# Estudio de propiedades de repartos con optimización de bienestar social igualitario

Máster en Métodos Formales en Ingeniería Informática

Universidad Complutense de Madrid
Facultad de Informática


Jonathan Carrero

*Dirigido por*: Ismael Rodríguez y Fernando Rubio

Curso 2019/2020.
Convocatoria de febrero.
Calificación: **NOTABLE 8,5**.

*"We can only see a short distance ahead,*
*but we can see plenty there that needs to be done."*

Alan Mathison Turing

## Agradecimientos

En primer lugar, quisiera dar las gracias a las personas sin las que ahora mismo ni siquiera podría estar dando las gracias: a mis dos directores, Ismael y Fernando. Además, quiero dar las gracias a todos aquellos familiares y amigos que se han preocupado por mí y siempre tenían un hueco para preguntarme si todo marchaba bien.

También, y aunque la inmensa mayoría nunca vaya a leer estas palabras, me gustaría dar las gracias a todas las personas que han influido en mí hasta el día de hoy. Todas y cada una de ellas han sido importantes; desde los últimos profesores que tuve durante este máster hasta los primeros cuando tan solo era un crío. Si alguno de ellos hubiera faltado, tal vez ahora mismo no sería exactamente la persona que soy.

Por último, al igual que en mi Trabajo de Fin de Grado, quiero volver a agradecer a todas las personas que, independientemente del campo en el que trabajen, ponen a disposición de los demás todo su conocimiento, haciendo del mundo un lugar mejor.

A todos, gracias.

## Índice de contenido

## Índice de Ilustraciones

## Índice de Tablas

## Resumen / Abstract

Durante mucho tiempo, el área de la economía ha sido un campo con un notable crecimiento que no se ha visto afectado por las nuevas tecnologías hasta hace unas pocas décadas. Con el auge de las ciencias de la computación, la economía ha sufrido notables cambios en todos sus aspectos. Uno de ellos es el de realizar repartos de recursos de la manera más eficiente posible. Abordar este tipo de problemas antes de la aparición de la computación moderna era una tarea enormemente lenta y costosa en tiempo, pero hoy en día es posible desarrollar pruebas altamente eficientes que permitan averiguar cómo se realizan estos repartos de recursos bajo unas determinadas reglas impuestas de antemano. A lo largo de esta memoria exploraremos un modelo concreto de reparto de recursos y estudiaremos diferentes tipos de experimentos que analizaremos y sobre los que obtendremos conclusiones que nos permiten saber cómo se comportan este tipo de problemas.

---

For a long time, the area of economics has been a growth field that has not been affected by new technologies until a few decades ago. With the rise of computer science, the economy has undergone changes in all its aspects. One of them is the need to allocate resources as efficiently as possible. Addressing these problems before the advent of modern computing was an enormously slow and time-consuming task, but today it is possible to develop highly efficient tests that can be used to find out how these resource allocations are made under certain pre-imposed rules. Throughout this report we will explore a specific model of resource allocation and study different types of experiments that we will analyze and on which we will obtain conclusions that allow us to know how these types of problems behave.

## Palabras clave / Keywords

Egalitarian social welfare, asignación de recursos multiagente, algoritmos genéticos, bienestar social.

---

Egalitarian social welfare, multi-agent resource allocation, genetic algorithms, social welfare.

## Listado de abreviaturas por orden alfabético

- AGI: Algoritmo Genético Inferior.
- AGS: Algoritmo Genético Superior.
- MARA: Multi-Agent Resource Allocation.
- TFM: Trabajo Fin de Máster.

# 1. Introducción

Existen una infinidad de escenarios en los que buscar la eficiencia al realizar repartos de recursos supone un desafío. Tratar de resolver estos repartos de la manera más eficiente posible implica grandes cantidades de tiempo. Es por eso que estamos interesados en buscar resultados en los que, aun empleando tiempos que no son considerablemente altos, aportan un grado de eficiencia aceptable.

Cuando mencionamos estos repartos, podríamos estar hablando de problemas que se presentan de forma natural en diferentes sectores, como por ejemplo el reparto de recursos tales como el ancho de banda o memoria en un entorno computacional [**1**], [**2**], [**3**]. En cualquier caso, independientemente de la naturaleza de la que provenga el problema con el que estemos tratando, muchos de ellos pueden generalizarse y agruparse. De esta forma, conseguimos abstraernos de los problemas concretos y abordamos uno más general que supone una visión más amplia y que a posteriori puede ser utilizada en los casos concretos.

