Universidad del Valle
Facultad de Ingeniería
Escuela de Ingeniería de Sistemas y Computación

Minimizando el Extremismo Presente en una Población (MinExt)

Anderson Johan Alban Angulo - 202310006
Andrés Felipe Asprilla Urrutia - 202224101
Andrés Mauricio Ortiz Bermúdez - 202110330
Carlos Mauricio Tovar Parra - 201741699

Profesor

Jesús Alexander Aranda Bueno Ph.D.

Curso

Análisis y Diseño de Algoritmos II (750020C)

26 de julio de 2025

Índice

1. Introducción

El fenómeno del extremismo ha cobrado especial relevancia en las sociedades actuales, caracterizándose por una polarización creciente de las opiniones en grupos claramente diferenciados. Este proyecto aborda el problema de la minimización del extremismo (MinExt), el cual busca reducir el nivel total de extremismo dentro de una población, tomando como base un conjunto inicial de opiniones diversas. Para esto, se plantea la implementación de un modelo de optimización que permita decidir cuáles personas cambiarán sus opiniones iniciales, con el objetivo de alcanzar la menor cantidad posible de extremismo, considerando costos limitados en términos económicos y logísticos.

El presente trabajo se desarrolla mediante técnicas avanzadas de programación entera mixta y optimización combinatoria, utilizando el método Branch and Bound y herramientas de modelamiento como MiniZinc. El proyecto también incluye el diseño de una interfaz gráfica que permite a un usuario final interactuar de manera intuitiva con el modelo desarrollado.

2. El modelo: Descripción y Justificación de su Adecuación

El modelo implementado para resolver el problema `MinExt` se fundamenta en los siguientes componentes:

2.1. Parámetros de Entrada

- n: Número total de personas en la población.
- m: Número de posibles opiniones.
- p_i: Número inicial de personas con opinión i.
- ext_i: Valor de extremismo de la opinión i.
- $c_{i,j}$: Costo de mover personas de opinión i a opinión j.
- c_i^e: Costo extra por mover personas hacia opiniones inicialmente desocupadas.
- c_t: Costo total máximo permitido.
- máx M: Número máximo permitido de movimientos totales.

2.2. Variables de Decisión

Variable $x_{i,j}$

Definición Representa el número de personas que cambiaron de la opinión i a la opinión j.

Dominio $i, j \in [1, m]$, donde $x_{i,j} \in \mathbb{N}$.

Propósito Esta variable permite cuantificar y analizar las transiciones entre diferentes opiniones, proporcionando una base para evaluar la dinámica de cambio de opinión en el modelo y para aplicar estrategias de moderación si es necesario.

2.3. Restricciones

- **Costo Máximo (Lineal):** Esta restricción limita el costo total de todos los movimientos a un presupuesto máximo `ct`. Se clasifica como **lineal** porque la variable `costo` es de tipo `float` (continua).

$$\text{costo} \leq \text{ct}$$

- **Límite de Movimientos (Lineal y Entera):** Asegura que el total de movimientos realizados no supere el límite `maxM`. Es una restricción **lineal y entera** porque la variable `movimientosTotales` es de tipo `int` y se compara con una constante entera.

$$\text{movimientosTotales} \leq \text{maxM}$$

- **Conservación de la Población de Origen (Lineal y Entera):** Para cada opinión i, esta restricción clave impide que se muevan más personas de las que existen inicialmente en esa opinión (p_i). Es **lineal y entera**, ya que opera sobre sumas de las variables enteras $x_{i,j}$.

$$\forall i \in [1, m] : \sum_{j=1}^{m} x_{i,j} \leq p_i$$

- **Prohibición de Movimientos Nulos (Lineal y Entera):** Impide las transiciones de una opinión a sí misma, fijando la diagonal de la matriz x a cero. Es una restricción simple **lineal y entera**.

$$\forall i \in [1, m] : x_{i,i} = 0$$

- **Límite de Población Total (Lineal y Entera):** Garantiza que el número total de personas movidas no exceda la población total n. Es **lineal y entera** porque `personaOpinion` es `int`.

$$\text{personaOpinion} \leq n$$

- **Movimientos entre Extremismos Iguales (Lineal y Entera):** Prohíbe mover personas entre dos opiniones distintas si estas comparten el mismo valor de extremismo. Es una restricción **lineal y entera** que reduce el espacio de búsqueda.

