \implies p\text{-value} \leq \alpha$.

Algunas auditorías de tipo *Comparison* combinan el SPRT con una métrica llamada *Maximum Relative Overstatement* (MRO) [22]. Esta toma un valor entre 0 y 1, y se puede interpretar aproximadamente como cuánto se sobre estimó el aporte de cierto *batch* o voto hacia el resultado reportado. Por ejemplo, si un *batch* recontado es exactamente igual al reportado, su MRO será 0. Pero será 1 si el recuento es “opuesto” al reportado.

2.3.5. Variaciones Algorítmicas

Se ha propuesto múltiples variaciones de RLA desde su creación el 2008 [23]. Cada una de estas intenta incluir más tipos de votación y ser más eficiente en auditar. A continuación se lista y describe los *papers* con las propuestas más relevantes al objetivo de este trabajo:

- *BRAVO: Ballot-polling Risk-limiting Audits to Verify Outcomes* [15]: Como lo dice su nombre, es un esquema para auditorías de tipo *Ballot-Polling*. Se crea para votaciones de pluralidad, y se diseña una transformación simple para mayoría absoluta. Audita votos con el método de muestra de tamaño fijo, y su métrica de certeza es el SPRT. Se construye una matriz de SPRT, la cual representa los *contests* entre las tuplas (ganador, perdedor), actualizando los *likelihood ratios* solo con los votos correspondientes al *contest*.
- *Bernoulli Ballot Polling: A Manifest Improvement for Risk-Limiting Audits* [18]: Se ba-

sa en BRAVO, pero audita votos usando el método de probabilidades independientes. Esto le permite a cada zona (estas pueden ser mesas de votación, regiones, distritos, etc) empezar a auditar votos de manera independiente, sin tener que esperar a que todas estén listas.

- *Verifiable European Elections: Risk-limiting Audits for D'Hondt and Its Relatives* [27]: Describe una transformación para elecciones de tipo D'Hondt y algunas variaciones de esta, para convertirla en una elección de pluralidad. Luego, se puede aplicar BRAVO a esta transformación. También define un algoritmo de tipo *Comparison* usando esta transformación, basado en MRO, pero con una pequeña variante (*Maximum In Contest Relative Overstatement*: MICRO).
- *ClipAudit: A Simple Risk-Limiting Post-Election Audit* [19]: Asume elecciones de pluralidad en el modo *Ballot-Polling*. Promete ser un método mucho más simple que BRAVO, requiriendo computaciones menos complejas, y auditando más eficientemente. Define un factor β dependiendo del total de votos y el *risk limit*. Luego por cada *contest* entre candidatos (A, B), se mantiene registro de la cantidad de votos para cada uno (a, b), hasta que se cumpla $a - b > \beta\sqrt{a + b}$.
- *Sets of Half-Average Nulls Generate-Risk Limiting Audits: SHANGRLA* [24]: Este esquema es mucho más complejo que los anteriores, y promete ser significativamente más eficiente. Existe variaciones para auditorías de tipo *Ballot-Polling* y *Comparison*, se puede usar con todos los métodos para auditar, y usa una métrica de certeza basada en la desigualdad de Kolmogorov [12]. Transforma todas las funciones de elección social a un mismo problema, para el cual se define la auditoría. A cada voto se le asigna un puntaje entre 0 y 1, y se define la hipótesis nula como “El promedio de la serie de puntajes es menor a 0.5”. Es decir, la auditoría se completa si y solo si se encuentra suficiente evidencia de que el promedio de la secuencia es mayor o igual a 0.5. Además existe una implementación *open-source* del algoritmo [25].

2.3.6. Aplicaciones

En Estados Unidos, el estado de Colorado ha aplicado RLA para confirmar los resultados de sus votaciones electrónicas, gracias a una ley adoptada el 2017 [17]. Incluso han desarrollado una plataforma open-source para ayudarlos en esto [16]. Además han desarrollado avances en cómo estratificar auditorías y combinar *batches* de distintos tamaños en un *Comparison Audit*.

En cuanto a la aplicación de estas auditorías a votaciones clásicas (en papel, como se hacen actualmente en Chile), en Dinamarca se hizo un piloto a pequeña escala para confirmar el resultado del distrito electoral de Copenhague [20]. Esta fue del tipo *Comparison*, teniendo bastante éxito y sugiriendo que debería ser posible aplicarlas a votaciones de gran escala. Además, da indicios de sobre cómo abordar este tema en el área política.

2.4. Aleatoriedad Verificable

2.4.1. Fuente de Aleatoriedad Verificable

Para la aleatoriedad verificable existe el “Faro de Aleatoriedad” [4], desarrollado por el *Laboratorio de Criptografía Aplicada y Seguridad Computacional de la Universidad de Chile* (CLCERT). Este emite un pulso aleatorio de 512 bits cada minuto, para el cual usa varias fuentes de entropía, como datos sobre sismos en Chile, *tweets*, entre otros. El sistema cuenta con tecnología criptográfica desarrollada por el CLCERT, en conjunto con el *National Institute of Standards and Technology of United States of America* (NIST), para verificar cada uno de los pulsos emitidos. Adicionalmente, el sistema garantiza que el valor aleatorio generado no es predecible ni siquiera por sus administradores [5].

2.4.2. Selección de Muestras Aleatorias

Los pulsos del Faro de Aleatoriedad se usarán como semilla para un algoritmo Generador de Números Pseudo-Aleatorios (PRNG por sus siglas en inglés) basado en ChaCha20 [13], diseñado por Franco Aníbal Pino Córdova durante el desarrollo de su memoria [6]. Esta logra tomar los 512 bits del pulso del Faro de Aleatoriedad y expandirlos a un valor más grande, manteniendo las propiedades de indistinguibilidad que se requiere, y conservando la entropía del pulso inicial.

Usando esto es posible obtener una secuencia aleatoria, ya sea uniformemente distribuida o con alguna otra función de probabilidad. Esta define una muestra aleatoria, ya sea de votos o de *batches*, la cual podrá ser auditada.

Capítulo 3

Problema

3.1. Descripción del Problema

En la sección 1.1 se describió el proceso de conteo y validación de las elecciones chilenas. Se explicó también la distribución de labores entre los Vocales de Mesa (conteo de votos), los Colegios Escrutadores (verificación de resultados) y los Tribunales Electorales Regionales (corrección de errores), y se mencionan las falencias de dicho proceso, en cuanto a su idoneidad para detectar errores en los totales reportados.

Esto se puede resumir en: falta de transparencia, falta de procesos estándar ante ciertas situaciones, no considerar errores globalmente, y el no poder cuantificar la certidumbre de los resultados reportados.

La resolución de estos problemas genera adicionalmente una oportunidad: convertir las votaciones chilenas del formato en papel a electrónico. Si la solución se construye cuidadosamente, se podría aplicar también a este nuevo tipo de votación. De esta forma se estaría validando sus resultados de manera cuantitativa, mitigando las vulnerabilidades inherentes a esta tecnología.

3.2. Relevancia

Las elecciones en Chile, y el mundo en general, son mecanismos de extrema importancia, que determinan a los representantes políticos e influyen de gran manera el futuro del país. Tienen repercusiones económicas, ambientales, sociales, entre muchas otras cosas que afectan las vidas de millones de personas.

Es por esto que se deben resguardar de no solo errores accidentales que puedan ocurrir durante su desarrollo, sino que también de agentes maliciosos que podrían querer alterar el resultado para su propio beneficio. Sin embargo los mecanismos de prevención actuales son insuficientes para lograr esto de manera efectiva.

Es más, en este momento no existe una forma de cuantificar la seguridad en los resultados ni procesos estandarizados para lidiar con esas situaciones. Esto deja al sistema de elección

particularmente vulnerable. Algo muy similar ocurre en los sistemas de votación electrónicos que no tienen resguardos, en los cuales se puede cambiar totalmente el resultado sin dejar ningún registro.

Por otro lado, el estado de Chile promete a todos sus ciudadanos el acceso libre a información referente a los órganos de la Administración del Estado, según la ley 20.285 [7]. Esto incluye toda la información relativa a los procesos oficiales de elección. Sin embargo, no es claro que las decisiones tomadas en la práctica a puertas cerradas, como sucede con los Tribunales Electorales, permitan que esto se cumpla.

El uso de una herramienta capaz de conducir y publicar Auditorías de Confirmación de Resultados podría resolver o limitar de gran manera el impacto de todos estos problemas: tales herramientas entregan una métrica clara de seguridad para los resultados, pueden encontrar y corregir errores en éstos con una probabilidad arbitrariamente alta, y entregan pasos claros a seguir que pueden ser verificados por quien quiera hacerlo.

Más allá de eso, existe una tendencia creciente en los países desarrollados de acelerar el proceso de conteo de votos mediante sistemas electrónicos. Esta tendencia se ha transmitido a otros países, similares a Chile, por lo que es bastante seguro asumir que tarde o temprano se intentará implementar *electronic voting* aquí. Sin embargo, este tipo de votación trae riesgos bastante altos, los cuales pueden ser mitigados con los mismos sistemas de confirmación que se propone para votaciones clásicas.

3.3. Requisitos de la Solución

3.3.1. Características Generales

Una solución satisfactoria debe ser capaz de eliminar todos los procesos inciertos de una elección clásica. Esto es, tener claramente definidos todos los pasos a seguir, para cualquier situación que pueda surgir dentro de este contexto. Además, no se deberá necesitar depositar la confianza en ninguna persona u organismo para asegurar el correcto desarrollo del procedimiento.

Adicionalmente, la solución debe poder ser capaz de validar una votación correctamente reportada el 100 % de las veces, con la esperanza de la cantidad de votos auditados siendo una fracción significativamente más pequeña que el total, aunque bajo ciertas condiciones se puede requerir un conteo manual completo. Si el resultado de la votación es incorrecto, la solución debe ser capaz de determinar esto con una probabilidad mayor o igual a la escogida por el usuario.

Esta solución no debe interferir de mayor manera los procesos actuales de revisión y corrección. Más específicamente, no debe requerir la modificación de ningún proceso existente, ni un trabajo de preparación significativo. Idealmente se debería poder introducir con tan solo cambios mínimos en la forma de almacenado y ordenación de los votos.

Finalmente, es deseable que la solución pueda ser aplicada no solo a votaciones clásicas, sino que a votación electrónica también. Se podría tener que modificar levemente la solución para ser aplicada a un tipo de votación electrónica, cuyas características y requisitos deberán ser bien definidos.

3.3.2. Aleatoriedad

Para escoger muestras aleatorias es necesario tener una semilla de aleatoriedad y un generador de números pseudo-aleatorios (o *pseudo-random number generator*, PRNG). Es importante que la semilla sea verificablemente aleatoria, para prevenir su manipulación o selección por conveniencia. La fuente de esta aleatoriedad debe ser de acceso público, pues se espera que cualquier persona sea capaz de verificarla. Los algoritmos de verificación deben ser abiertos, con una implementación *open-source*, para no perder la transparencia.

El PRNG debe ser lo suficientemente robusto como para poder generar varios miles, o incluso millones de valores, sin perder sus propiedades. Su implementación también debe ser *open-source*, de forma que no haya duda de su correcto funcionamiento. Finalmente, se considera deseable que este pueda tomar provecho de toda la semilla aleatoria, sin perder mucha entropía.

Adicionalmente se debe cumplir que no exista ninguna fuente de aleatoriedad distinta a la semilla obtenida del Faro de Aleatoriedad. Tampoco puede haber dependencias hacia factores irrelevantes del sistema ni de los datos (por ejemplo el orden en el que se entrega los resultados reportados). Esto es para prevenir que el proceso de verificación se vea afectado de forma irreproducible, con lo cual no podría ser replicable por observadores externos.

3.3.3. Validación de Elecciones

La validación de elecciones debe entregar una métrica que diga explícitamente qué tan seguro es el resultado. Esto se deberá poder interpretar como una probabilidad, con la cual se determinará si se tiene suficiente certeza en el resultado o no.

El proceso de confirmación necesitará sacar muestras aleatorias de votos, con las cuales se calculará la certeza. Estas muestras deberán considerar una porción significativamente menor al total de votos. Este límite lo podrá definir el usuario antes de empezar la validación, y si se llega a superar, la votación se considerará como no confirmada. Para solucionar esto se podrá hacer un conteo manual completo.

El usuario podrá también definir la certeza que se quiere para la votación, y el proceso de validación deberá seguir pidiendo recuento de muestras hasta que se logre esa certeza o se llegue al límite de votos. El tamaño de cada muestra se deberá ajustar en base al resultado reportado y los votos auditados, para optimizar la cantidad de votos recontados, con la menor cantidad de rondas posibles, sin sobrepasar por mucho los recuentos necesarios.

3.3.4. Transparencia

La transparencia, si bien puede parecer un requerimiento un poco mundano, es uno de los puntos más importantes. Si el proceso de validación no es transparente, poco valen sus resultados, puesto que no habría seguridad para las personas externas de que este se realizó correctamente. Luego, fácilmente se podría caer en el mismo problema actual, teniendo que confiar en las personas encargadas de conducir el proceso para realizar la verificación.

Para evitar esto es necesario que la solución sea totalmente transparente. Esto significa tener algún mecanismo a través del cual se pueda reportar toda la información relevante para la verificación. Se considera los datos iniciales, como lo sería la certeza requerida, cantidad máxima de votos recontados, resultado reportado y configuraciones iniciales, además de todos los recuentos de muestras. De esta forma se podrá reconstruir el proceso en su totalidad, sin ninguna ambigüedad.

De la mano a esto va el proceso de recuento de votos. De poco sirve poder reconstruir la verificación completa, si no se tiene seguridad en que el recuento de muestras se hizo correctamente. Para esto se deberá recurrir a métodos como *video streaming* u otros parecidos, para mantener registro del procedimiento.

Capítulo 4

Solución

4.1. Selección de Variantes Algorítmicas

Para desarrollar este proyecto primero fue necesario elegir qué variantes algorítmicas de *Risk Limiting Audits* se utilizaría. Estas de preferencia deberían ser aplicables a todas las funciones de elección social usadas en Chile, por simplicidad de la solución. Además se requiere tener la opción de elegir entre Ballot-Polling y Batch-Level Comparison (Ballot-Level Comparison queda claramente descartado, por su inaplicabilidad a votaciones en papel), ya que la primera suele ser más eficiente, pero la segunda más simple de llevar a cabo, para el caso de votaciones en papel. Esto ya que es mucho más fácil recontar una urna completa que escoger unos pocos votos específicos.

Con este fin, es necesario primero entender las ventajas y garantías que ofrecen las RLA. En primer lugar, estas definen un procedimiento estructurado de qué acciones se deben tomar para cada situación, quitando la ambigüedad. Esto además significa que no se depositará la confianza en una persona u organismo en particular para tomar las decisiones correctas, sino que ya no será necesario tomarlas, pues todas las condiciones y posibles acciones estarán definidas con anterioridad. Esto, de la mano con el uso de aleatoriedad verificable, impiden o dificultan de gran manera la posibilidad de coerción, corrupción y manipulación de los resultados.