En este Trabajo de Fin de Master vamos a centrarnos en uno de estos casos. Concretamente, vamos a abordar la Asignación de Recursos Multiagente, comúnmente denominado como MARA[1]. En este modelo, se realiza el reparto de algunos recursos a algunos agentes, los cuales, previamente, han establecido unas determinadas preferencias para cada uno de los recursos a repartir. Conozcamos los conceptos que van a intervenir en nuestro reparto.

## 1.1. Definiendo el problema general

A la hora de realizar un reparto de recursos, debemos definir algunos conceptos que nos serán útiles durante las siguientes secciones. En primer lugar, un reparto de recursos consiste en la asignación de recursos a unos determinados agentes (participantes que reciben los recursos). Esta asignación se lleva a cabo por un participante ajeno a los agentes, el cual recibe el nombre de repartidor. Al mismo tiempo, es necesario que cada uno de los agentes que participan en el reparto asignen una determinada preferencia a cada uno de los recursos disponibles. Esta preferencia indica el interés de cada agente por cada recurso; cuanto mayor sea, más interés por el recurso en cuestión.

Antes de que el repartidor proceda a realizar el reparto de recursos entre los agentes, estos deben comunicarle qué preferencias tienen por los recursos disponibles. De esta forma, el repartidor es el único participante que conoce las preferencias que tienen todos los agentes por todos los recursos. Una vez que el repartidor conoce todas las preferencias, es cuando procede a realizar el reparto, pero no de cualquier forma; existen una gran variedad de objetivos que puede perseguir el repartidor cuando lleva a cabo el reparto, como por ejemplo Utilitarian Social Welfare, Nash Product, Rank Dictators, etc. [**4**], [**5**]. Al final, una vez

[1] Multi-Agent Resource Allocation.

que se ha hecho el reparto e independientemente de cuál fuese el objetivo del repartidor, cada agente recibe una determinada utilidad. La utilidad es una medida que refleja la satisfacción de los agentes una vez finalizado el reparto. Esta utilidad, como es de esperar, procede de los recursos que finalmente han sido asignados a cada agente.

Llegados a este punto en el que conocemos las nociones básicas del problema de reparto de recursos, es el momento de abordar el modelo concreto con el que vamos a trabajar. Es decir, vamos a ver qué objetivo queremos perseguir tal y como si nosotros mismos estuviésemos en el papel de repartidor.

## 1.2. Egalitarian Social Welfare

A la hora de realizar el reparto de recursos, deben establecerse algunas variables que determinan cómo se lleva a cabo el reparto. Asumiremos que la utilidad total de cada agente con cada reparto será la suma de sus preferencias hacía los recursos que le son asignados (preferencias *aditivas*). En primer lugar, debemos estar de acuerdo en el objetivo que queremos perseguir cuando se haga el reparto. Para nosotros, el objetivo consistirá en maximizar la utilidad del agente cuya utilidad resulta ser la mínima. Este modelo recibe el nombre de *Egalitarian social welfare*.

Antes de continuar profundizando en este modelo, veamos un simple ejemplo que refleje este objetivo. Veremos más adelante que, incluso dentro de este modelo, vamos a poder modificar ciertas variables para obtener diversos subproblemas.

| | **Recurso 0** | **Recurso 1** | **Recurso 2** | **Recurso 3** |
|---|---|---|---|---|
| **Agente 0** | 20 | 32 | 40 | 1 |
| **Agente 1** | 32 | 53 | 76 | 42 |
| **Agente 2** | 27 | 7 | 6 | 21 |

Tabla 1: Ejemplo de preferencias

Supongamos un caso muy concreto en el que contamos con tres agentes y disponemos de cuatro recursos a repartir. Como vemos en la **Tabla 1**, cada uno de los agentes tiene una determinada preferencia por cada recurso. En este punto, nos disponemos (ya que estamos haciendo el papel de repartidor) a realizar el reparto de recursos y nos preguntamos: ¿cuál será la mejor distribución tal que maximicemos la utilidad obtenida por el agente que menos utilidad logra obtener? Por ejemplo, yendo de nuevo a la tabla, observamos que el Agente 0 tiene una preferencia de 20 unidades por el Recurso 0. Es decir, que si establecemos una solución tal que en dicho reparto el Agente 0 tan solo recibe el Recurso 0, entonces el Agente 0 obtendrá una utilidad de 20 unidades.