$$\forall i, j \in [1, m] \text{ donde } \text{ext}_i = \text{ext}_j \implies x_{i,j} = 0$$

- **No Negatividad (De Cota):** Son restricciones de cota que aseguran que las variables no tomen valores negativos. Son un tipo fundamental de restricción lineal y entera.
  - costo ≥ 0
  - movimientosTotales ≥ 0
  - $\forall i, j \in [1, m] : x_{i,j} \geq 0$

2.4. Función Objetivo y Propósito del Modelo

2.4.1. Fórmula del Objetivo

El objetivo principal del modelo es **minimizar** el extremismo total de la población en su estado final. La función objetivo se define como:

$$\text{minimizar} \sum_{i=1}^{m} \text{nueva\_p}_i \cdot \text{ext}_i$$

Donde nueva_p_i es una variable que representa la población final en la opinión i después de aplicar todos los movimientos $x_{i,j}$.

2.4.2. Explicación y Justificación

Explicación: La función objetivo calcula una métrica de **extremismo agregado** para toda la población. Esto se logra a través de una suma ponderada: para cada opinión posible, se multiplica su extremismo (el parámetro ext_i) por el número de personas que finalmente pertenecen a esa opinión (la variable $nueva\_p_i$). De esta manera, las opiniones con valores altos de extremismo y una gran cantidad de adherentes contribuyen más al valor total de la función.

Justificación: El propósito fundamental del modelo es encontrar la estrategia de moderación más efectiva que se ajuste a las restricciones de recursos (como el costo ct y el número de movimientos $maxM$). Al establecer la minimización del extremismo total como el objetivo, el modelo no se limita a realizar movimientos al azar. En cambio, el solver está instruido para encontrar la combinación de movimientos ($x_{i,j}$) que deliberadamente desplace a la población de opiniones más extremas a otras menos extremas, resultando en el menor valor posible de la función objetivo.

3. Datos Relevantes de la Implementación

Para llevar a la práctica la solución del problema `MinExt`, se desarrolló una aplicación web que facilita la interacción del usuario con el modelo de optimización, desde la carga de datos hasta la visualización de los resultados. La arquitectura de la solución se basa en una clara separación entre el cliente y el servidor.

Desarrollo del Backend: El servidor de la aplicación fue desarrollado en **Python** utilizando el micro-framework **Flask**, el cual se encarga de gestionar las rutas y la lógica de negocio. El backend es responsable de recibir los archivos de instancia `.txt` cargados por el usuario, validarlos y procesarlos. Una vez que los datos son parseados, el servidor genera dinámicamente un archivo de datos en formato `.dzn`, que es el formato requerido por MiniZinc para definir los parámetros de una instancia específica del problema. Posteriormente, invoca al motor de **MiniZinc** a través de un subproceso del sistema, especificando el uso del solver **Coin-BC (COIN-OR Branch and Cut)**. Una vez que el solver finaliza, el backend captura la salida, la procesa para extraer las métricas clave y la envía al frontend en un formato JSON estructurado.

Desarrollo del Frontend: La interfaz de usuario fue construida con **HTML**, **CSS** y **JavaScript**. Para la manipulación dinámica del contenido y la comunicación asíncrona con el servidor se utilizó la biblioteca **jQuery**, que permite realizar solicitudes AJAX para enviar el archivo de instancia y recibir los resultados sin necesidad de recargar la página. La presentación visual de los datos se enriquece mediante el uso de **Chart.js**, que se utiliza para mostrar tanto la distribución inicial de la población como la distribución final obtenida tras la optimización. El diseño y la maquetación se apoyan en el framework de CSS **TailwindCSS**.

Flujo de Interacción: El proceso completo es transparente para el usuario final. Comienza cuando el usuario selecciona un archivo de instancia en la interfaz. El frontend envía este archivo al servidor Flask. El servidor lo procesa, ejecuta el modelo de optimización y, tras obtener una respuesta del solver, la devuelve al cliente. Finalmente, el frontend interpreta los datos recibidos y actualiza la vista dinámicamente, mostrando tablas con resúmenes numéricos, la matriz de movimientos y gráficos comparativos que permiten un análisis visual e inmediato de la efectividad de la solución encontrada.