Por otro lado, estas auditorías garantizan cierta probabilidad de corregir una votación mal reportada, mientras que no pueden cambiar el resultado de una correcta. Además, se intenta minimizar la cantidad de votos recontados, lo que ahorra tiempo y recursos. Esto se discute más adelante, en la Sección 4.3, junto a otras consideraciones que se debe tener para conducir una auditoría en el caso de Chile.

Para esto se consideró las siguientes opciones: BRAVO, BBP, ClipAudit, MICRO y SHANGRLA. Las primeras 3 son de tipo Ballot-Polling, MICRO es para Ballot-Level y Batch-Level Comparison, y SHANGRLA es aplicable a todas las formas. Se eligió este grupo inicial por ser los algoritmos más conocidos (BRAVO y MICRO), o por prometer una ventaja significativa en algún ámbito sobre los anteriores (BBP, ClipAudit y SHANGRLA). Por defecto se escogería los algoritmos “clásicos”, a menos que uno de los otros produjera realmente mejores resultados.

Primero se descartó BBP, ya que es equivalente a BRAVO, y requiere una forma levemente más compleja de auditar. Su diseño es más apto para auditar las distintas regiones, mesas o subdivisiones de la zona de forma paralela y sin tener que esperar a que cada una de ellas esté lista para iniciar el proceso, lo cual podría ser deseable un sistema real. Sin embargo esto agregaría complejidad innecesaria para un prototipo de esta escala.

En segundo lugar se consideró ClipAudit, que dice ser mucho más simple y eficiente que BRAVO. Propone un diseño novedoso, el cual no necesita usar SPRT. Sin embargo, su definición no parecía suficientemente fundamentada y desarrollada, y utiliza variables que no pueden ser calculadas analíticamente. Por esto se decidió no seguir con ClipAudit.

Luego se analizó SHANGRLA. La complejidad de este algoritmo es notablemente mayor a la de los otros, y necesitó de un análisis profundo para empezar a entenderlo. Este transforma todas las funciones de elección social al mismo problema, asignando a cada voto un puntaje entre 0 y 1, y en donde la hipótesis nula es “El promedio de la serie completa de puntajes es menor a 0.5”.

Lamentablemente, luego de haber pasado un par de semanas trabajando con el código abierto de este algoritmo, se encontró que no es compatible con votaciones de más de unos miles o cientos de miles de votos, el cual es el caso en Chile. Esto es porque uno de los cálculos para obtener la certeza de los resultados reportados diverge, invalidando todo el proceso. Así, se tuvo que descartar esta opción también.

Finalmente, se concluyó que ninguno de los algoritmos más novedosos presentaba suficientes beneficios o era aplicable al caso chileno. Por esto se decidió implementar el sistema basándose en BRAVO, para el caso de Ballot-Polling, y MICRO, para Batch-Level Comparison.

4.2. Diseño de Algoritmos

4.2.1. Suposiciones

La solución ideada se basa en las proposiciones de *Risk Limiting Audits*. Sin embargo estos asumen ciertas cosas razonables, las cuales son necesarias para llevar a cabo una auditoría de manera efectiva. La principal suposición es la integridad de la traza física. Esto es, tener la seguridad de que la intención real de los electores no ha sido manipulada. Esto puede ser comprobado usando otro tipo de auditorías, llamadas *Compliance Audits*, las cuales intentan verificar que los procedimientos y políticas regulatorias de una organización se desarrollan de manera correcta.

También se asume que los recuentos de votos ingresados al sistema son correctos. Esto se debe verificar de manera externa, dejando algún registro público de que la auditoría se llevó a cabo con la metodología correcta. Se puede lograr con una transmisión en vivo del recuento, u otras formas que permitan al público verificar lo que se está haciendo en tiempo real.

La proporción de votos inválidos, esto es nulos y en blanco, en general tiende a ser mucho menor al total de votos. Es más, con la excepción de votaciones proporcionales, no tiende a haber candidatos con una cantidad menor o cercana a la cantidad de estos votos. Por esto, se toman como una fracción despreciable y no se consideran en algunos cálculos, como la estimación de tamaños de muestras, ya que los simplifica de gran manera.

Otra suposición, que no es fundamental, sino que está dada por los algoritmos de auditoría escogidos, es el conteo de votos. Se asume que la cantidad de votos reportados por cada mesa es correcto. Se podría idear una solución que tome en cuenta diferencias entre la cantidad de votos reportados y los reales, pero en este caso es necesario tener la capacidad de auditar cada uno de los votos reportados. Para esto se podría usar el método de *Phantoms to Evil Zombies* [1], el cual está diseñado para considerar votos que fueron contados inicialmente para generar el reporte, pero no pueden ser auditados (usualmente para mantener el anonimato cuando la mesa de votación es muy pequeña u otros casos parecidos). Este asume que toman el valor menos favorable para la auditoría, y la acercan lo más posible a un recuento manual completo. No está claro cómo se haría esto en la práctica, pero es una idea que vale la pena explorar.

4.2.2. Definiciones

A continuación se listan algunas definiciones importantes, las cuales serán utilizadas más adelante:

α *risk limit* $\in (0, 1)$.

N Cantidad de votos totales.

S Número de ganadores o puestos que serán asignados.

M Cantidad máxima de votos a auditar.

m Cantidad de votos auditados hasta el momento.

\mathcal{W} Conjunto de candidatos ganadores o partidos que ganaron al menos 1 puesto, según el reporte.

\mathcal{L} Conjunto de candidatos perdedores o partidos que perdieron al menos 1 puesto, según el reporte.

t_c Cantidad de votos reportados para candidato o partido c .

a_c Cantidad de votos verdadera para el candidato o partido c .

$e_c = t_c - a_c$, error para el candidato o partido c .

d_i Divisor para columna i para elecciones proporcionales. En el caso de D'Hondt, $d_i = i + 1$ para $i = 0 \dots S - 1$. Para elecciones de pluralidad y mayoría absoluta siempre se tiene que $i = 0$.

$p_{ps} = t_p / d_s$.

\mathcal{W}^P Pares (p, s) con los S mayores valores de p_{ps} .

\mathcal{L}^P Pares $(p, s) \notin \mathcal{W}^P$.

s_c Fracción de votos reportados para el candidato c .

p_c Fracción de votos reportados para el candidato c , tomando en cuenta solo los votos válidos (todos excepto nulos y en blanco). En este proyecto se asume que la cantidad de votos inválidos es insignificante, por lo que se toma $s_c = p_c$.

$$s_{w\ell} = s_w / (s_w + s_\ell), w \in \mathcal{W}, \ell \in \mathcal{L}.$$

$$S_w(p) = \max\{s : (p, s) \in \mathcal{W}^P\}.$$

$$S_\ell(p) = \min\{s : (p, s) \in \mathcal{L}^P\}.$$

γ Factor de seguridad usado para escalar la auditoría. Si γ es cercano a 1, se necesitará muy pocos votos para confirmar la auditoría, pero aumenta significativamente con cada error encontrado. Si es más cercano a 0, se necesitará más votos, pero no aumentará tanto con los errores. Usualmente se recomienda el valor de 0.95.

Además se referirá por “Esquema de auditoría” al conjunto completo de algoritmos y reglas necesarias para llevar a cabo el proceso. Esto incluye la generación de muestras, forma de auditar votos, métricas de certeza y restricciones para cada uno de estos.

4.2.3. Tamaño de las Muestras

Para determinar el tamaño de las muestras se utilizan tres algoritmos diferentes, que dependen del tipo de auditoría y de la función de elección social. En particular, para auditorías Ballot-Polling y las funciones de elección social pluralidad y mayoría absoluta, se usa el Average Sample Number (ASN), el cual aproxima la cantidad de votos que debería ser necesario auditar para confirmar la votación en una ronda. Este se define como:

$$ASN = \max_{w \in \mathcal{W}, \ell \in \mathcal{L}} \frac{\log(1/\alpha) + z_w/2}{p_w \cdot z_w + p_\ell \cdot z_\ell}$$

Para $z_w = \ln(2s_w)$ y $z_\ell = \ln(2 - 2s_w)$, según lo propuesto en BRAVO. D’Hondt usa un método distinto para obtener el tamaño de la muestra, que está definido en [27]. Se calcula con:

$$s = \frac{\log(1/\alpha)}{\log\left(\frac{\gamma}{1-\gamma/(N \cdot u)} + 1 - \gamma\right)}$$

Y u se obtiene por:

$$u = \max_{(w, s_w) \in \mathcal{W}^P, (\ell, s_\ell) \in \mathcal{L}^P} \frac{d_{s_w} + d_{s_\ell}}{d_{s_\ell} \cdot t_w - d_{s_w} \cdot t_\ell}$$

En el caso de Batch-Level Comparison, se usa una estrategia diferente ideada durante el desarrollo del proyecto. Se calcula la cantidad mínima de mesas con recuento exactamente igual al reportado que se necesitaría para confirmar la votación, y se divide por el factor de

seguridad γ para ajustar por la cantidad de errores que se cree que hay. Para eso, se encuentra el valor máximo que puede aportar una mesa al *likelihood ratio*, y ya que estos factores se multiplican, se llega fácilmente la cantidad mínima de mesas necesarias. Luego el tamaño de la muestra está dado por:

$$s = \frac{\log(\frac{1}{\alpha})}{\gamma \cdot \log(LR)}$$

Con LR siendo el máximo *likelihood ratio* que puede aportar una mesa, el cual quedará definido más adelante. En todos los casos, si se necesita escalar la auditoría (esto es, seguir auditando votos), se recalcula el tamaño de la muestra considerando el *risk limit* como $\frac{\alpha}{p_value}$, con *p-value* siendo la estimación para la probabilidad de que el resultado de la votación sea incorrecto. Si bien este ajuste no tendrá un efecto en los resultados de la auditoría, reduce de gran manera la cantidad de rondas necesarias y evita que se deba auditar muchos votos en una ronda cuando se está cerca del objetivo.

Usualmente todos estos tamaños de muestras son multiplicados por algún factor $r > 1$, para incrementar el tamaño de cada muestra, y así aumentar la probabilidad de confirmar la votación en una sola ronda. Esto tampoco tiene efecto alguno en los algoritmos de verificación. Para efectos de este trabajo, se asume $r = 1$.

4.2.4. Selección de Muestras

Las muestras son conjuntos de votos que serán recontados para tratar de confirmar la votación. Estos se obtienen usando un PRNG, el cual genera números aleatorios, con los que se calcula una lista de índices de votos. Esta es la muestra aleatoria.

Dados los esquemas de *Risk Limiting Audits* escogidos, los votos y mesas deben ser muestreados con reposición. Además, las auditorías Ballot-Polling deben hacerlo con una distribución uniforme, mientras que las Batch-Level Comparison definen una probabilidad de elección para cada *batch*, dependiendo de la cantidad de votos en ella y otras características generales de la votación.

Para este proyecto se decidió por elegir muestras de tamaño fijo, ya que usar las muestras de probabilidades independientes no trae muchos beneficios en este contexto, mientras que complejiza significativamente el desarrollo. El algoritmo definido para ambos tipos de auditoría es el mismo, teniendo en cuenta que para Ballot-Polling la probabilidad de elegir cada voto es el mismo. También cambia el universo: en Ballot-Polling son votos, y en Batch-Level Comparison se refiere a *batches*, lo que corresponde a mesas de votación en este contexto.

Sea \mathcal{P} el universo de votos o mesas disponibles para elegir, $r \in [0, 1]$ un número pseudo aleatorio generado por el PRNG, y w la lista de pesos o probabilidades de elección para cada elemento de \mathcal{P} . Sea también la función *approx_index_of*, que recibe un arreglo numérico y un valor numérico i , y retorna el índice del mayor elemento menor o igual a i , en la suma acumulativa del arreglo. Luego para elegir 1 elemento x de la muestra aleatoria, se hace lo siguiente:

$$\begin{aligned}
W &= \sum w \\
r &= PRNG.random() \\
i &= r \cdot W \\
index &= approx_index_of(w, i) \\
x &= \mathcal{P}[index]
\end{aligned}$$

Si la muestra aleatoria es de tamaño s , este proceso se repite s veces, generando un valor aleatorio r cada vez. Es importante mantener el orden aleatorio en la muestra, ya que, para evitar tener que mantener el estado del PRNG, se decidió generar la muestra aleatoria completa antes de iniciar el recuento. Esto es, producir una secuencia aleatoria de votos de tamaño M , para Ballot-Polling, y una que comprende todas las mesas de la votación necesarias para Batch-Level Comparison, inmediatamente luego de obtener la semilla aleatoria.

Si en la ronda i se necesita una muestra de tamaño s_i , se usan los primeros s_i elementos de la muestra completa, los cuales son removidos antes de la siguiente ronda. En Batch-Level Comparison no es posible tener este nivel de granularidad, por lo que se toma el promedio de votos por mesa y se selecciona la cantidad de mesas que alcancen a llegar al tamaño requerido de la muestra. Esto significa que el tamaño real puede ser mayor o menor al dicho anteriormente, pero no tiene ningún efecto en los cálculos de certeza, ya que cada *batch* puede aportar exactamente lo mismo a la auditoría, independiente de su tamaño.

Para Batch-Level Comparison, la distribución de probabilidad de elección para cada *batch* está definida de la siguiente manera: Por cada *batch* i con n_i votos, se calcula su peso u_i .

$$u_i = \max_{w \in \mathcal{W}, \ell \in \mathcal{L}, w \neq \ell} \frac{v_{iw} - v_{i\ell} + n_i}{t_w - t_\ell}$$

En donde v_{ic} es la cantidad de votos para el candidato c en el *batch* i . Luego, asumiendo que hay B *batches*, la probabilidad de elegir el *batch* i está dada por:

$$P_i = \frac{u_i}{\sum_{j=0}^B u_j}$$

Reemplazando estos valores en el algoritmo descrito anteriormente, se logra obtener una muestra aleatoria de *batches* de diferentes tamaños.

4.2.5. Confirmación de Resultados

La confirmación de resultados se refiere específicamente a verificar que los candidatos ganadores o puestos asignados son los correctos. No se está interesado en tratar de confirmar el valor numérico de los resultados, el cual puede diferir un poco del valor reportado, pero no tendrá ninguna consecuencia práctica. Para hacer esto, las Auditorías de Confirmación de Resultados se llevan a cabo con el siguiente algoritmo o alguna variación de este:

1. Definir valores de *risk limit* α , cantidad máxima de votos por auditar M , $m = 0$ y $max_p_value = 1.0$.
2. Elegir una cantidad s de votos por auditar y auditar una muestra aleatoria, incrementando m en s .
3. Actualizar max_p_value en base a los resultados del recuento de votos.
4. Si $max_p_value \leq \alpha$ detener la auditoría y marcar la votación como confirmada.
5. Si $m \geq M$ detener la auditoría e iniciar un conteo manual completo.
6. Volver al paso 2.