Podemos observar que el número de repartos posibles, incluso en un modelo tan sencillo como este, enseguida crece de manera exponencial. Pero, para este primer ejemplo,

consideremos únicamente las soluciones mostradas en la siguiente tabla y veamos qué solución es mejor bajo nuestro modelo de *Egalitarian social welfare*.

| | **Recurso 0** | **Recurso 1** | **Recurso 2** | **Recurso 3** |
|---|---|---|---|---|
| **Solución 1** | 0 | 0 | 1 | 2 |
| **Solución 2** | 0 | 2 | 1 | 1 |
| **Solución 3** | 2 | 1 | 2 | 2 |

Tabla 2: Ejemplo de solución al reparto de recursos

En la **Tabla 2**, cada número índica el agente al que ha sido asignado el recurso en cuestión. Con la primera solución observamos que el Agente 0 recibe los Recursos 0 y 1, el Agente 1 recibe el Recurso 2 y el Agente 2 recibe el Recurso 3. Si calculamos la utilidad que obtiene cada agente, el resultado es que cada agente recibe 52, 76 y 21 unidades respectivamente. Recordemos que en este modelo nos interesa especialmente la utilidad mínima obtenida y, de entre todos los repartos posibles, escoger la mayor de todas ellas. El agente que menos utilidad recibe en la primera solución es el Agente 2 con 21 unidades. En la segunda solución, de nuevo es el Agente 2 con tan solo 7 unidades. Y por último, en la tercera solución, el agente que menos utilidad recibe es el Agente 0 con 0 unidades. Si ahora nos fijamos en la solución que aporta una mayor utilidad de entre las consideradas anteriormente, es fácil ver que la primera solución, en la que el Agente 2 recibe 21 unidades, es la de mayor valor. Luego nuestra mejor solución encontrada para este modelo es la Solución 1.

Con este sencillo ejemplo queda claro el objetivo que queremos perseguir: encontrar la solución óptima que maximice la utilidad del agente cuya utilidad es la mínima (o acercarnos tanto como podamos).

Hasta aquí el modelo es sencillo. Ahora bien, ¿qué ocurre si un agente comunica unas preferencias al repartidor que no son reales? ¿Podría un agente mentir fingiendo que un recurso le interesa muy poco solo para que haya más probabilidades de que se lo asignen? Es más, ¿podría hacer esto mismo con los $n$ recursos disponibles? Nótese que, si un agente tiende a bajar sus preferencias fingiendo que no le interesan los recursos que se han presentado en el reparto, entonces el repartidor, para lograr su objetivo (que es maximizar la menor utilidad), debe asignar muchos más recursos a dicho agente para así complacerle.

| | **Recurso 1** | **Recurso 2** | **Recurso 3** | **Recurso 4** |
|---|---|---|---|---|
| **Agente 0** | 2 | 6 | 3 | 1 |
| **Agente 1** | 32 | 53 | 76 | 42 |
| **Agente 2** | 27 | 7 | 6 | 21 |

Tabla 3: El Agente 0 finge preferencias más bajas

En efecto; en la **Tabla 2** el Agente 0 ha comunicado unas preferencias muy bajas al repartidor respecto al resto de agentes, en comparación con las de la **Tabla 1**. Ahora,

notemos que los demás agentes tendrán una utilidad que no será la mínima de los tres a poco que reciban 1 o 2 recursos. En particular, si el repartidor desea obtener una utilidad mínima alta, debe proporcionar muchos recursos al Agente 0 para así hacer que al menos esté igual de satisfecho que el resto de los agentes.