Descripción del modelo en MiniZinc. La implementación del problema se materializó en un modelo declarativo utilizando el lenguaje MiniZinc. El diseño del modelo inicia con la definición de un conjunto de **parámetros de entrada** que caracterizan cualquier instancia del problema, como la población, los costos y los niveles de extremismo. El núcleo del problema reside en la **variable de decisión** principal, una matriz `x` que representa el flujo de personas entre cada par de opiniones. Para mejorar la legibilidad y la modularidad, se utilizan **variables** que encapsulan cálculos complejos, como el costo total (incluyendo una lógica condicional para los cargos extra), el total de movimientos y la distribución final de la población. El comportamiento del sistema está regido por un conjunto de **restricciones** explícitas que definen el espacio de soluciones factibles, asegurando el cumplimiento de los límites de recursos, la conservación de la población y las reglas lógicas del negocio. Finalmente, el modelo establece una **función objetivo** clara: minimizar el extremismo total en la distribución final de opiniones, instruyendo así al solver para que encuentre la solución óptima entre todas las configuraciones posibles.

Factibilidad y Escalabilidad de la Solución

El tiempo necesario para encontrar una solución válida depende directamente del tamaño de la población (n) y del número de posibles opiniones (m), ya que estos parámetros influyen en la complejidad y el costo del modelo. Cuanto mayor sean n y m, mayor será el costo y el tiempo de procesamiento. Además, el número de recursos disponibles también impacta en la rapidez con la que el modelo puede generar una solución. Para asegurar que la solución sea tanto precisa como factible, el modelo toma en cuenta restricciones clave, incluyendo limitaciones en el número de movimientos y los costos adicionales asociados con los cambios de opiniones no representadas inicialmente. Esto permite respetar los límites de recursos y garantiza que las soluciones propuestas sean realistas y alcanzables.

4. Análisis Branch and Bound

El árbol de búsqueda generado por el solver Gecode/Gist proporciona un mapa visual del proceso de razonamiento seguido para encontrar la solución óptima. A través del método *Branch and Bound*, el solver explora sistemáticamente el espacio de soluciones, podando ramas que no pueden conducir a un resultado mejor que el ya encontrado. A continuación, se detalla el recorrido

del árbol paso a paso.

Instancia de Prueba Utilizada

A continuación se presenta la instancia en formato `.dzn` que fue resuelta para generar el árbol de búsqueda analizado. Los parámetros de costo (`ct`) y movimientos (`maxM`) fueron ajustados para forzar al solver a tomar decisiones de ramificación estratégicas.

- $n = 20$
- $m = 5$
- $p_i = [8, 6, 4, 2, 0]$
- $ext_i = [1{,}0, 0{,}8, 0{,}5, 0{,}2, 0{,}1]$
- $c_{i,j} =$

	1	**2**	**3**	**4**	**5**
1	0	8,0	5,0	3,0	1,0
2	10,0	0	4,0	2,0	3,0
3	15,0	12,0	0	1,0	2,0
4	18,0	15,0	12,0	0	1,0
5	25,0	22,0	20,0	18,0	0

- $c_j^e = [10{,}0, 10{,}0, 10{,}0, 10{,}0, 40{,}0]$
- $c_t = 6{,}0$
- máx $M = 7$

Visualización del Árbol de Búsqueda

La ejecución de la instancia anterior con el solver Gecode/Gist generó el siguiente árbol de búsqueda, que sirve como base para el análisis del recorrido.

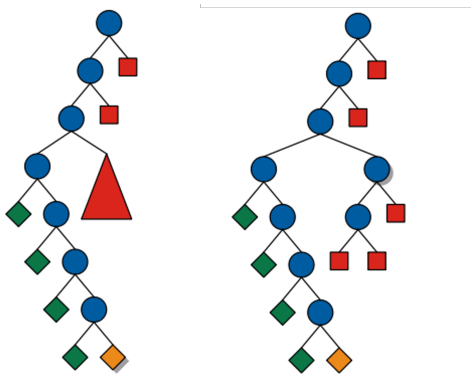

Figura 1: Expansión del Arbol Branch and Bound.

Paso 1 - El Punto de Partida (Nodo Raíz): La búsqueda se inicia en el nodo raíz (**el círculo azul superior**). En este estado inicial, el solver solo ha procesado los parámetros de entrada y las restricciones. Las variables de decisión aún no tienen valores fijos, sino que se representan por sus dominios iniciales, que son amplios y reflejan todas las posibilidades teóricas. Por ejemplo, el costo potencial se encuentra en un rango inicial de `[0..19.4]`, y las variables de movimiento (`x`) permiten múltiples transiciones. El objetivo en este punto es comenzar a tomar decisiones para acotar este vasto espacio de búsqueda.