La mayoría de las propuestas para *Risk Limiting Audits* difieren en el paso 3, con algunas definiciones extra en el paso 1. En el caso de BRAVO, la auditoría además inicializa la variable T como:

$$T_{w,\ell} = 1.0, \forall w \in \mathcal{W}, \ell \in \mathcal{L}$$

Luego, para actualizar max_p_value , por cada voto recontado, si este es para un ganador reportado w , entonces para todo $\ell \in \mathcal{L}$, multiplicar $T_{w\ell}$ por $2s_{w\ell}$. Si el voto es para un perdedor reportado ℓ , entonces para todo $w \in \mathcal{W}$, multiplicar $T_{w\ell}$ por $2(1 - s_{w\ell})$. Es importante notar que la actualización del *contest* entre w y ℓ solo se hace si $T_{w\ell} < 1/\alpha$, ya que se requiere verificar cada uno independientemente, y no todos al mismo tiempo. Luego, el nuevo max_p_value será:

$$max_p_value = \max_{w \in \mathcal{W}, \ell \in \mathcal{L}} \frac{1}{T_{w\ell}}$$

Esto solo funciona para votaciones de tipo pluralidad. Si se quiere auditar votaciones de mayoría absoluta, se hace un cambio muy simple: Se agrupa todos los candidatos perdedores en un solo candidato *losers*, y se hacen competir en una pluralidad contra el candidato ganador *winner*. Ya que es una mayoría absoluta, el candidato ganador debería tener más de la mitad de los votos, por lo tanto debería también ganar en una pluralidad contra todo el resto de los candidatos.

Para aplicarlo a votaciones de tipo D'Hondt, es necesario hacer algunas transformaciones que la convierten efectivamente en una pluralidad. Primero, antes de empezar la auditoría, se define 2 factores constantes que dependen solo de los resultados reportados. Estos son:

$$\begin{aligned} \gamma_{pS_w(p)qS_\ell(q)}^+ &= \frac{t_p}{t_p + t_q} \cdot \frac{d_{S_w(p)} + d_{S_\ell(q)}}{d_{S_w(p)}} \\ \gamma_{pS_w(p)qS_\ell(q)}^- &= \frac{t_q}{t_p + t_q} \cdot \frac{d_{S_w(p)} + d_{S_\ell(q)}}{d_{S_\ell(q)}} \end{aligned}$$

Notar que si se intercambia los índices p y q en una expresión, se obtiene la otra. Luego, por cada voto recontado, si es para un partido $w \in \mathcal{W}$ que reportó ganar al menos un puesto, para todo $\ell \neq w \in \mathcal{L}$, $T_{w\ell}$ se multiplica por $\gamma_{pS_w(p)qS_\ell(q)}^+$. Si es para un partido $\ell \in \mathcal{L}$ que

reportó perder al menos un puesto, para todo $w \neq \ell \in \mathcal{W}$, $T_{w\ell}$ es multiplicado por $\gamma_{pS_w(p)qS_\ell(q)}^-$.

Notar que el factor de la izquierda de γ^+ y γ^- produce el mismo resultado que $s_{w\ell}$ y $(1 - s_{w\ell})$ respectivamente, lo cual corresponde a los factores definidos para pluralidad. También, por las restricciones impuestas para pluralidades y mayorías absolutas en la sección de definiciones, siempre se dará que para estas funciones de elección social el factor de la derecha es igual a 2, no así para D'Hondt.

Ya que en Chile se elige no solo la cantidad de puestos por cada partido, sino que también los miembros del partido que serán asignados, es necesario hacer más verificaciones que lo recién descrito. Más específicamente, por cada partido $p \in \mathcal{W}$, se debe conducir paralelamente otra auditoría, esta vez para una pluralidad. El objetivo es verificar que los miembros del partido que serán asignados efectivamente obtuvieron más votos que aquellos que no serán asignados. Como se mencionó anteriormente, esto no tiene un mayor efecto en el proceso de auditoría más que requerir una mayor cantidad de votos, ya que la nueva verificación se hace al mismo tiempo, usando las mismas muestras aleatorias.

No es difícil ver que estas transformaciones pueden ser generalizadas para funcionar con pluralidades, y por ende con mayorías absolutas. De hecho, si se considera a cada candidato como un partido de tan solo un miembro, todas las transformaciones son aplicables directamente, sin tener un impacto en la cantidad de votos necesarios. Para simplificar este trabajo y hacer el desarrollo más extensible, esto fue lo que se hizo en el prototipo.

Las auditorías de tipo Batch-Level Comparison utilizadas en este proyecto funcionan de manera parecida, pero la variable T solo contiene un valor global para toda la auditoría, en vez de ser una matriz. Además, se tiene que los valores utilizados para actualizar la variable dependen de la diferencia entre el *batch* recountado y el reportado, a diferencia de los casos anteriores, en los cuales es una constante.

Debido a que el algoritmo para Batch-Level Comparison no estaba definido explícitamente en [27], sino que se aludía como una generalización para Ballot-Level Comparison, se decidió modificar este último para que funcionara con *batches*. Lo que se hace es considerar cada mesa como si fuera una sola papeleta, con múltiples votos. Dicho de otro modo, se agrupa cada mesa en un solo voto con una o más preferencias marcadas. Esta variación de Ballot-Level Comparison está bien definida en [27], y la diferencia con la versión simple es nada más que una multiplicación.

Para esto primero se define:

$$u = \max_{w \in \mathcal{W}, \ell \in \mathcal{L}, w \neq \ell} \frac{d_{S_\ell(\ell)} + d_{S_w(w)}}{d_{S_\ell(\ell)} \cdot t_w - d_{S_w(w)} \cdot t_\ell}$$

$$U = N \cdot u$$

Luego, por cada *batch* en la muestra, se calcula:

$$MICRO = \max_{(p_w, s_w) \in \mathcal{W}^P, (p_\ell, s_\ell) \in \mathcal{L}^P} \frac{d_{S_\ell} \cdot e_{p_w} - d_{S_w} \cdot e_{p_\ell}}{d_{S_\ell} \cdot t_{p_w} - d_{S_w} \cdot t_{p_\ell}}$$

$$D_m = \frac{MICRO}{u}$$

Ahora se actualiza el valor de T :

$$T_m = T_{m-1} \left[\gamma \frac{1 - D_m}{1 - 1/U} + 1 - \gamma \right]$$

Y el valor máximo LR que puede aportar un *batch* está dado por $D_m = 0$:

$$LR = \gamma \frac{1}{1 - 1/U} + 1 - \gamma$$

Finalmente el valor de max_p_value será:

$$max_p_value = \frac{1}{T_m}$$

Estas operaciones se hacen pensando en una votación de tipo pluralidad, sin embargo, usando las transformaciones anteriormente explicadas, es posible generalizarlas a votaciones de mayoría absoluta y D'Hondt también. Con esto se cubre todos los casos requeridos.

4.3. Adaptación al Caso Chileno

Como ya se ha mencionado antes, los algoritmos descritos en la sección anterior fueron diseñados para ser aplicados a sistemas de votación electrónicos. En este trabajo se quiere adaptarlos para ser compatibles con las votaciones realizadas en Chile, las cuales solo son en papel y no cuentan con ningún tipo de automatización. Adicionalmente, cada votación es escrutada, revisada y corregida por distintas entidades, cada una con sus propios poderes limitados.

Esto genera múltiples restricciones para las auditorías, que se salen del ámbito técnico y entran al área política y de administración. A continuación se discute algunos de estos problemas y cómo podrían ser enfrentados de manera que no imponga dificultades o costos muy altos, pero tampoco le quiten validez al proceso.

4.3.1. ¿Por Qué Auditar?

Algunas personas pueden pensar que conducir una auditoría es demasiado trabajo, teniendo que configurar sistemas, usar aleatoriedad verificable, pasar por rondas, entre otras cosas. Se podría sugerir en vez, simplemente contar el 10% de todos los votos, o algo parecido, lo cual podría tomaría menos tiempo, ya que se puede hacer inmediatamente y sin tener que esperar a los pulsos aleatorios ni otros retrasos. Sin embargo esto pierde algunas de las garantías que ofrecen las Auditorías de Confirmación de Resultados.

Por un lado, una auditoría intenta limitar la cantidad de votos recontados lo más posible. Eso significa que tomando una estrategia como la anterior podría conllevar a recontar muchos más de los necesarios. Esto desperdicia recursos y hace más difícil el monitoreo del proceso.

Paralelamente, el no usar aleatoriedad verificable hace el proceso muy poco transparente. Le entrega la confianza total a quien esté eligiendo los votos o las mesas por auditar. Esto por su parte le quita toda validez a la auditoría, y la convierte en el mismo problema que se intenta solucionar. Es decir, la efectividad de la confirmación dependerá completamente de las personas quienes la lleven a cabo, y el resto de la gente no tendrá cómo verificarlo.

Finalmente, las auditorías definen concretamente pasos a seguir, dependiendo de la evidencia encontrada. No dejan espacio para la interpretación y pueden ser llevados a cabo sin necesidad de tomar decisión alguna. Estas reglas vuelven muy difícil la manipulación o alteración de los resultados, por lo que se puede depositar la confianza en ellos.

Estas razones vuelven a las Auditorías de Confirmación de Resultados una opción difícil de mejorar.

4.3.2. Selección del Tipo de Auditoría

Para llevar a cabo una auditoría primero se debe escoger el tipo. Estos son Ballot-Polling y Batch-Level Comparison. Cada uno tiene sus ventajas y desventajas, las cuales deberán ser sopesadas caso a caso.

En particular, Ballot-Polling tiende a requerir el recuento de menos votos. Esto puede significar una auditoría más rápida y menos costosa, sin embargo la metodología para auditar es más compleja. Se necesita poder elegir votos específicos dentro de una mesa para ser recontados, lo cual toma tiempo y da espacio para errores. Esto se explica con mayor detalle más adelante.

Para Batch-Level Comparison, el proceso es un poco más simple. Cada mesa o *batch* deberá ser recontado completamente, algo no muy diferente a lo que se hace en la actualidad. Esto significa que no se deberá encontrar votos singulares dentro de un conjunto, ahorrando trabajo. Sin embargo, se necesitará auditar más votos con esta metodología, lo que se puede convertir en un costo mayor si no se tiene cuidado.

4.3.3. Selección de Parámetros

Antes de iniciar la auditoría se debe definir algunos parámetros importantes. El primero es el *risk limit*, el cual puede estar reglamentado mediante una ley u otro método similar. Se recomienda un valor no mayor a 10%, idealmente entre 1 y 5%, dependiendo de la importancia de la votación y su tamaño.

Similarmente se puede reglar la hora en la cual se obtendrá el pulso aleatorio. Esta debe

estar lo suficientemente en el futuro como para haber alcanzado a preparar todo el proceso antes de que se emita el pulso. Idealmente unas horas o pocos días luego de haber terminado el conteo inicial.

Finalmente está la cantidad máxima de votos que se desea auditar. Este dependerá fuertemente de los márgenes reportados y de los recursos que se tenga para llevar a cabo la auditoría. Se debe tener en cuenta que, si se alcanza dicha cantidad de votos, se deberá iniciar un conteo manual completo desde cero. En la práctica se querrá que este valor sea el menor posible, considerando la cantidad esperada de votos necesarios, y algún margen para errores o fluctuaciones aleatorias.

4.3.4. Condiciones para Auditar Votos

Auditar votos es la parte central del proceso. Se vuelve a contar algunos votos de la votación, lo que genera evidencia a favor o en contra del resultado reportado. Para esto, el resultado recontado se considera como una verdad absoluta o fundamental, sin espacio para errores. Sin esta condición, no se puede confiar en ningún resultado obtenido a través de la auditoría.

El cumplimiento de esta depende de que el proceso no pueda ser manipulado de ninguna manera, y de tener un nivel de transparencia tal que asegure a los electores que efectivamente no hubo manipulación. Para evitar la manipulación, se utiliza la aleatoriedad verificable. Con esta se determina las muestras que deberán ser recontadas, sin ningún tipo de sesgo o conflicto de interés.

Esto no es suficiente para el caso de Ballot-Polling, pues la muestra solo indica los índices de los votos que deberán ser recontados (Por ejemplo los votos número 24, 63 y 87 de la mesa 1). Para definir estos índices es necesario tener una ordenación de los votos. No importa qué ordenación sea, ya que la probabilidad de elegir cada voto es la misma, pero esta debe estar definida y publicada (para evitar modificaciones posteriores) antes de obtener el pulso aleatorio.

La transparencia se puede ganar mediante el uso de transmisiones en vivo, ya sea por televisión o internet. Se debe poder verificar visualmente la interpretación de cada voto. Esto implica que todos los recuentos deben ser transmitidos en su totalidad, lo cual no debería ser demasiado costoso, ya que son una fracción del total de votos.

4.3.5. ¿Quién Puede Auditar?

El proceso de auditar votos debe ser realizado por alguna entidad con las facultades de recontar todos los votos que sea necesario. Este tipo de cosas actualmente se regula mediante la ley 18.700, sin embargo no pareciera haber una institución con los permisos necesarios en este momento.

Para hacer esto, hay dos candidatos obvios: los Vocales de Mesa y los Tribunales Electorales Regionales. Los primeros serían la solución más rápida, ya que podrían empezar la

auditoría poco tiempo después de haber terminado el conteo inicial. Sin embargo, estas personas podrían estar cansadas luego de haber hecho el trabajo usual de Vocal de Mesa, por lo que podrían ser más propensos a errores durante la auditoría. Por otro lado, si alguno de ellos intentó adulterar los resultados, probablemente intentaría hacerlo de la misma forma durante el proceso de confirmación.

Por otra parte, los Tribunales Electorales Regionales ya hacen recuento de votos en ciertas ocasiones, por lo que hace sentido darles el trabajo a ellos de realizar la auditoría. Además, hay 1 Tribunal por región, excepto en la Región Metropolitana que hay 2, lo que les permite distribuir la carga del recuento y organización. También, al ser pocos Tribunales Electorales, es más fácil hacer la transmisión en vivo del recuento, lo que reduce costos.

4.4. Prototipo

4.4.1. Descripción General

A modo de prototipo, se construyó un servicio web encargado de monitorear y almacenar auditorías [30]. Este toma las configuraciones iniciales y los resultados reportados, y dirige al usuario, indicándole qué se debe auditar. Finalmente produce un resultado, señalando si la votación efectivamente fue confirmada, o si es necesario iniciar un conteo manual completo.

Esto se hace mediante el lenguaje de programación `Python3.6` y el framework `Django` para manejar el servicio web. Además se usa una base de datos `PostgreSQL`, en la cual se registra la información de cada auditoría.

El servicio web fue pensado para ser utilizado por personas que vayan a llevar a cabo Auditorías de Confirmación de Resultados, considerando las restricciones y algoritmos descritos en Capítulos anteriores. Sin embargo, también se ofrece la posibilidad de consultar auditorías pasadas, con lo cual es posible verificar su validez.

Para esto último, se creó un software específicamente para validar los resultados de la auditoría [29]. En este, se entrega los resultados reportados, los parámetros de la auditoría, los recuentos de votos, y el *p-value* final, con lo cual se determina si los resultados de la auditoría efectivamente fueron correctos. Nuevamente, se debe verificar de manera externa los recuentos, usando alguno de los métodos explicados anteriormente.