Este sencillo mecanismo que permite a los agentes mentir para así obtener más recursos a la hora de realizar la distribución de recursos, funciona (tal y como veremos durante los experimentos). Pero entonces, ¿existe una solución (ya sea total o parcial) para impedir que los agentes puedan mentir a la hora de realizar el reparto? Hasta ahora no parece haberse encontrado un mecanismo que logre eliminar completamente el incentivo que conlleva mentir a la hora de comunicar las preferencias al repartidor para el modelo de la *Egalitarian social welfare*. En particular, parece que el encontrar dicho mecanismo es una tarea considerablemente compleja, pues un agente siempre podrá reducir falsamente las preferencias comunicadas y así aumentar sus posibilidades de ser el agente con una utilidad mínima, la cual es, precisamente, la que el repartidor debe maximizar.

### 1.2.1. Desincentivando la mentira

A pesar de la aparente dificultad de eliminar por completo el incentivo que tiene mentir en las preferencias comunicadas, es fácil darnos cuenta de que realizando algunos cambios en las reglas de nuestro reparto podemos hacer que el hecho de mentir deje de ser tan sencillo para los agentes. En particular, si obligamos a los agentes a que la suma de las preferencias que comunican al subastador sea igual a una constante, entonces cada agente se verá obligado a que las preferencias de sus recursos siempre sumen lo mismo. Si esto es así, entonces un agente ya no podrá mentir y reducir sus preferencias de cualquier manera, ya que, si disminuye algunas, debe aumentar otras para que la suma se siga manteniendo.

| | **Recurso 0** | **Recurso 1** | **Recurso 2** | **Recurso 3** |
|---|---|---|---|---|
| **Agente 0** | 2 | 32 | 40 | 26 |
| **Agente 1** | 32 | 53 | 5 | 10 |
| **Agente 2** | 27 | 7 | 45 | 21 |

Tabla 4: Limitando las preferencias de los agentes

Supongamos que exigimos que la suma de todas las preferencias que tienen los agentes por los recursos disponibles debe sumar 100. Como vemos en la **Tabla 4** (creada como una modificación de la **Tabla 1**), si el Agente 0 desea mentir sobre la preferencia comunicada en el recurso 0, entonces deberá hacer que una o varias preferencias del resto de recursos aumente(n). Si nos fijamos, hemos incrementado las preferencias en color verde y disminuido las preferencias en color naranja.

Este mecanismo, como veremos más adelante, no elimina completamente el incentivo que se obtiene mintiendo, pero mentir ya no resulta tarea fácil. Y, de hecho, el incentivo por mentir se reduce considerablemente, lo cual es una muy buena primera aproximación a

lograr un mecanismo que desincentive totalmente la mentira bajo la *Egalitarian Social Welfare*, si es que lo hay.

### 1.2.2. Complejidad del problema

Si pasamos a estudiar la complejidad que tienen los problemas de repartos de recursos a la hora de tratar de optimizarlos, existen algunos artículos que recogen detalles sobre las complejidades en los modelos más comunes [**5**]. En nuestro caso, el modelo *Egalitarian* también ha sido estudiado en profundidad; Roos y Rothe [**6**] demostraron la NP–completitud de la *Egalitarian social welfare*, así como otros modelos parecidos que quedan fuera de este TFM.

Esta demostración presupone que los recursos que el repartidor debe distribuir son indivisibles e intransferibles, es decir: un recurso es asignado o no a un solo agente y, además, una vez asignado, ese recurso no puede ser transferido a otro agente hasta que el reparto haya finalizado. Estas dos condiciones también estarán implícitamente presentes durante los experimentos que llevaremos a cabo.

### 1.2.3. Formalizando el modelo Egalitarian

Una vez que nos hemos hecho una idea de cómo es el problema al que vamos a enfrentarnos y las características que tiene, conviene hacer una mejor definición del mismo. En esta pequeña sección vamos a introducir algunas definiciones formales, las cuales nos van a permitir establecer un marco más estricto del *Egalitarian*.

***Definición 1.*** Consideremos $R = \{r_1, \ldots, r_m\}$ como el conjunto de recursos disponibles y $A = \{A_1, \ldots, A_n\}$ como el conjunto de agentes sobre los que el repartidor asignará los recursos. El conjunto de posibles repartos de $R$ a $A$ es el conjunto $\mathcal{A} = A \times \ldots \times A_m$. Además, dado a $\alpha \in \mathcal{A}$ con $\alpha = (\alpha_1, \ldots, \alpha_m)$, decimos que el reparto $\alpha$ asigna cada recurso $r_j$ al agente $\alpha_j$ para todo $1 \leq j \leq m$.