Paso 2 - Exploración Inicial y Poda por Infactibilidad: El solver desciende por el árbol y comienza a tomar las primeras decisiones. En los niveles superiores, se encuentra rápidamente con nodos de fallo (**los cuadrados rojos**). Estos nodos representan estados en los que se ha violado una o más restricciones de manera irreparable. Por ejemplo, al intentar un movimiento, el costo acumulado puede haber excedido el límite `ct` o el número de movimientos el límite `maxM`. Al detectar esta infactibilidad, el solver aplica la técnica de *poda (pruning)*, descartando por completo estas ramas y evitando así una exploración innecesaria. El solver retrocede (backtracking) y continúa por el único camino que, hasta el momento, sigue siendo viable.

Paso 3 - El Punto de Ramificación (El Triángulo Rojo): A medida que desciende, el solver llega a un punto crítico, representado por el **triángulo rojo**. Este nodo es un **punto de ramificación (branching point)**. Aparece porque la simple propagación de restricciones ya no es suficiente para que el solver deduzca el siguiente paso. Se enfrenta a un dilema y debe tomar una decisión estratégica sobre una variable que aún no está fija. Para resolverlo, crea dos o más ramas nuevas, una para cada posible decisión. En este caso, el análisis de la expansión del triángulo revela lo siguiente:

- **Exploración de la Rama Derecha:** El solver elige explorar una de las nuevas ramas. Este camino resulta ser infructuoso. Al seguirlo, los nodos subsiguientes (**los cuadrados rojos dentro de la expansión del triángulo**) rápidamente violan una restricción. En este caso, aunque el costo se mantenía dentro del rango, es probable que se haya superado el número máximo de movimientos permitidos (`maxM`). El solver poda esta sub-rama por completo.
- **Exploración de la Rama Izquierda:** Tras el fallo anterior, el solver retrocede y explora la otra alternativa creada en el punto de ramificación. Este camino resulta ser prometedor y es el que el solver continuará explorando para buscar la solución óptima.

Paso 4 - Búsqueda de Soluciones y Mejora de la Cota: Continuando por el camino viable, el solver encuentra sus primeras soluciones completas y factibles, representadas por los **rombos verdes**. La primera de estas soluciones establece la **cota superior inicial (upper bound)** para la función objetivo (el *récord a batir* de extremismo). Cada vez que se encuentra una nueva solución factible con un valor de extremismo inferior al récord actual, esta nueva solución se convierte en la mejor encontrada hasta ahora y la cota superior se actualiza. Esta cota es fundamental para

el algoritmo, ya que le permite podar cualquier otra rama que, según sus cálculos, no tenga el potencial de generar una solución mejor que la que ya posee.

Paso 5 - Convergencia a la Solución Óptima: Finalmente, tras haber explorado o podado todas las ramas del árbol, el solver llega al **rombo naranja**. Este nodo representa la **solución óptima final**. En este punto, todas las variables de decisión tienen un valor específico. Para este problema, la solución óptima consistió en mover 4 personas de la opinión 3 a la 4, con un costo total de 4.8 y una distribución final de la población de `[8, 6, 0, 6, 0]`. El solver puede garantizar que esta es la mejor solución posible porque todas las demás alternativas han sido sistemáticamente evaluadas y descartadas, ya sea por ser infactibles o por conducir a soluciones de peor calidad que la óptima encontrada.

5. Pruebas

5.1. Rendimiento del equipo

Las pruebas fueron ejecutadas en un equipo con sistema operativo Windows 11 Pro, equipado con un procesador Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz, con 6 núcleos físicos y 12 hilos lógicos, y 64 GB de memoria RAM. Para la resolución de las instancias se utilizó MiniZinc versión 2.9.3 (build 1832600886) junto con el solver COIN-BC 2.10.12/1.17.10. Este entorno computacional permitió llevar a cabo simulaciones de alta demanda, haciendo posible procesar casos complejos con grandes cantidades de personas, opiniones y restricciones.