4.4.2. Arquitectura General

4.4.2.1. Dependencias

El desarrollo de este proyecto se hizo en `Python3.6`, y tiene como dependencias las siguientes librerías:

- `django`: Maneja todo el *backend*, incluyendo modelos, vistas, requests, etc. También renderiza en HTML los templates a modo de *frontend*.

- **django-picklefield**: Permite almacenar objetos serializados en la base de datos, como listas y diccionarios.
- **requests**: Envía requests HTTP, usado para obtener la semilla de aleatoriedad desde la API del Faro de Aleatoriedad.
- **pandas**: Ayuda a manejar archivos csv, simplificando la agregación de datos, consultas, entre otras cosas.
- **psycopg2**: Comunicación con la base de datos Postgres.
- **clcert_chachagen**: PRNG basado en ChaCha20, para generar muestras aleatorias de votos o *batches*.

4.4.2.2. Componentes

Dado que la solución toma forma de un servicio web, está conformada por 3 componentes principales, como se muestra en la Figura 4.1. Estas son frontend, backend y base de datos. Frontend y backend son administrados por **Django**, usando el patrón Model-View-Template (MVT). La base de datos está configurada para ser manejada por Postgres, aunque **Django** tiene la flexibilidad de usar algunas otras, como MariaDB, MySQL, Oracle y SQLite.

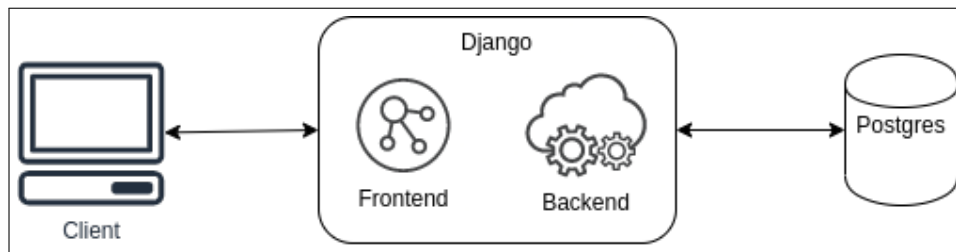


Figura 4.1: Diagrama de Componentes Principales del Servicio Web

La parte del modelo de **Django** corresponde al modelo de datos, por lo que será explicado en la sección siguiente. Las vistas, o *views*, se refieren a la lógica operacional del servicio. Es decir, manejan la forma en la que se procesan las consultas HTTP, calculan resultados, verifican condiciones, entre otras cosas.

Los templates son simplemente formas de generar interfaces gráficas en HTML en base a código. Es posible usar otras tecnologías más avanzadas y que pueden producir visualizaciones más complejas, como **React**, **Angular**, etc. Sin embargo ese no es el enfoque de este proyecto, y se prefirió esta solución solo por ser más rápida de implementar.

El cliente, o usuario, accede al servicio a través de su navegador. Ya que solo se usa los templates de **Django** para generar el frontend, lo que se convierte en HTML solamente, no debería haber ningún tipo de incompatibilidad de navegadores.

4.4.2.3. Django

Django fue dividido en 4 aplicaciones que manejan el funcionamiento de la auditoría: **audit**, **SimpleMajority**, **SuperMajority** y **DHONDT**. La aplicación **audit** se encarga de definir algoritmos generales, los cuales serán ocupados para todas las funciones de elección social. Funciona como una especie de clase abstracta para las otras aplicaciones. También define las formas HTML con las cuales se hará las consultas al servidor.

SimpleMajority, **SuperMajority** y **DHONDT** se ocupan de manejar sus auditorías correspondientes, extendiendo o modificando lo definido en **audit**. Cada una se encarga de saber qué tipo de auditoría se está corriendo, cuál es su estado y qué se debe hacer luego.

4.4.3. Diseño de Base de Datos

4.4.3.1. Relaciones

Django genera automáticamente varias tablas en la base de datos, las cuales no son usadas en el proyecto. En esta sección se hablará solo de los modelos de datos relevantes para el sistema.

En la Figura 4.2 se muestra el diseño general de la base de datos. Este fue generado automáticamente usando la extensión **pydotplus** para **Django** para diagramas UML. Contiene 3 modelos, los cuales se explican a continuación:

- **Audit**: Representa una Auditoría de Confirmación de Resultados. Almacena la información relevante a la auditoría, y algunos campos extra para evitar recalcular lo mismo repetidas veces. Puede tener referencias a múltiples objetos de tipo **SubAudit** y **RecountRegistry**.
- **SubAudit**: Representa una sub-auditoría. Se usa para hacer los cálculos específicos de una auditoría, obteniendo el *p-value* correspondiente.
- **RecountRegistry**: Mantiene el historial de los recuentos de votos de toda la auditoría, incluyendo la hora y fecha de cuando se registró.

4.4.3.2. Audit

El modelo **Audit** contiene 17 campos, descritos a continuación:

- **id**: id generado por la base de datos.
- **accum_recount**: Total de votos recontados para cada candidato o partido de la elección. Se almacena en forma de diccionario.
- **audit_type**: Tipo de auditoría. Estas pueden ser **ballot-polling** o **batch-level comparison**.
- **date**: Fecha y hora en la cual se creó la auditoría.

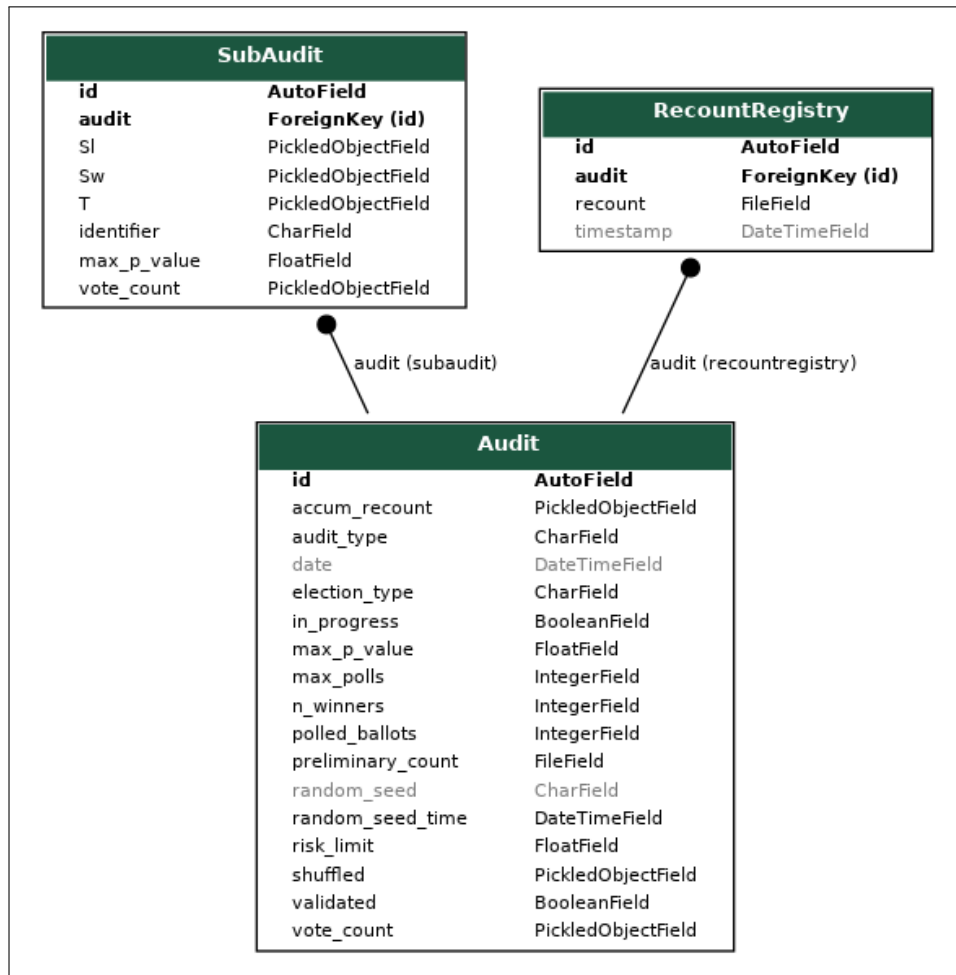


Figura 4.2: Diagrama UML del Modelo de Datos

- **election_type**: Función de elección social que se usó en la votación. Estas pueden ser plurality, supermajority o dhondt.
- **in_progress**: Indica si la auditoría está en progreso o no. Se marca como falso una vez que la elección fue confirmada, o cuando se llega al límite de votos auditados definido.
- **max_p_value**: Valor máximo dentro de todos los *p-values* actuales de la auditoría.
- **max_polls**: Cantidad máxima de votos a auditar.
- **n_winners**: Cantidad de ganadores de la elección.
- **polled_ballots**: Cantidad de votos auditados hasta el momento.
- **preliminary_count**: Archivo csv con el conteo reportado, a nivel de mesa.
- **random_seed**: Semilla de aleatoriedad en formato hexadecimal obtenida del Faro de Aleatoriedad. Tiene valor *null* hasta que sea emitida.
- **random_seed_time**: Hora exacta en la cual será emitida la semilla por el Faro de Aleatoriedad.

- **risk_limit**: Riesgo máximo elegido con el cual se quiere verificar la votación.
- **shuffled**: Muestra aleatoria de votos o mesas a auditar. Se define completamente antes de iniciar el recuento para evitar recalculación algunas cosas.
- **validated**: Indica si la auditoría ha sido validada o no. Solo se marca como verdadero cuando todas las sub-auditorías han sido validadas.
- **vote_count**: Diccionario con la cantidad de votos totales reportados para cada candidato o partido. Se mantiene en la base de datos para evitar recalcularlo en base al archivo csv entregado.

4.4.3.3. RecountRegistry

El modelo RecountRegistry tiene 4 campos:

- **id**: Identificador generado por la base de datos.
- **audit**: Referencia a una instancia del modelo Audit. Representa la auditoría a la cual pertenece el recuento.
- **recount**: Archivo csv que contiene el resultado de un recuento de votos. Este puede estar a nivel de candidato para auditoría de tipo Ballot-Polling, o a nivel de mesa en el caso de Batch-Level Comparison.
- **timestamp**: Fecha y hora exactas en la cual se ingresó el recuento.

4.4.3.4. SubAudit

Este modelo contiene 8 campos:

- **id**: Identificador generado por la base de datos.
- **audit**: Referencia a una instancia del modelo Audit. Representa al auditoría a la cual pertenece esta sub-auditoría.
- **Sl**: Diccionario con factores de modificación para cada perdedor reportado. Definidos para convertir elecciones de tipo D'Hondt en pluralidades.
- **Sw**: Equivalente de Sl, para cada ganador reportado.
- **T**: Contiene el valor acumulado de SPRT para la auditoría. En Batch-Level Comparison es un solo valor, mientras que para Ballot-Polling es un diccionario con un valor para cada *contest* entre ganadores y perdedores reportados.
- **identifier**: Usado para identificar el tipo de sub-auditoría dentro de una auditoría con más de una.
- **max_p_value**: *p-value* máximo actual para la sub-auditoría.
- **vote_count**: Diccionario con el conteo reportado por candidato o partido. Puede ser diferente al de Audit, dependiendo de las transformaciones aplicadas.

4.4.4. Servicio Web

4.4.4.1. Vistas

El servicio se construyó con 6 vistas, las cuales se navegan principalmente de manera lineal, con algunas excepciones. Estas son:

1. **Vista Principal:** Primera vista que aparece al ingresar al sitio. Se lista todas las auditorías creadas anteriormente, con algunos datos como la función de elección social, tipo de auditoría, *p-value*, etc. También se da la opción de ir a crear otra auditoría.
2. **Crear Auditoría:** Se permite la creación de más auditorías, entregando los datos pertinentes, como el *risk limit*, función de elección social, tipo de auditoría, cantidad máxima de votos por auditar y resultados reportados. Además se debe ingresar la hora exacta de la cual se obtendrá el pulso aleatorio del Faro de Aleatoriedad.
3. **Resumen del Reporte:** Se muestra un resumen de los resultados reportados entregados en la vista anterior. Funciona como una vista de espera a que se haya emitido el pulso aleatorio.
4. **Ingreso del Recuento:** Una vez que se haya emitido el pulso aleatorio se permite el ingreso a esta vista. Muestra los votos o mesas específicas que se debe auditar. El recuento se debe ingresar como un archivo csv con el total de votos auditados para cada candidato, en cada mesa.
5. **Resumen de la Auditoría:** Se entrega un resumen de los votos auditados, y el *p-value* actual. Si la votación ha sido confirmada, se informa y se permite volver a la vista principal. Si se ha superado la cantidad máxima de votos, se termina la auditoría y se recomienda iniciar un conteo manual completo. Si la auditoría sigue, se permite volver a la vista anterior para auditar más votos.
6. **Detalle de la Auditoría:** Esta vista contiene toda la información de la auditoría, incluyendo *p-value*, pulso aleatorio, fecha de creación, número de ganadores, entre otras cosas. Además se permite descargar los archivos de resultados reportados y todos los archivos de recuento de votos. Se llega a través de la lista de auditorías en la vista principal.

4.4.4.2. Ejemplo de Uso

Para demostrar el uso de la plataforma, se procederá a confirmar el resultado de una votación de mayoría simple, con 100.000 votos, distribuidos en 1000 mesas. En esta, compiten 2 candidatos, A y B, con 55 % de los votos para A y 45 % para B. Esto se hará con el método de Ballot-Polling a un 5 % de riesgo, y por simplicidad, cada mesa tendrá exactamente 100 votos con preferencias aleatorias, lo cual no afecta a la auditoría. Además se asumirá que no

hubo errores en los resultados reportados.

Se inicia el proceso en la vista principal (Ver Figura 4.3), donde se puede ver la lista de auditorías realizadas anteriormente, con sus estados respectivos y otros datos. Se hace *click* en el botón *New Audit* para ir a crear una nueva auditoría.

| New Audit | | | | | | |
|-----------|----------------|---------------|--------------------------|-------------|-----------|----------------------|
| # | Election Type | Audit Type | Date | In Progress | Validated | View |
| 1 | simplemajority | ballotpolling | July 8, 2020, 10:14 p.m. | False | True | Link |
| 2 | simplemajority | ballotpolling | July 8, 2020, 10:13 p.m. | True | False | Link |
| 3 | simplemajority | ballotpolling | July 8, 2020, 10:12 p.m. | True | False | Link |
| 4 | simplemajority | ballotpolling | July 8, 2020, 10:12 p.m. | False | True | Link |
| 5 | simplemajority | ballotpolling | July 8, 2020, 10:11 p.m. | False | True | Link |

Figura 4.3: Vista Principal

Aparece un formulario para ingresar los datos necesarios para iniciar una nueva auditoría, como se ve en la Figura 4.4. Se selecciona *Simple Majority* como tipo de elección y *Ballot Polling* como tipo de auditoría. En Random Seed Time se ingresa la hora exacta del pulso aleatorio, la cual debería ser en el futuro, sin embargo por simplicidad del ejemplo se define como 2020-07-08 12:00:00 en el pasado.

| | |
|---------------------------------------|--|
| Election Type: | Simple Majority ▼ |
| Audit Type: | Ballot Polling ▼ |
| Random Seed Time: | 2020-07-08 12:00:00 |
| Risk Limit: | 0.05 |
| Number of Winners: | 1 |
| Maximum Pool Count: | 2000 |
| Preliminary Count File: | <input type="button" value="Choose File"/> example.csv |
| <input type="button" value="Submit"/> | |

Figura 4.4: Crear Auditoría

Luego se define el *risk limit* como 0.05, cantidad de ganadores como 1 y número máximo de votos por auditar como 2000. Finalmente se ingresa el archivo csv con los resultados reportados. Esto crea una auditoría en la base de datos, y las otras entradas para llevarla a cabo.