***Definición 2.*** Una función de utilidad es una función que recibe una distribución de recursos y retorna un número real. Si la función de utilidad de un agente es $u$ y $u(\alpha_1) < u(\alpha_2)$, entonces el agente prefiere el reparto $\alpha_1$ sobre el reparto $\alpha_2$. Definimos una función de utilidad como $u : \mathcal{A} \rightarrow \mathrm{IR}$. La función de utilidad $u$ depende únicamente del agente $A_i$ si, para todo $\alpha = (\alpha_1, \ldots, \alpha_m) \in \mathcal{A}$ y $\beta = (\beta_1, \ldots, \beta_m) \in \mathcal{A}$ que satisfagan que $\alpha_j = \beta_j = A_j$ o $\alpha_j \neq A_j \neq \beta_j$ para toda $1 \leq j \leq m$, tenemos que $u(\alpha) = u(\beta)$. Además, decimos que $u$ es aditiva para cada agente $A_i$ si para todo $\alpha = (\alpha_1, \ldots, \alpha_m) \in \mathcal{A}$ tenemos que $u(\alpha) = \sum_{j \mid \alpha_j = A_i} u(\underbrace{A_k, \ldots, A_k}_{j-1}, A_i, \underbrace{A_k, \ldots, A_k}_{m-i-1})$, donde $A_k$ es un agente arbitrario con $A_k \neq A_i$.

Asumiremos que la función de utilidad de cada agente depende únicamente de dicho agente, y que es aditiva. Llegados a este punto, podemos definir el problema de optimización al que nos enfrentamos.

*Definición 3.* Dado $A = \{A_1, \ldots, A_n\}$, $R = \{r_1, \ldots, r_m\}$ y $U = (u_1, \ldots, u_n)$, para todo $\alpha \in \mathcal{A}$ sea $\mathbf{eg}_{A,R,U}(\alpha) = min\{u_j(\alpha) \mid 1 \leq j \leq n\}$. El problema de la optimización de la egalitarian social welfare consiste en, dado A, R, U, encontrar un $\alpha$ que maximice $\mathbf{eg}_{A,R,U}(\alpha)$. Denotaremos la solución al problema como **egsol**(A, R, U).

Pasamos a definir formalmente el problema de mentir de manera óptima para un determinado agente. El término $\mathbf{fake}_{A,R,U,i}(f)$ denota la utilidad lograda por el agente $A_i$ si la función de utilidad que comunica al repartidor es $f$.

*Definición 4.* Dado A, R, U y $i \in \mathbb{N}$, para todo $f \in U$ sea $\mathbf{fake}_{A,R,U,i}(f) = u_i(\alpha_f)$ con $\alpha_f =$ **egsol**(A, R, $(u_1, \ldots, u_{i-1}, f, u_{i+1}, \ldots, u_n)$). El problema de la utilidad falsa óptima consiste en, dado A, R, U, $i$, encontrar $f$ maximizando $\mathbf{fake}_{A,R,U,i}(f)$. Denotamos la solución para el problema A, R, U, $i$ como **fakesol**(A, R, U, $i$).

## 1.3. Cómo abordar el problema

Una vez que conocemos el escenario sobre el que vamos a trabajar, ha llegado la hora de explicar cómo vamos a estudiar nuestro problema de reparto de recursos bajo el modelo *Egalitarian* y su sensibilidad a la mentira. El primero de los conceptos que debe quedar claro es el de diferenciar entre preferencias *no limitadas* y preferencias *limitadas*. Para nosotros, las preferencias no limitadas serán aquellas que no tienen por qué sumar una constante, pudiendo cada agente establecer cualquier valor de preferencia para cada recurso disponible independientemente de los demás. Por el contrario, las preferencias limitadas sí deberán sumar una constante. Asimismo, en ambos casos los valores que establecen los agentes podrán ser cualesquiera dentro de un intervalo dado.