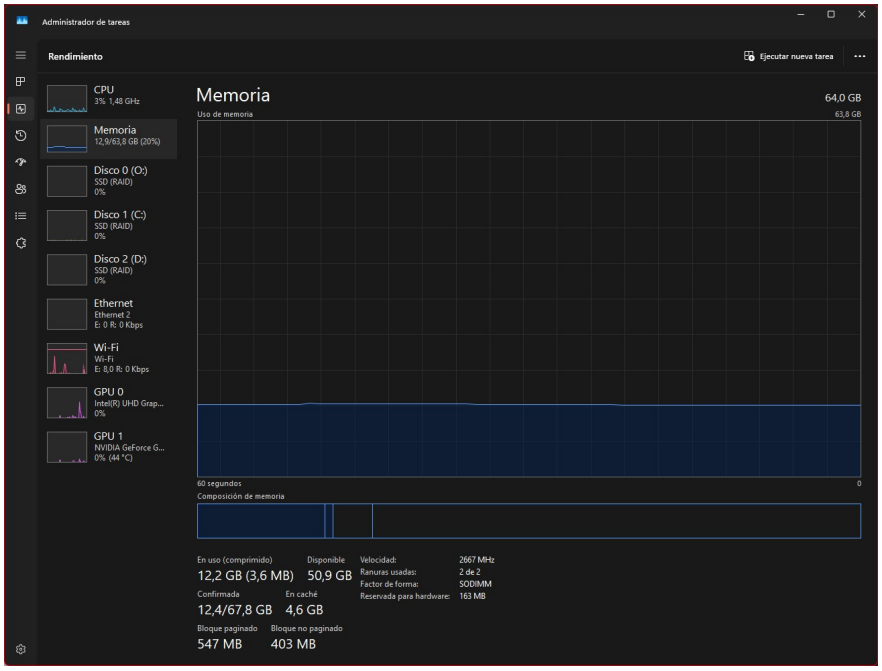

Figura 2: Rendimiento del equipo en el que se realizaron las pruebas

5.2. Resultados obtenidos con las intancias de la batería de pruebas

Prueba	Personas	Opiniones	Costo Máximo	Mov. Máx.	Tiempo	Ext. Inicial	Ext. Final	Costo Total	Mov. Totales	% Reducción
Enunciado	20	5	22,00	18	0,18	9,8	6,3	22	8	35,71 %
Prueba del informe	20	5	6,00	7	0,19	15,2	14	4,8	4	7,89 %
Prueba1.txt	10	5	75,01	9	0,17	6,145	3,846	22,74	9	37,41 %
Prueba2.txt	10	5	24,14	4	0,19	7,409	4,582	8,35	4	38,16 %
Prueba3.txt	10	5	81,78	22	0,19	2,905	0,261	72,53	14	91,02 %
Prueba4.txt	10	5	96,95	25	0,16	3,844	1,06	19,65	11	72,42 %
Prueba5.txt	10	5	128,25	1	0,18	5,369	4,448	4,12	1	17,15 %
Prueba6.txt	15	5	60,84	10	0,18	12,123	9,583	29,05	10	20,95 %
Prueba7.txt	15	5	141,96	25	0,17	5,275	0,829	76,64	25	84,28 %
Prueba8.txt	20	5	112,21	27	0,20	8,367	3,541	38,99	27	57,68 %
Prueba9.txt	20	5	87,46	9	0,17	8,645	3,654	33,30	9	57,73 %
Prueba10.txt	20	5	17,35	32	0,18	13,303	10,353	16,79	18	22,18 %
Prueba11.txt	40	10	151,21	58	0,19	15,22	2,12	130,00	58	86,07 %
Prueba12.txt	40	10	210,90	16	0,19	21,10	13,339	51,34	16	36,78 %
Prueba13.txt	40	10	7,79	14	0,20	22,623	15,977	7,51	14	29,38 %
Prueba14.txt	40	15	180,06	35	0,22	13,79	2,686	85,27	35	80,52 %
Prueba15.txt	40	15	92,67	75	0,21	16,612	2,729	92,49	74	83,57 %
Prueba16.txt	60	15	218,28	86	0,21	40,015	10,099	158,62	86	74,76 %
Prueba17.txt	60	15	341,71	112	0,21	33,925	10,496	145,73	112	69,06 %
Prueba18.txt	60	15	307,48	111	0,22	32,041	6,618	93,54	111	79,35 %
Prueba19.txt	75	15	351,34	56	0,20	33,053	19,749	96,70	56	40,25 %
Prueba20.txt	75	15	427,21	106	0,21	51,375	21,65	174,63	106	57,86 %
Prueba21.txt	100	10	465,21	67	0,20	49,512	15,703	209,57	67	68,28 %
Prueba22.txt	100	10	88,92	117	0,22	51,216	13,868	84,50	116	72,92 %
Prueba23.txt	100	20	505,07	38	0,23	49,116	27,999	48,68	38	42,99 %
Prueba24.txt	100	20	111,23	184	0,22	35,201	3,787	110,94	184	89,24 %
Prueba25.txt	100	20	309,98	8	0,22	44,634	37,859	21,00	8	15,18 %
Prueba26.txt	150	20	872,85	283	0,23	79,792	14,28	288,75	283	82,10 %
Prueba27.txt	150	20	653,84	261	0,23	74,552	11,404	547,54	261	84,70 %
Prueba28.txt	200	25	750,26	1	0,24	94,074	93,27	3,02	1	0,85 %
Prueba29.txt	200	25	1018,16	275	0,27	109,717	27,626	396,51	275	74,82 %
Prueba30.txt	200	25	243,66	339	0,33	117,456	48,086	243,54	339	59,06 %