La siguiente vista es el resumen del reporte, en la Figura 4.5. Esta contiene la agregación de votos, agrupados por candidato. Una vez que ya se ha emitido el pulso aleatorio, el botón *Go to Recount* inicia la auditoría. Se calcula la muestra aleatoria completa, y otros factores

dependientes de los resultados reportados.

| Candidate | Reported Ballot Count |
|-----------|-----------------------|
| A | 55000 |
| B | 45000 |

Figura 4.5: Resumen del Reporte

Después se pide auditar ciertos votos o mesas en específico, dependiendo del tipo de auditoría. En este caso son los votos por índice, empezando desde 0, en cada una de las mesas. En la Figura 4.6 se ve parte de la muestra requerida, y la forma en donde se ingresa el recuento (las columnas son Contador, Mesa e Índice de Votos). Particularmente se pide 608 votos en total, lo cual se revisa al enviar el recuento, distribuidos en 453 mesas.

| | | |
|-----|-----|--------|
| 450 | 994 | 3, 5 |
| 451 | 997 | 26, 56 |
| 452 | 998 | 42, 49 |
| 453 | 999 | 23, 54 |

Total: 608

Recount: No file chosen

Figura 4.6: Ingreso del Recuento 1

Para efectos de este ejemplo, se extrajo una muestra aleatoria de 608 votos, sin tener en cuenta la mesa de la cual proviene cada uno. Esto no es diferente estadísticamente a haber auditado correctamente los votos, puesto que se obtienen usando la misma distribución aleatoria que la forma en la que se define la muestra.

Esto lleva al resumen de la auditoría. Contiene el total de votos reportados y auditados por candidato, además de su total, como se muestra en la Figura 4.7. También aparece el riesgo, o *p-value* máximo, actual. Ya que no se ha confirmado la elección, con un *p-value* de $0.38 > 0.05$, se pide volver a la vista anterior.

Nuevamente se pide auditar votos, pero ya que se ha obtenido algo de evidencia en favor de que la elección es correcta, se reduce la cantidad solicitada a 417. También disminuye la cantidad de mesas que se audita a 337 (Figura 4.8).

Esto aún no es suficiente para confirmar la votación, obteniendo un *p-value* de $0.06 > 0.05$. Una ronda más de 54 votos y 51 mesas lo logran (Ver Figura 4.9), llegando a un *p-value*

| Candidate | Reported Ballot Count | Recount |
|--------------|-----------------------|------------|
| A | 55000 | 324 |
| B | 45000 | 284 |
| Total | 100000 | 608 |

Max P-Value: 0.383616543564837

Not Validated Yet

Figura 4.7: Resumen de la Auditoría 1

| | | |
|-----|-----|--------|
| 334 | 984 | 4b |
| 335 | 989 | 95 |
| 336 | 990 | 13 |
| 337 | 993 | 14, 51 |

Total: 417

Recount: No file chosen

Figura 4.8: Ingreso del Recuento 2

final de 0.03. En total, solo se tuvo que auditar 1079 votos, o un 1 % del total.

| Candidate | Reported Ballot Count | Recount |
|--------------|-----------------------|-------------|
| A | 55000 | 584 |
| B | 45000 | 495 |
| Total | 100000 | 1079 |

Max P-Value: 0.0299660634758292

Election Validated!

Figura 4.9: Resumen de la Auditoría 2

En este punto se informa al usuario que la elección ha sido validada. En caso de no haber sido así, y si se llegara a alcanzar el límite de 2000 votos, se cierra la auditoría y se le recomienda al usuario iniciar un conteo manual completo. Finalmente se vuelve a la vista principal, lo que permite ir al detalle de la auditoría, en la Figura 4.10.

En esta vista se muestra toda la información definida al crear la auditoría, además del valor del pulso aleatorio en formato hexadecimal, la cantidad de votos auditados, y el *p-value*

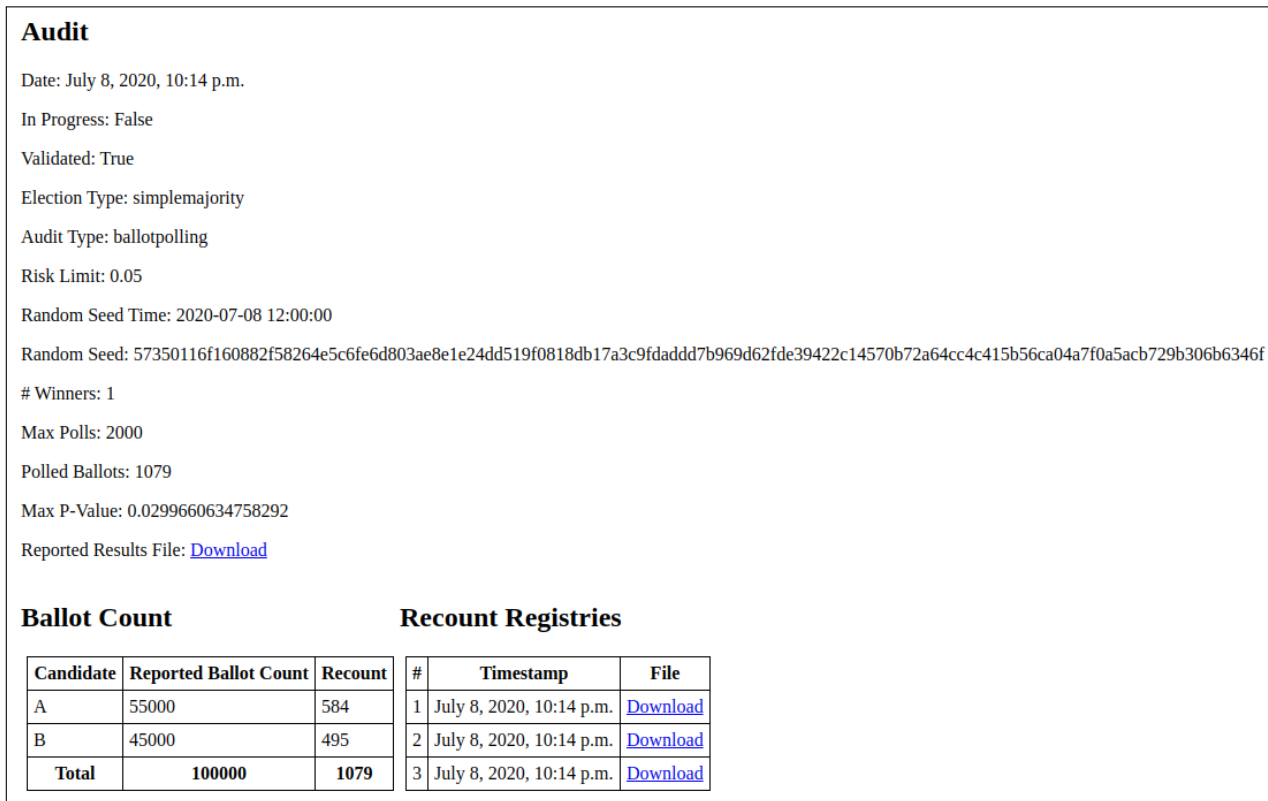


Figura 4.10: Detalle de la Auditoría

alcanzado. También aparece el recuento de votos, agrupado por candidato, y se permite descargar el archivo de resultados reportados, al igual que todos los recuentos.

4.4.5. Limitaciones

El prototipo desarrollado presenta algunas limitaciones, mayoritariamente por simplicidad del código. Estas no deberían tener un impacto muy grande en los resultados obtenidos al usarlo, pero de todas formas se deben tener en cuenta para solucionarlas antes de auditar una elección real.

En primer lugar, las elecciones de tipo mayoría absoluta deberían poder aceptar 1 o 2 ganadores, dependiendo de si alguno de los candidatos alcanzó a obtener más del 50% o no. Esto determinaría si la auditoría es de tipo mayoría absoluta, o pluralidad de 2 ganadores con una verificación adicional para confirmar que efectivamente no se alcanzó la mayoría absoluta.¹ Por simplicidad, se decidió no hacer esto, de forma que si no se alcanzó esta mayoría, manualmente se debe seleccionar una votación de tipo pluralidad con 2 ganadores.

Adicionalmente, un sistema finalizado debería tener la flexibilidad de modificar ligeramen-

¹ Para realizar esta verificación, simplemente se debe hacer una nueva subauditoría, en la cual el candidato con mayor cantidad de votos compite contra la agregación de todos los otros candidatos, simulando ser un solo candidato. El ganador reportado sería este último. Esto verificaría si el candidato en primero lugar obtuvo más votos que todos los otros candidatos juntos (lo que le otorgaría la mayoría absoluta).

te algunos parámetros durante la auditoría, como por ejemplo la cantidad máxima de votos a auditar. Esto podría dar la facilidad de terminar tempranamente el proceso, en caso de que haya mucha evidencia en contra del resultado reportado, o alternativamente, extenderla en caso de que se esté muy cerca del objetivo.

Por último, se producen ligeras inconsistencias cuando aparecen elementos repetidos (con el mismo índice) dentro de una misma muestra aleatoria, ya sean votos o mesas. La interfaz gráfica solicita ingresar múltiples veces el mismo elemento, sin embargo solo hay que hacerlo una vez. Luego, para calcular la certeza, se debería tomar en cuenta la repetición de elementos, sin embargo no se está haciendo, por lo que cada elemento solo aporta 1 vez por muestra. Finalmente, los elementos repetidos se cuentan hacia el total auditado, por lo que el total final puede ser diferente al real. Ya que el tamaño de cada muestra es mucho menor al total de votos, la probabilidad de que esto ocurra es bastante baja, y no genera un impacto significativo.

4.4.6. Extensibilidad a un Producto Terminado

El prototipo desarrollado durante este trabajo de ninguna manera es un producto terminado, y no debería ser usado para conducir una auditoría real. Carece de cosas necesarias para su usabilidad, como una interfaz gráfica intuitiva y bien diseñada, manejo de permisos, un mejor flujo de vistas, detección de errores del usuario, las limitaciones anteriormente mencionadas, entre otras cosas. Sin embargo, los algoritmos desarrollados e implementados definitivamente deberían estar en un producto final.

En particular, aquellos algoritmos usados para seleccionar muestras, agregar votos recontados, y calcular certezas podrían ser reutilizados sin mayores modificaciones. El formato web también podría ser reutilizado, pues entrega propiedades deseables, como la posibilidad de paralelizar los procesos de las distintas zonas, facilidad de acceso y la posibilidad de publicar la información a través del mismo medio.

El diseño de la base de datos también podría ser de utilidad, aunque necesita más trabajo. Almacena todos los campos esenciales para conducir la auditoría, pero se podría mejorar para incluir otras cosas para optimizar la experiencia de usuario, o acelerar algunos cálculos.

4.5. Aplicabilidad a Votación Electrónica

Como se ha visto en este trabajo, es posible conducir Auditorías de Confirmación de Resultados a cualquier votación que mantenga un registro inalterado en papel de la “intención verdadera del elector”. Ayuda que el registro sea a nivel de mesa, sin embargo, no es absolutamente necesario. Un sistema de votación electrónico con estas propiedades podría ser utilizado en vez del sistema actual, y las auditorías se llevarían a cabo exactamente de la misma manera.

La votación electrónica requiere de máquinas dedicadas a registrar votos. La compra de estas máquinas para todo Chile puede ser prohibitivamente costoso, pero afortunadamente

esto no es necesario. Gracias a las propiedades de la votación clásica, es posible tener mesas de votación manejadas con un sistema electrónico al mismo tiempo que se usa la forma en papel. Esto no requerirá ninguna modificación en los métodos de auditoría, confirmando ambas formas en un solo proceso, reduciendo efectivamente el costo de entrada a la votación electrónica y manteniendo intactos todos los mecanismos legales asociados.

Las características necesarias para las máquinas de votación dependen del tipo de auditoría que se quiera hacer. Para el caso de Ballot-Polling, no se necesita nada en particular del sistema electrónico, ya que solo se necesitaría recontar algunos votos de las urnas. Para Batch-Level Comparison sería necesario mantener un registro de a dónde se almacena cada *batch*, y su resultado parcial.

Para el caso de votación por internet, no existe, o no se conoce, una forma efectiva de emitir votos con traza física de manera remota, que sea realmente *voter verifiable*, y que al mismo tiempo mantenga las propiedades necesarias de una votación. Por esto no es posible aplicar las auditorías a este tipo de votación.

Capítulo 5

Validación

5.1. Resolución del Problema

Dados los problemas descritos en la Sección 3.1, se puede ver lo siguiente:

- **Transparencia:** La solución es totalmente transparente, considerando las suposiciones descritas en la Sección 4.4, permitiendo a cualquier persona verificar el proceso completo. Además, todo el código del prototipo y del software de verificación es abierto, y se mantendrá así.
- **Procesos Estándar:** Se introduce algunos procesos estándar para confirmar los resultados de las votaciones, sin embargo aún queda por definir qué sucede exactamente si se encuentra que este resultado es incorrecto. Deberá ser definido en la legislación los pasos a seguir luego de que se encuentre que el recuento manual completo es diferente al resultado reportado.
- **Errores Globales:** El proceso de auditoría intenta encontrar el resultado real en base a la tendencia de todos los votos, por lo que sí se toma en consideración los errores globales. Esto es aún más directo para el caso de Batch-Level Comparison, ya que compara directamente los votos recontados con su contraparte reportada.
- **Cuantificación de Certidumbre:** La base de las Auditorías de Confirmación de Resultados es encontrar la probabilidad de que el resultado reportado sea incorrecto, por lo que claramente define una cuantificación de la certidumbre. Esta suele ser demasiado conservadora, por lo que la probabilidad real puede ser menor a la calculada, pero se puede tomar como una cota superior al riesgo.

De esta forma se encuentra que la solución resuelve en su mayor parte los problemas propuestos. Los problemas restantes deberán ser solucionados junto al Servel, abogados y otras personas con la capacidad de hacer cambios a la legislación. Esto se deberá hacer de tal forma que no comprometa las propiedades del sistema, ni imponga dificultades muy altas.

5.2. Simulación de Casos

5.2.1. Configuración General

Todas las simulaciones descritas a continuación se hicieron con un *risk limit* de 5%, usando ambos Ballot-Polling y Batch-Level Comparison. La cantidad máxima de votos auditados para cada caso se elige de manera que sea cercana a la cantidad esperada, asumiendo que el resultado es correcto y el error de conteo es pequeño.

Además se simuló 3 casos en cada tipo de auditoría, por cada función de elección social, con 1.000 repeticiones, y una semilla de aleatoriedad diferente para cada una. Cada conjunto de 1.000 repeticiones tomó entre 40 minutos y 2 horas, dependiendo del tipo de auditoría y la cantidad de rondas necesitadas, corriendo en Google Colaboratory. Adicionalmente se utilizó GNU Parallel para reducir lo más posible el tiempo de simulación, usando todo el poder de procesamiento de la herramienta.

Los casos se basan en los resultados oficiales reportados por el Servel. Se utilizó estos y no los resultados finales del TRICEL, ya que al momento de correr las simulaciones no se conocía la existencia de estos últimos. Los casos son:

1. **Recuento Perfecto:** Los resultados reales son exactamente los mismos que los reportados. Se utiliza la misma información tanto para los resultados reportados como los recontados.
2. **Recuento Cercano:** Los resultados reales son los mismos a los reportados, pero se reduce el margen entre el ganador con menor cantidad de votos y el perdedor con mayor cantidad de votos.
3. **Recuento Incorrecto:** Los resultados reales son diferentes a los reportados, hay una diferencia entre las asignaciones de cargos o puestos. El margen se modifica intencionalmente para ser muy pequeño, simulando una situación en la cual el resultado podría ser manipulado cambiando muy pocos votos.

Para construir estas variaciones, se usa los resultados reportados como caso base. Luego, se elige cierta fracción p , la cual se extrae de los votos asignados para el ganador con menor cantidad, y se le entregan al perdedor con mayor cantidad de estos. Esto se hace mesa por mesa, lo que reduce el margen general, e incluso puede llegar a cambiar el resultado.

En los datos entregados por el Servel no se enumera las mesas de manera única. Sin embargo, se encontró que con la concatenación de la Circunscripción Electoral, el Local, la Mesa y el Tipo de Mesa se puede identificar cada una de estas de manera inequívoca.

Se reporta la cantidad de votaciones que se logró confirmar, correcta e incorrectamente, usando menos que la cantidad máxima de votos definida. Finalmente, se reporta el número de mesas que se necesitó auditar para los resultados confirmados, junto a la cantidad de votos por mesa para el caso de Ballot-Polling.

5.2.2. Mayoría Simple

Las votaciones para alcaldes se utilizaron para simular la pluralidad de 1 ganador. Esto es, una mayoría simple. En particular, se usó los resultados de la municipalidad de Maipú del año 2016.

En la elección, Catherine Barriga Guerra gana con 35.303 y un margen de 4.688 de un total de 102.399 votos. Esto es un 34.5 % y un 4.6 % respectivamente. Hay 5 otros candidatos perdedores, que suman un total de 62.055 votos, o un 60.6 %, lo cual se detalla en la Tabla 5.1. Los votos restantes son nulos o en blanco.

Tabla 5.1: Resultados Reportados Alcalde Maipú, 2016

| Candidato | Barriga | Vittori | Campusano | Mix | Uribe | Sanhueza |
|-----------|---------|---------|-----------|--------|-------|----------|
| Votos | 35.303 | 30.615 | 17.040 | 11.004 | 2.222 | 1.174 |
| Porción | 34.5 % | 29.9 % | 16.6 % | 10.7 % | 2.2 % | 1.1 % |

Estos están distribuidos en 1.085 mesas de votación, con un promedio de 94 votos por mesa y un máximo de 182. Para las auditorías de tipo Comparison, se asume que cada mesa es 1 voto físico, con un máximo de 182 preferencias marcadas en cada uno.

Para el recuento cercano, se redujo el margen entre Catherine Barriga Guerra y el segundo lugar, Christian Vittori Muñoz. Este quedó en 620 votos, o un 1.2 %. Luego, para el recuento incorrecto, se intercambia las posiciones del primer y el segundo lugar, con un margen de 212 votos, o un 0.2 %.

En la Tabla 5.2 se detalla los resultados de la auditoría. En esta se precisa el tipo de auditoría, el tipo de recuento, la cantidad máxima de votos por auditar M , los votos auditados promedio m , el número de mesas auditadas promedio, el número de rondas promedio, y finalmente la cantidad de confirmaciones logradas. Por la forma que se utilizó para estimar la cantidad de *batches* a auditar, algunas simulaciones recontaron ligeramente más votos que el máximo definido.

Tabla 5.2: Resultados Auditoría Mayoría Simple

| Auditoría | Recuento | M | m | Mesas | Rondas | Confirmados |
|----------------|------------|--------|--------|-------|--------|-------------|
| Ballot-Polling | Perfecto | 5000 | 2295 | 859 | 2.8 | 943 |
| Ballot-Polling | Cercano | 5000 | 4872 | 1047 | 3.6 | 47 |
| Ballot-Polling | Incorrecto | 5000 | 4973 | 1054 | 3.3 | 10 |
| Comparison | Perfecto | 30.000 | 27.431 | 260 | 2.0 | 1000 |
| Comparison | Cercano | 30.000 | 30.884 | 285 | 2.0 | 0 |
| Comparison | Incorrecto | 30.000 | 30.885 | 285 | 2.0 | 0 |

En la Figura 5.1 se muestra histogramas para la cantidad de votos auditados por mesa, en promedio para todas las simulaciones, en escala logarítmica.

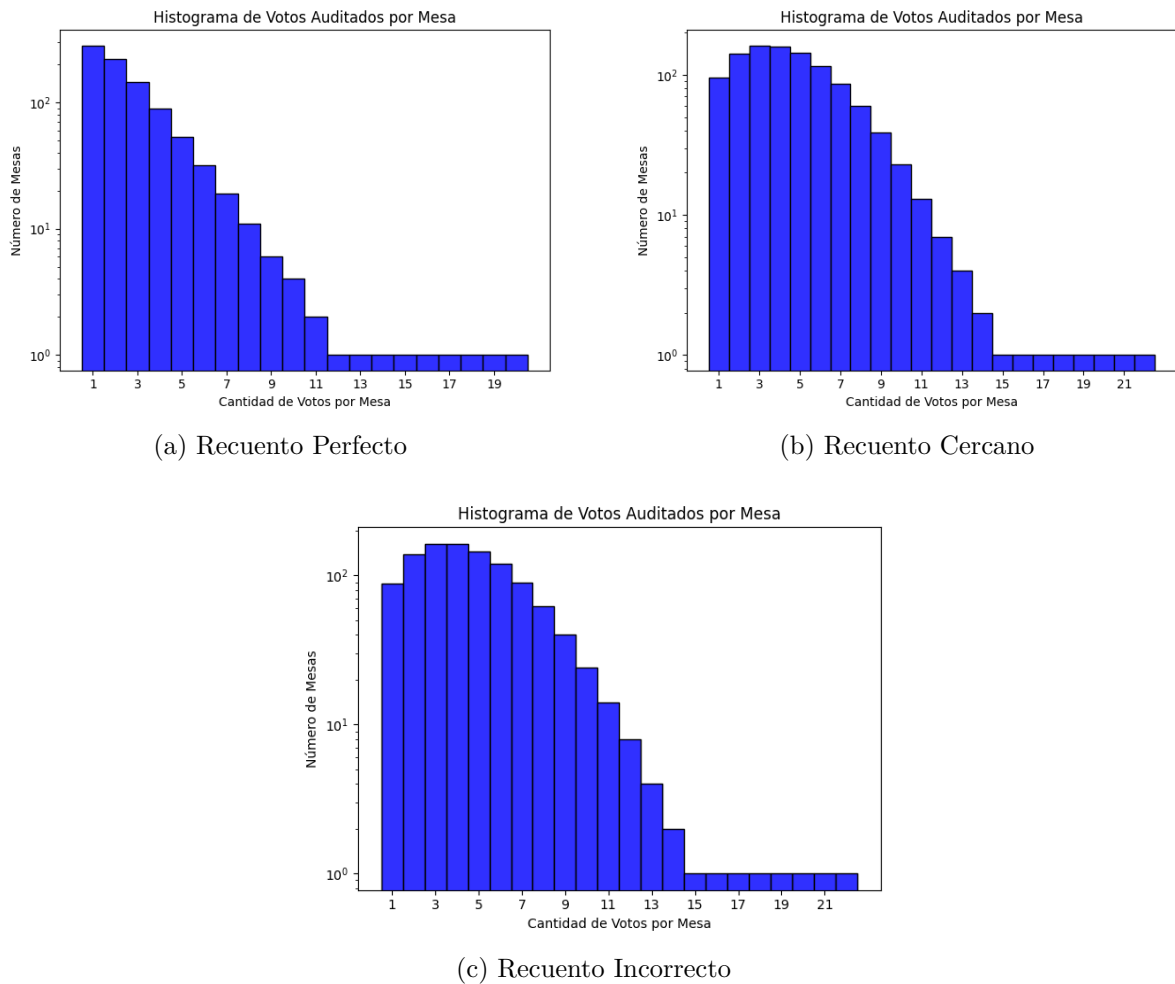


Figura 5.1: Histograma de Votos por Mesa para Mayoría Simple

5.2.3. Pluralidad

Se decidió además verificar las votaciones primarias presidenciales del 2017 como caso de estudio para una pluralidad de 2 ganadores. En esta se tiene 6.680.021 votos, distribuidos en 42.868 mesas de votación. En promedio hay 156 votos por mesa, con un máximo de 359. Sus ganadores reportados son, en primer lugar, Sebastián Piñera Echenique con 2.411.317 votos, y en segundo lugar, Alejandro Guiller Álvarez con 1.491.473 votos, dentro de 6 candidatos en total.

El tercer lugar obtuvo 1.332.450 votos, lo cual da un margen de 159.023, o un 2.4%. Los resultados reportados están especificados en la Tabla 5.3. Por el tamaño de la tabla, se decidió agrupar los 3 candidatos con menor cantidad de votos (Enriquez-Ominami, Artes y Navarro) en la columna Otros.

Las modificaciones para el recuento cercano resultan en un margen de 51.603 votos (0.8%). Para el recuento incorrecto se intercambia el segundo y tercer lugar (Beatriz Sánchez Muñoz), y se llega a un margen de 493 votos, o un 0.007%.

Tabla 5.3: Resultados Reportados Presidenciales Primarias, 2017

| Candidato | Piñera | Guillier | Sánchez | Kast | Goic | Otros |
|-----------|-----------|-----------|-----------|---------|---------|---------|
| Votos | 2.411.317 | 1.491.473 | 1.332.450 | 522.145 | 386.398 | 433.434 |
| Porción | 36.1 % | 22.3 % | 19.9 % | 7.8 % | 5.8 % | 6.5 % |

En la Tabla 5.4 están detallados los resultados de la auditoría, de la misma forma que en la sección anterior. En la Figura 5.2 se muestra el histograma de votos recontados por mesa, promediado por las 1000 simulaciones de cada tipo, en escala logarítmica.

Tabla 5.4: Resultados Auditoría Pluralidad

| Auditoría | Recuento | M | m | Mesas | Rondas | Confirmados |
|----------------|------------|---------|---------|-------|--------|-------------|
| Ballot-Polling | Perfecto | 5000 | 3815 | 3625 | 3.2 | 635 |
| Ballot-Polling | Cercano | 5000 | 4859 | 4577 | 3.0 | 83 |
| Ballot-Polling | Incorrecto | 5000 | 5000 | 4705 | 2.0 | 0 |
| Comparison | Perfecto | 100.000 | 100.500 | 610 | 1.0 | 1000 |
| Comparison | Cercano | 100.000 | 99.293 | 603 | 1.1 | 0 |
| Comparison | Incorrecto | 100.000 | 99.300 | 603 | 1.1 | 0 |

5.2.4. Mayoría Absoluta

Para mayoría absoluta se consideró el caso de la segunda vuelta para las votaciones presidenciales chilenas de 2017. En esta solo competían 2 candidatos: Sebastián Piñera y Alejandro Guillier, con un total de 7.011.558 votos. Sebastián Piñera fue asignado como ganador. Los resultados se especifican en la Tabla 5.5

Tabla 5.5: Resultados Reportados Presidenciales Secundarias, 2017

| Candidato | Sebastián Piñera | Alejandro Guillier |
|-----------|------------------|--------------------|
| Votos | 3.788.641 | 3.147.868 |
| Porción | 54.0 % | 44.9 % |

El margen fue de 640.773 votos, o un 9.1 %. Se tenía un total de 42.868 mesas de votación, con un promedio de 164 votos en cada una, y un máximo de 382. Luego para el recuento cercano se disminuyó a un margen de 147.131 (2.1 %), mientras que para el incorrecto se intercambió sus lugares con un margen de 12.847 votos (0.2 %).

En la Tabla 5.6 se muestra el detalle de los resultados para las auditorías. También, los histogramas de votos recontados por mesa están en la Figura 5.3, nuevamente en escala logarítmica.

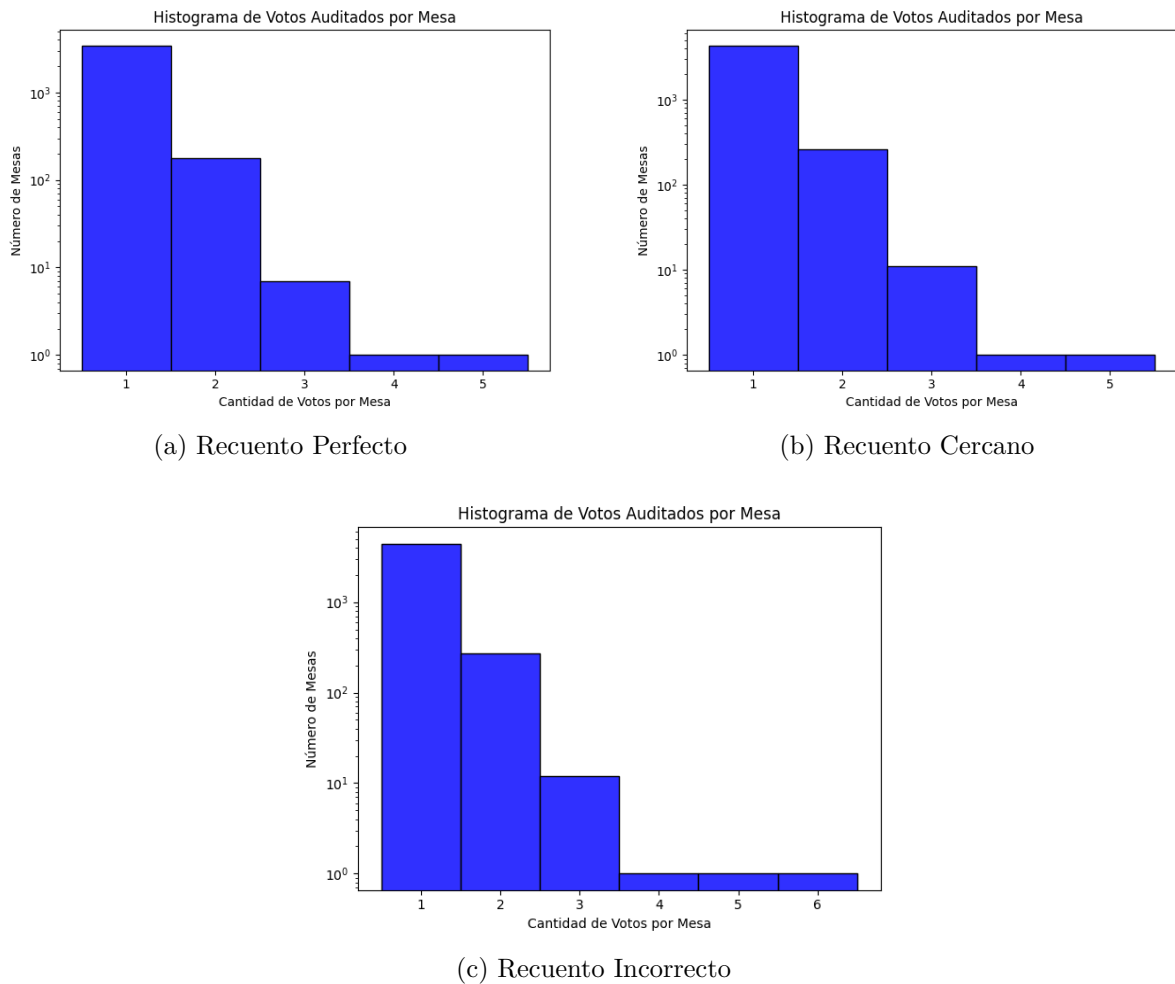


Figura 5.2: Histograma de Votos por Mesa para Pluralidad

5.2.5. D'Hondt

Las elecciones tipo D'Hondt para diputados se hacen regionalmente. Como caso de estudio se tomó los resultados reportados de la primera región para las elecciones del 2017. En estas se asigna 2 puestos usando el método proporcional.