Cuadro 1: Resultados de la bateria de pruebas.

5.3. Resultados obtenidos con las 5 instancias retadoras

Prueba	Personas	Opiniones	Costo Máximo	Mov. Máx.	Tiempo	Ext. Inicial	Ext. Final	Costo Total	Mov. Totales	% Reducción
PruebaRetadora1.txt	500	35	5000	800	0.37	239.258	34.368	911.71	800	**85.64 %**
PruebaRetadora2.txt	700	45	20000	2000	0.53	376.172	36.532	1673.18	2000	**90.29 %**
PruebaRetadora3.txt	600	40	500	2000	0.59	255.533	46.267	499.91	2000	**81.89 %**
PruebaRetadora4.txt	1000	60	30000	1000	0.80	493.238	156.861	1735.91	1000	**68.20 %**
PruebaRetadora5.txt	1000	50	8000	200	0.57	472.691	340.034	583.69	200	**28.06 %**

Cuadro 2: Resultados obtenidos por el modelo con instancias retadoras de gran tamaño y restricciones complejas.

6. Descripción de las pruebas realizadas

Estas pruebas simulan el proceso de moderación y agrupación de opiniones, permitiendo evaluar cómo responde el modelo ante distintas configuraciones de entrada.

Cada prueba se basa en una instancia diferente del problema, esta es definida por los parametros de entrada, los criterios para las instancias probadas incluyen variaciones en la distribución inicial de moderación y la intensidad del extremismo entre las personas.

De las pruebas se puede deducir que:

- Para una población pequeña y esfuerzo bajo se puede evidenciar que el modelo es capaz de cumplir con las restricciones proporcionadas, favoreciendo que la distribución de personas se concentre entre las pocas opciones compatibles.

- Al incrementar el esfuerzo máximo permitido, se puede evidenciar una disminución notable del extremismo. Dado a que se proporciona una mayor flexibilidad para realizar redistribuciones, lo cual implica reducir el conflicto interno a costa de un mayor esfuerzo.
- Cuando el costo de mover personas es significativo en relación con el esfuerzo total permitido, el modelo priorizará cambios de bajo esfuerzo o en algunos casos evitará realizar ciertos movimientos. Lo cual puede llevar a que algunos grupos lleguen a permanecer sin cambios y el extremismo no disminuya considerablemente, dado que solo algunos movimientos son viables.
- Cuando el esfuerzo total permitido es bajo, se puede presentar la situación en la cual el modelo no encuentra una solución factible, dado que en este caso el modelo no podría realizar los movimientos suficientes para cumplir con las restricciones.

7. Análisis de las pruebas

A partir de los resultados obtenidos en la batería de pruebas, es importante resaltar que el modelo implementado en MiniZinc logra cumplir con las restricciones establecidas y en todos los casos logra obtener una solución óptima en terminos del extremismo final.

Gráfica de Extremismo inicial y Extremismo final

La siguiente gráfica muestra una comparación entre el nivel de extremismo inicial proporcionado y el extremismo final obtenido mediante el modelo. Se puede observar que el extremismo final corresponde al resultado óptimo esperado, ya que se busca que este sea siempre menor que el valor inicial.