En esta participaron 7 listas, con 22 miembros en total. En la Tabla 5.7 se especifica los resultados reportados por lista. Más específicamente, los miembros de cada lista fueron:

- **Lista B:** Carlos Eduardo Santelices Lagos y Myla Etelvina Chavez Fajardo.
- **Lista G:** Sebastián Vergara Alfaro, Miguel Alfredo Rocha Sarmiento, Rodrigo Oliva Vicentelo y Roxana Vigueras Cherras.
- **Lista H:** Rafael Montes González, Hector Ticuna Vilca, Carolina Vásquez Espinoza y Manuel Sánchez Loyola.
- **Lista N:** Hugo Gutiérrez Galvez, Patricio Martínez Fuentes, Danisa Astudillo Peiretti y Juan Marroquín Guzmán.

Tabla 5.6: Resultados Auditoría Mayoría Absoluta

| Auditoría | Recuento | M | m | Mesas | Rondas | Confirmados |
|----------------|------------|--------|--------|-------|--------|-------------|
| Ballot-Polling | Perfecto | 2000 | 1168 | 1150 | 2.2 | 880 |
| Ballot-Polling | Cercano | 2000 | 1991 | 1943 | 2.4 | 8 |
| Ballot-Polling | Incorrecto | 2000 | 1994 | 1945 | 2.4 | 6 |
| Comparison | Perfecto | 30.000 | 30.000 | 190 | 1.0 | 1000 |
| Comparison | Cercano | 30.000 | 30.011 | 173 | 1.0 | 0 |
| Comparison | Incorrecto | 30.000 | 30.020 | 173 | 1.0 | 0 |

- **Lista O:** Mariela Basualto Avalos, Ana María Luksic Romero y Pablo Ornaldo Valenzuela Huanca.
- **Lista P:** Renzo Trisotti Martínez, Gabriela Monsalvez Rojas, Natan Olivos Nuñez y Ramón Galleguillos Castillo.
- **Lista R:** Freddy Marcos Araneda Barahona.

Tabla 5.7: Resultados Reportados Diputado, Primera Región, 2017

| Lista | P | N | G | O | R | H | B |
|---------|--------|--------|--------|-------|-------|-------|-------|
| Votos | 38.960 | 27.770 | 11.652 | 4.330 | 3.746 | 2.504 | 1.444 |
| Porción | 40.5 % | 28.9 % | 12.1 % | 4.5 % | 3.9 % | 2.6 % | 1.5 % |

Se reportó un total de 96.224 votos, con Renzo Trisotti Martínez (partido Unión Demócrata Independiente, lista P) y Hugo Gutiérrez Gálvez (partido Comunista de Chile, lista N) en primer y segundo lugar, respectivamente. En tercer lugar estuvo Ramón Galleguillos Castillo (partido Unión Demócrata Independiente, lista P), con un margen de 8.290 votos, o un 8.6 %. Eran 719 mesas de votación, con 134 votos en promedio y máximo 251.

Para las variaciones hay que tener en cuenta el funcionamiento de las elecciones proporcionales. En particular, se quitó votos para Hugo Gutiérrez Gálvez y se los dio a Ramón Galleguillos Castillo. Con esto se redujo el margen a 4.425 (4.6 %), para el recuento cercano, y se invirtió la relación para el recuento incorrecto, llegando a un margen de 2.192 (2.3 %).

Los resultados de las auditorías aparecen en la Tabla 5.8, utilizando el mismo formato de las secciones anteriores. Los histogramas de votos recontados por mesa están en la Figura 5.4, en escala logarítmica.

5.3. Optimización para Batch-Level Comparison

En los resultados obtenidos anteriormente, se ve que la cantidad de votos necesarios para Batch-Level Comparison es significativamente mayor a los necesarios para Ballot-Polling. Por una recomendación de Vanessa Teague, autora importante en el área de *Risk Limiting Audits*, se dividió las mesas de votación en subconjuntos de no más de 50 votos. En la práctica esto

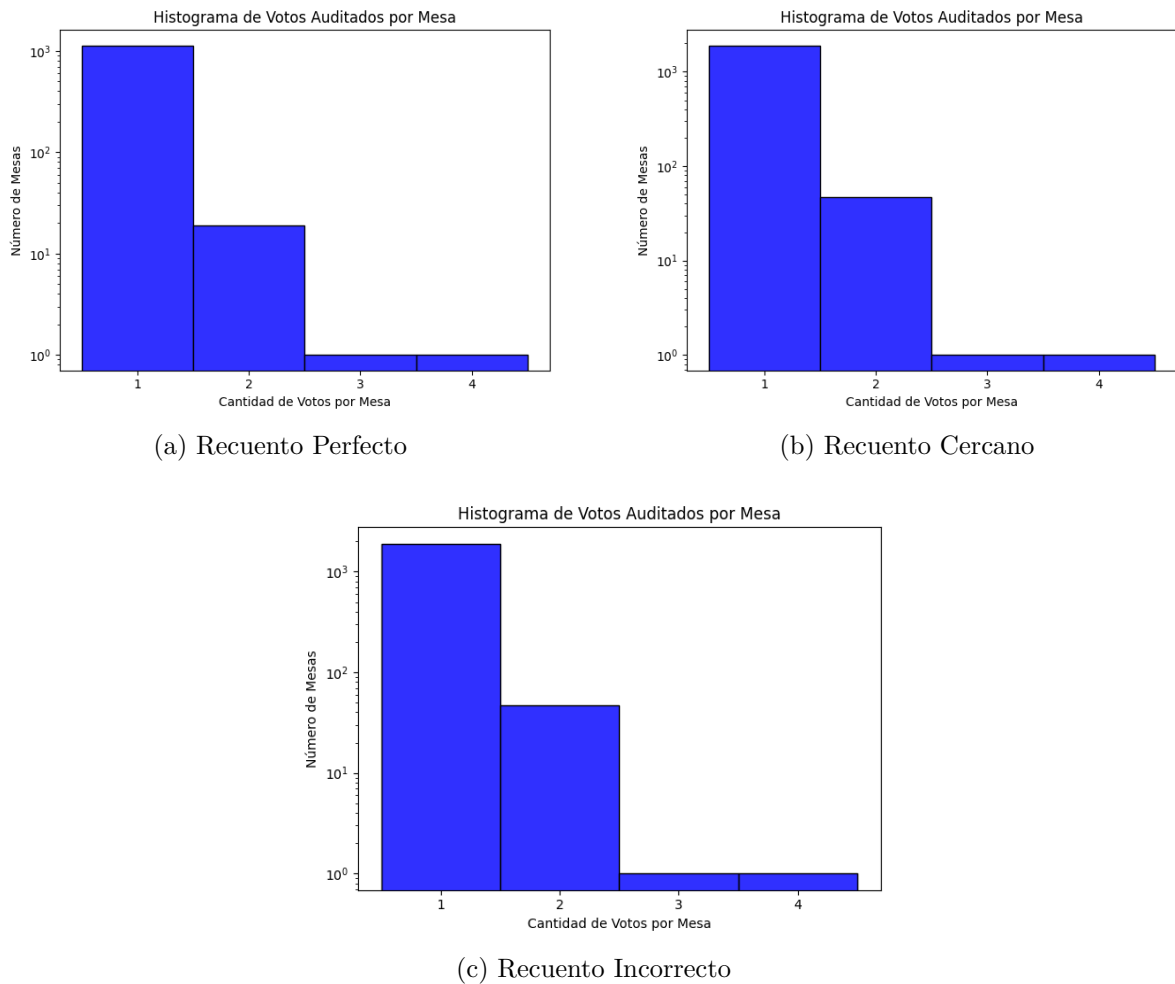


Figura 5.3: Histograma de Votos por Mesa para Mayoría Absoluta

se puede hacer agregando más mesas, lo cual puede ser caro, o separando los votos de cada mesa justo antes de empezar el conteo. Esto último no debería generar mayores problemas, y se recomienda fuertemente hacerlo.

Como ejemplo se toma el caso de la pluralidad para las elecciones presidenciales primarias del 2017 utilizado en la Sección 5.2.3. Originalmente esta contaba con casi 43.000 *batches* de 156 votos en promedio, y un máximo de 359. Necesita de al rededor de 100.000 votos auditados para lograr la confirmación. Luego de dividir las mesas, se obtuvo 155.000 *batches* con 43 votos en promedio. Esto disminuyó la cantidad de votos necesarios a tan solo 14.000.

De la misma forma se analiza las elecciones presidenciales secundarias del 2017 vistas en la Sección 5.2.4, la cual contaba de la misma cantidad de mesas de votación, con un promedio de 164 votos y un máximo de 382. Batch-Level Comparison necesita de 30.000 votos para poder confirmar esta votación, sin embargo, dividiendo los *batches* en subconjuntos de 43 votos en promedio, se redujo esta cantidad a 4.000 votos.

Esto muestra cómo se puede hacer una mejora significativa a la auditoría con un método

Tabla 5.8: Resultados Auditoría D'Hondt

| Auditoría | Recuento | M | m | Mesas | Rondas | Confirmados |
|----------------|------------|--------|--------|-------|--------|-------------|
| Ballot-Polling | Perfecto | 10.000 | 3218 | 664 | 4.8 | 978 |
| Ballot-Polling | Cercano | 10.000 | 9733 | 719 | 6.8 | 46 |
| Ballot-Polling | Incorrecto | 10.000 | 10.000 | 720 | 3.3 | 0 |
| Comparison | Perfecto | 30.000 | 24.501 | 159 | 6.1 | 1000 |
| Comparison | Cercano | 30.000 | 27.976 | 181 | 3.0 | 0 |
| Comparison | Incorrecto | 30.000 | 27.736 | 181 | 2.0 | 0 |

bastante simple. A expensas de un poco más de organización en el conteo y almacenamiento de votos, se reduce la cantidad de votos necesitados en 7 veces, acelerando de gran manera la auditoría.

Reduciendo el tamaño de estos *batches* aún más, conllevaría a incluso mejores resultados, necesitando progresivamente menos votos. En algún punto, se tendría la misma eficiencia esperada que para Ballot-Polling, pudiendo superarla con *batches* suficientemente pequeños. Esto es así para conjuntos de tamaño 1, lo cual se convierte en Ballot-Level Comparison, sin embargo sería demasiado caro implementarlo de esta manera.

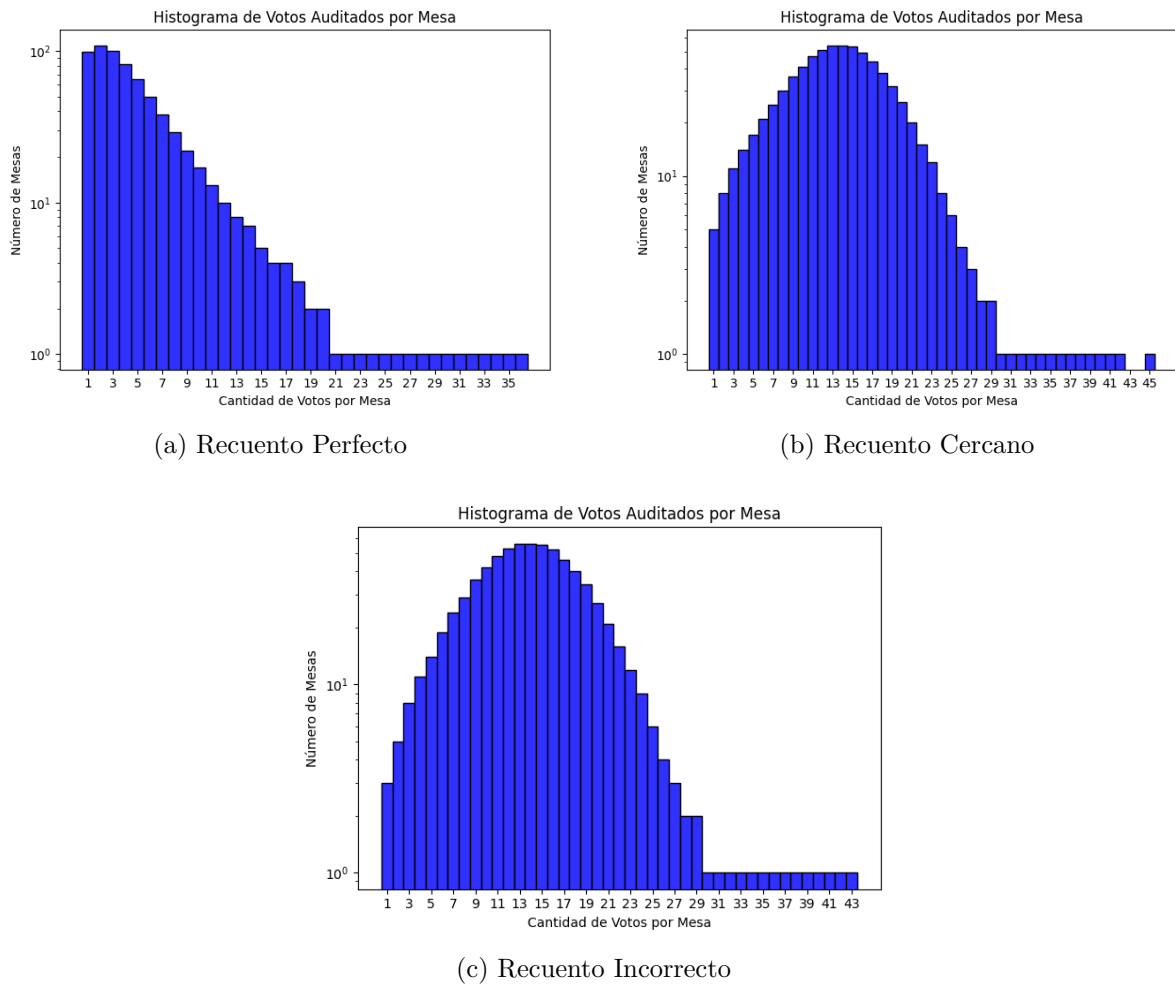


Figura 5.4: Histograma de Votos por Mesa para D'Hondt

Capítulo 6

Conclusiones

6.1. Resumen

En este trabajo de título se ha analizado diferentes formas de *Risk Limiting Audits*, y se ha escogido las más apropiadas para el desarrollo de un prototipo que logra aplicarlas a las votaciones chilenas actuales.