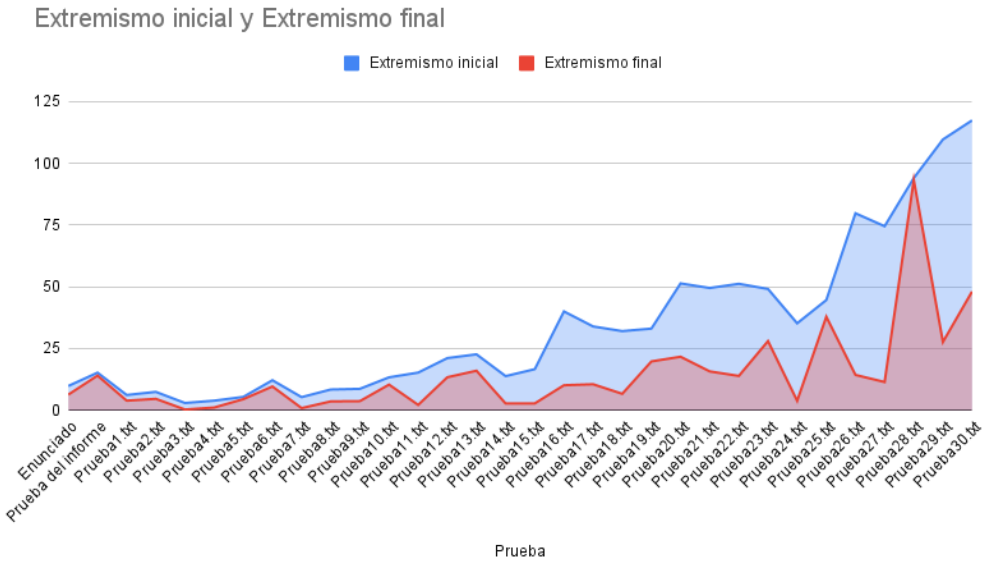

Figura 3: Comparación entre los niveles de extremismo inicial y final obtenidos en las pruebas

Grafica de Costo total y Extremismo final

La siguiente gráfica permite observar que, en general, cuanto mayor cantidad de personas y movimientos totales se hayan realizado en el modelo, disminuye el extremismo final y aumenta el costo total de la solución.

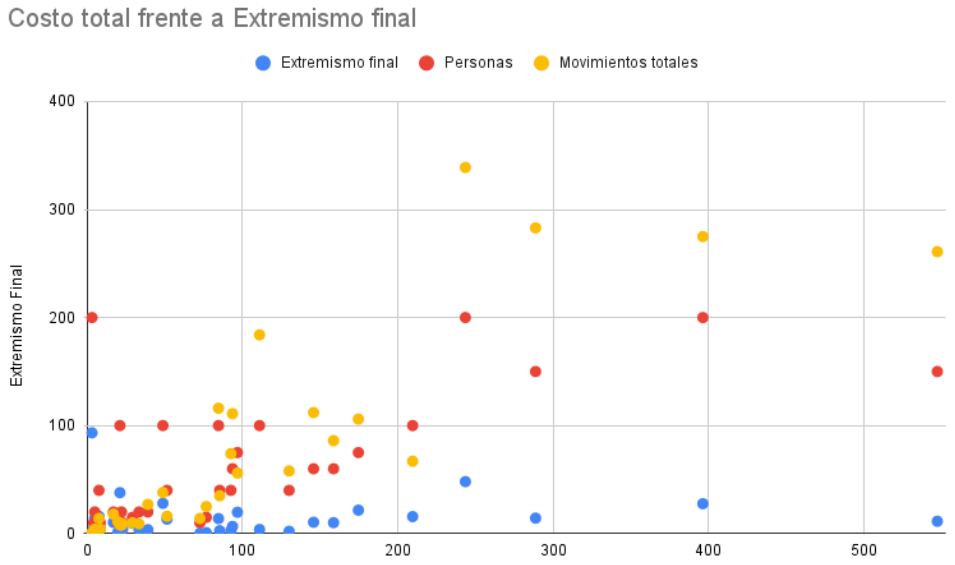

Figura 4: Comparación entre el costo total y el extremismo final obtenidos en las pruebas

7.1. Gráfica de Costo máximo y Costo total

A través de la siguiente gráfica se puede evidenciar que en la mayoría de casos el costo total obtenido por el modelo se mantiene por debajo del costo inicial, lo que sugiere que el modelo no requiere agotar completamente el costo disponible para alcanzar soluciones óptimas.