Se implementó este prototipo en Python, usando el framework Django, lo que facilita una interfaz web. Esto permite conducir una auditoría por internet, para luego publicar sus resultados, habilitando a cualquier persona verificarlos. Se hace uso del Faro de Aleatoriedad de la Universidad de Chile para dar aún más transparencia al proceso, y evitar la manipulación de auditorías.

Simulaciones de gran escala se realizan utilizando datos reales de votaciones pasadas. El sistema logra confirmar las votaciones que no han sido alteradas con una gran probabilidad. Sin embargo, suele tener problemas cuando se reduce los márgenes, incluso si el resultado final no ha sido cambiado. Además, no parece ser muy útil para votaciones relativamente pequeñas, como las para alcaldes o diputados, ya que se necesita auditar un porcentaje significativo de los votos para poder hacer la confirmación.

Se analiza la aplicabilidad a votación electrónica, y se concluye que requiere muy poco esfuerzo adicional, dado un sistema con las propiedades necesarias. Además, ayuda a esparcir el costo en el tiempo, permitiendo la adopción gradual del sistema electrónico.

6.2. Revisión de Objetivos

Este trabajo logra cumplir con casi todos los objetivos propuestos en la Sección 1.2. En particular, los objetivos 1 y 2 se llevan a cabo en las Secciones 4.1, en donde se analiza los distintos métodos y se escoge BRAVO y MICRO. Luego, los objetivos 4 y 5 se desarrollan en la Sección 4.2, en la cual se define cómo ocupar el Faro de Aleatoriedad para escoger muestras aleatorias y se explica el funcionamiento del Sequential Probability Ratio Test para llevar a cabo una Auditoría de Confirmación de Resultados.

En la Sección 4.4 se muestra la construcción de un prototipo hecho para conducir una auditoría (objetivo 3), el cual logra tener propiedades de transparencia, permitiéndole a cualquier persona externa verificar la correcta ejecución de la auditoría (objetivo 6).

El objetivo 7 se cumplió parcialmente: se logró validar el funcionamiento del sistema desarrollado usando simulaciones de gran escala con datos reales para todos los tipos de votaciones, y usando ambas formas de auditoría. Sin embargo, no fue posible conducir una auditoría sobre una votación real dentro de la FCFM, por la situación de cuarentena en Santiago.

Finalmente, el objetivo 8 se discute en la Sección 4.5, en la cual se concluye cómo se podría aplicar el sistema desarrollado a votación electrónica, y qué beneficios le otorga. Una vez que ya se tiene instaurado un sistema de Auditorías de Confirmación de Resultados, el costo de entrada a la votación electrónica es mínimo, con la posibilidad de hacer una transición suave a esta.

6.3. Análisis de Resultados

A continuación se analizará los resultados obtenidos en el Capítulo 5, correspondientes a las simulaciones realizadas utilizando resultados de votaciones reales y variaciones de estos.

6.3.1. Votos Auditados

Primero, en términos generales, se puede notar que las auditorías de tipo Batch-Level Comparison toman significativamente más votos auditados que las Ballot-Polling. En ciertas ocasiones hasta 50 veces más, lo que las puede volver excesivamente costosas de hacer y disminuir el interés en ellas.

Esto sucede ya que este tipo de auditoría depende fuertemente del tamaño del *batch*, o mesa en este caso. En promedio se tiene que las mesas en Chile almacenan al rededor de 150 votos cada una, con máximos de hasta casi 400. Esto excede por mucho el tamaño recomendado de 50 votos que se usa en Estados Unidos.

Luego de hacer esta optimización, dividiendo cada *batch* en conjuntos de tamaño relativamente uniforme, con un máximo de 50 votos, se logró una mejora muy significativa. Para el caso particular que se probó, se redujo la cantidad de votos auditados de 100.000 a tan solo 14.000, en el primer ejemplo, lo cual es menos de un séptimo. De igual manera sucedió en el segundo ejemplo, bajando de 30.000 a tan solo 4.000 votos.

Esta mejora en eficiencia seguiría aumentando, mientras más se divida los conjuntos de votos. Llegar a *batches* de tamaño 1 parece ser demasiado caro, y significaría costos significativos de administración, por lo que probablemente no se llegue a ese extremo. Sin embargo, dados los beneficios obtenidos por un esta optimización, se espera que haya un fuerte interés en reducir sus tamaños lo más posible.

Con respecto a la cantidad de votos por mesa, se reportó histogramas con la distribución promedio de estos, para cada una de las funciones de elección social, y cada caso simulado. Se encontró que para un mismo tipo de elección, los recuentos cercano e incorrecto producen distribuciones parecidas, con un sesgo hacia cantidades más altas que el recuento perfecto. Esto se debe a que esos casos necesitaron auditar más votos, por lo tanto en promedio se necesitó hacer más recuentos por mesa.

6.3.2. Confirmaciones

Por otro lado, las auditorías de tipo Ballot-Polling parecen ser menos sensibles a errores en los resultados reportados. Para los recuentos cercanos, estas lograron confirmar los resultados de algunas votaciones, aunque en ningún caso fue más de un 5 %, mientras que Batch-Level Comparison no logró verificar ninguna.

Esto también significa que Ballot-Polling puede ser más propenso a confirmar erróneamente votaciones, cuyos resultados son incorrectos. Esto no sucedió más del 1 % de las veces, sin embargo Batch-Level Comparison no cometió esos errores. No es claro si esto se debe a que efectivamente Batch-Level Comparison produce menos errores, o si es un artefacto de la metodología para generar las nuevas distribuciones de votos.

En lo que concierne al recuento perfecto en Ballot-Polling, es posible estimar la probabilidad de confirmar una votación usando a lo más la cantidad límite de votos definida. Para la mayoría simple, se logró verificar 943 votaciones. Esto significa que es posible confirmar una votación de características parecidas, a un 5 % de riesgo, usando a lo más 5000 votos, con casi un 95 % de probabilidad.

De la misma manera, para la pluralidad, con casi un 65 % de probabilidad se necesitará 5000 votos o menos para realizar la confirmación. Haciendo el mismo análisis para el caso de mayoría absoluta y D'Hondt, se encuentra que las probabilidades son de casi un 90 % con un máximo de 2000 votos, y 98 % para 10.000 votos, respectivamente.

Finalmente, las confirmaciones incorrectas de votaciones fueron significativamente menores a las estimadas con el *risk limit* (1 % contra 5 %), incluso teniendo resultados muy cercanos a los reales. Esto se puede deber a que los métodos de auditoría toman suposiciones demasiado conservadoras, lo que resulta en una estimación de *p-values* subóptimo. Mejorar esto, como lo intenta hacer SHANGRLA, podría llevar a una auditoría más corta y con menos votos recontados.

6.4. Trabajo Futuro

Si bien el proyecto realizado logra cumplir su cometido de confirmar votaciones, claramente necesita mejorar en el ámbito de usabilidad y diseño. Las personas encargadas de usar este sistema no necesariamente conocerán los fundamentos matemáticos del proceso, sin em-

bargo deberán poder llevarlo a cabo sin mayores problemas. Para esto es esencial una buena interfaz gráfica, intuitiva y fácil de usar.

Esto va de la mano con mejoras al registro de votos auditados. En una situación real, probablemente se querrá almacenar imágenes de cada voto, u otros formatos parecidos. De esta forma habría una mayor trazabilidad del proceso, aunque podría aumentar los tiempos de auditoría.

Además es necesario incluir algunas restricciones y opciones no disponibles actualmente, como verificar que no se obtuvo una mayoría absoluta y no asumir que se está en una pluralidad. Esto no debería cambiar mucho los resultados con respecto a los obtenidos.

Otro punto no considerado es la repetición de votos o *batches* en la muestra aleatoria. Si uno de estos aparece más de una vez en la misma muestra, se debería contar esa misma cantidad de veces en el cálculo de la certeza. Sin embargo, por simplicidad del código, esto solo se hace una vez, lo cual tiene un efecto bastante menor en los resultados, pero de todos modos debería ser corregido.

Asimismo, elementos de las muestras aleatorias pasadas no deberían ser solicitados nuevamente, sino que almacenados e integrados en el recuento automáticamente. Esto simplificaría el proceso y reduciría la posibilidad de errores por parte del usuario.

Sobre mejoras en de los algoritmos, SHANGRLA propone ideas muy interesantes y promete necesitar significativamente menos votos que los otros esquemas de RLA. El problema de la divergencia impidió que esto se pudiese explorar a fondo, por lo que una solución a este permitiría la incorporación del algoritmo al sistema. Con esto se podría mejorar la eficiencia de las auditorías, lo que significaría aún menos trabajo para sus encargados, haciéndolas más atractivas de utilizar.

Finalmente, queda por clarificar y resolver todos los problemas políticos y legales asociados a introducir las Auditorías de Confirmación de Resultados como el método oficial de verificación de votaciones. Esto es posiblemente uno de los trabajos más difíciles y lentos de realizar, pero es un paso absolutamente necesario, sin el cual el proyecto no tiene mucho sentido.

Bibliografía

- [1] Jorge Bañuelos and Philip B. Stark. Limiting risk by turning manifest phantoms into evil zombies. *arXiv: Applications*, 2012.
- [2] Josh Benaloh, Douglas W. Jones, Eric Lazarus, Mark Lindeman, and Philip B. Stark. Soba: Secrecy-preserving observable ballot-level audit. *ArXiv*, abs/1105.5803, 2011.
- [3] Josh Benaloh, Matthew Bernhard, J. Alex Halderman, Ronald L. Rivest, Peter Y. A. Ryan, Philip B. Stark, Vanessa Teague, Poorvi L. Vora, and Dan S. Wallach. Public evidence from secret ballots. *ArXiv*, abs/1707.08619, 2017.
- [4] CLCERT. Random uchile. <https://beacon.clcert.cl>, 2018.
- [5] CLCERT. Beacon verifier. <https://github.com/clcert/beacon-verifier>, 2018.
- [6] Franco Aníbal Pino Córdova. Elección de vocales de mesa con aleatoriedad verificable. *Universidad de Chile*, 2019. Memoria de Ingeniería Civil en Computación.
- [7] Gobierno de Chile. Sobre acceso a la información pública. <https://www.leychile.cl/Navegar?idNorma=276363>.
- [8] Gobierno de Chile. Ley orgánica constitucional sobre votaciones populares y escrutinios. <https://www.leychile.cl/Navegar?idNorma=30082&idParte=0>, 1988.
- [9] Biblioteca del Congreso Nacional de Chile. Elecciones municipales. <https://www.bcn.cl/leyfacil/recurso/elecciones-municipales>, 2012.
- [10] Biblioteca del Congreso Nacional de Chile. Elecciones parlamentarias. <https://www.bcn.cl/leyfacil/recurso/elecciones-parlamentarias>, 2012.
- [11] Biblioteca del Congreso Nacional de Chile. Elecciones presidenciales. <https://www.bcn.cl/leyfacil/recurso/elecciones-presidenciales>, 2012.
- [12] William Feller. An introduction to probability theory and its applications. *Wiley*, II, 1971.
- [13] Camilo Gómez. Chacha20-generator-utilities - python. <https://github.com/clcert/ChaCha20-Generator-Utilities>.
- [14] Mark Lindeman and Philip B. Stark. A gentle introduction to risk-limiting audits. *IEEE Security & Privacy*, 10:42–49, 2012. IEEE Press.
- [15] Mark Lindeman, Philip B. Stark, and Vincent S. Yates. Bravo: Ballot-polling risk-limiting audits to verify outcomes. In *EVT/WOTE*, 2012.
- [16] State of Colorado. Coloradorla. <https://github.com/FreeAndFair/ColoradoRLA/>, 2017.

- [17] State of Colorado. Colorado audit center. <https://www.sos.state.co.us/pubs/elections/auditCenter.html>, 2017.
- [18] Kellie Ottoboni, Matthew Bernhard, J. Alex Halderman, Ronald L. Rivest, and Philip B. Stark. Bernoulli ballot polling: A manifest improvement for risk-limiting audits. <https://arxiv.org/abs/1812.06361>, 2018.
- [19] Ronald L. Rivest. Clipaudit: A simple risk-limiting post-election audit. *ArXiv*, abs/1701.08312, 2017.
- [20] Carsten Schürmann. A risk-limiting audit in denmark: A pilot. In *E-VOTE-ID*, 2016.
- [21] SERVEL. Resultados definitivos elecciones presidencial, parlamentarias y de cores. <https://www.servel.cl/resultados-en-excel-por-mesa-a-partir-del-ano-2012/>.
- [22] Philip B. Stark. A sharper discrepancy measure for post-election audits. 2008.
- [23] Philip B. Stark. Conservative statistical post-election audits. *The Annals of Applied Statistics*, 2:550–581, 2008. Institute of Mathematical Statistics.
- [24] Philip B. Stark. Sets of half-average nulls generate risk-limiting audits: Shangrila. *ArXiv*, abs/1911.10035, 2019.
- [25] Philip B. Stark and Vanessa Teague. Sets of half-average nulls generate risk-limiting audits (shangrila). <https://github.com/pbstark/SHANGRLA>.
- [26] Philip B. Stark and David A. Wagner. Evidence-based elections. *IEEE Security Privacy*, 10:33–41, 2012. IEEE Press.
- [27] Philip B. Stark, Vanessa Teague, and Aleksander Essex. Verifiable european elections: Risk-limiting audits for d’hondt and its relatives. *JETS: USENIX Journal of Election Technology and Systems*, 3.1:18–39, 2014.
- [28] TRICEL. Resultados de elecciones. <https://tribunalcalificador.cl/resultados-de-elecciones/>.
- [29] Diego Vargas. Results validator for rla web service. <https://github.com/diegovargasj/rla-verifier>, 2020.
- [30] Diego Vargas. Web service rla implementation. <https://github.com/diegovargasj/RLA>, 2020.
- [31] Abraham Wald. Sequential tests of statistical hypotheses. *IEEE Security and Privacy*, 1945. IEEE Press.