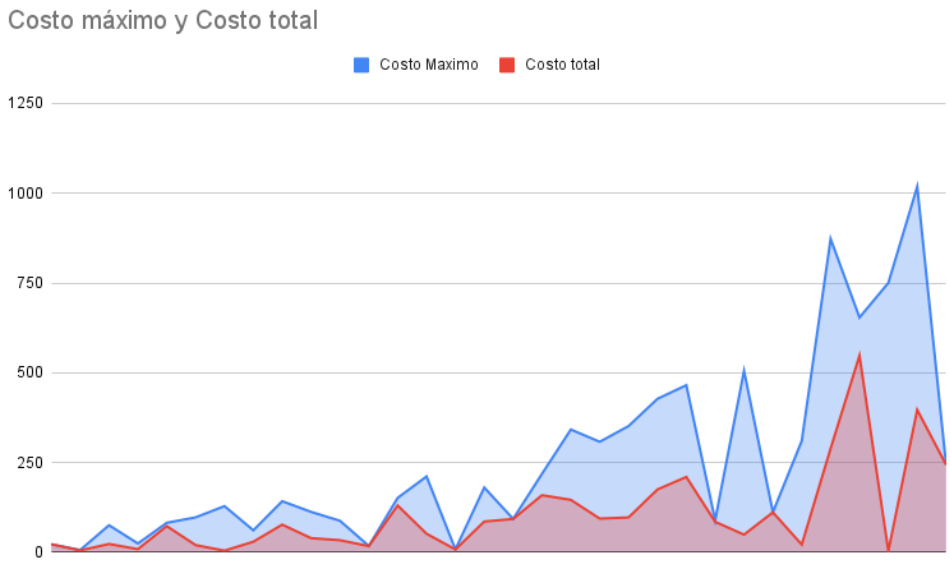

Figura 5: Comparación entre el costo máximo asignado y el costo total obtenido por el modelo

8. Conclusiones

A partir del desarrollo e implementación del modelo para minimizar el extremismo presente en una población (MinExt), se pueden destacar diversas conclusiones importantes:

- **Eficacia del Modelo en la Reducción del Extremismo:** El modelo de optimización, implementado mediante programación entera mixta, demostró ser altamente eficaz para reducir el nivel de extremismo en una población. Los resultados obtenidos en la batería de 35 pruebas confirman que, en todos los casos, se encontró una solución que disminuyó el extremismo inicial, logrando en algunos escenarios reducciones de hasta un 91.0 %. Esto valida el enfoque como una herramienta estratégica para la moderación de opiniones.
- **Impacto de los Recursos en la Solución:** Se evidenció una correlación directa entre los recursos disponibles (costo y movimientos máximos) y la capacidad del modelo para reducir el extremismo. Las pruebas demuestran que una mayor flexibilidad en estos parámetros permite al solver explorar soluciones más efectivas que implican mayores redistribuciones de la población, llevando a una disminución más significativa del conflicto interno. Por el contrario, restricciones muy ajustadas limitan las opciones del modelo, resultando en reducciones de extremismo más modestas, como se vio en la prueba 28 con una reducción de solo el 0.9 %.

- **Viabilidad y Rendimiento Computacional:** La elección de MiniZinc junto con el solver COIN-BC y el método Branch and Bound fue adecuada para abordar la complejidad del problema. El análisis del árbol de búsqueda revela cómo el solver poda ramas infactibles de manera eficiente, convergiendo a una solución óptima garantizada. A pesar de la naturaleza NP-difícil de los problemas de optimización combinatoria, los tiempos de ejecución para todas las instancias, incluso las más grandes con 200 personas y 25 opiniones, se mantuvieron bajos (generalmente por debajo de 0.3 segundos), lo que demuestra la escalabilidad y aplicabilidad del modelo para casos de uso prácticos.

Referencias

Belegundu, A. D., & Chandrupatla, T. R. (2019). *Optimization Concepts and Applications in Engineering* (3.ª ed.). Cambridge University Press.

Google. (2025). Gemini 2.5 Pro: Modelo de lenguaje de gran escala [Consultado en julio de 2025]. https://gemini.google.com

OpenAI. (2025). ChatGPT: Modelo de lenguaje de gran escala de OpenAI [Consultado en julio de 2025]. https://chat.openai.com