Nuestro estudio se centrará en estudiar la facilidad que podrían tener los agentes para mentir a la hora de comunicar sus preferencias (en el caso de preferencias no limitadas). Durante todos los experimentos que se harán en este Trabajo de Fin de Máster, asumiremos únicamente un agente mentiroso, que para nosotros será el Agente 0. En torno a él mediremos si realmente es eficaz mentir y hasta qué punto merece la pena hacerlo.

Abordaremos la resolución aproximada del problema utilizando la metodología de algoritmos genéticos, para lo cual se implementarán diferentes algoritmos que serán usados en función de los pequeños hitos que queramos ir consiguiendo.

## 1.4. Objetivos

El objetivo principal de este Trabajo de Fin de Máster es investigar, de forma experimental, qué modificaciones de las reglas a la hora de realizar repartos de recursos bajo la *Egalitarian social welfare* traen consigo un mayor desincentivo a la hora de mentir en las preferencias comunicadas por los agentes.

- En primer lugar, haremos que el Agente 0 mienta sobre sus preferencias de diferentes maneras: modificando la preferencia por el primer recurso o por los $n$ primeros recursos, por el recurso que menos valora (o por el que más), por el recurso que menos valora el resto de los agentes, etc. De esta forma veremos si el Agente 0 dispone de alguna estrategia fija que funciona en todos (o la mayoría) de los casos, incluso cuando el número de recursos y agentes es modificado. Estas primeras pruebas se llevarán a cabo con un algoritmo genético que hemos denominado como AGI (Algoritmo Genético Inferior).

- Más tarde nos centraremos en buscar dinámicamente cuál es la mejor mentira del Agente 0 para cada caso posible. De hecho, trataremos de encontrar la estrategia óptima para el Agente 0 cuando, de antemano, dicho agente realiza una estimación de las preferencias que comunicarán el resto de los agentes. Para encontrar la mentira que permita al Agente 0 obtener la mayor utilidad posible una vez realizado el reparto, utilizaremos otro algoritmo genético al que hemos denominado AGS (Algoritmo Genético Superior).

- Yendo un paso más allá, supongamos que hemos logrado el objetivo anterior: tenemos una manera de encontrar las mentiras óptimas para el Agente 0. Ahora bien, ¿qué ocurre si las preferencias del resto de los agentes no son exactamente las que el Agente 0 supuso? Veremos cómo de buena sigue siendo la mentira del Agente 0 a medida que las preferencias del resto de agentes difieren de las que el Agente 0 supuso en un principio.

## 1.5. Plan de trabajo

El plan de trabajo seguido fue el siguiente. En primer lugar, se explicó con detalle en qué consistía el problema que queríamos resolver y cómo íbamos a abordarlo. Cuando se trabaja en problemas relacionados con repartos de recursos, es fácil perder el objetivo (u objetivos) de vista, pues mínimas modificaciones pueden dar lugar a problemas muy diferentes. Una vez que se aclaró, el siguiente paso fue documentarse con estudios que ya habían sido hechos; tuvimos que buscar artículos relacionados con el modelo *Egalitarian* y ver si lo que íbamos a hacer realmente aportaba conocimiento nuevo dentro del área.

A continuación comenzó la fase de implementación; se desarrollaron varios algoritmos genéticos para realizar pruebas experimentales que aportasen información útil a nuestro

problema (o subproblemas). Cuando los algoritmos terminaban de ser implementados, había que probarlos para corroborar su correcto funcionamiento, luego en algunas ocasiones no era tan sencillo como simplemente terminar el desarrollo y comenzar a hacer pruebas.

Una última parte del Plan de Trabajo consistió en analizar los resultados obtenidos, ver si había resultados contradictorios y sacar conclusiones acerca de los objetivos que se propusieron en un principio.

## 1.6. Estructura de la memoria

- En primer lugar, nos encontraremos con una parte de *Introducción*. Aquí se hablará de las motivaciones del proyecto, objetivos y del plan de trabajo que se ha llevado a cabo.
- En el segundo punto nos encontraremos con *Trabajo relacionado*. Profundizaremos en todo el trabajo que se ha llevado a cabo sobre el estudio de esta área en distintos artículos de investigación.
- Seguidamente pasaremos a abordar el *Desarrollo de los proyectos* que han sido necesarios desarrollar para llevar a cabo los experimentos y veremos brevemente algunos conceptos sobre algoritmos genéticos.
- Yendo a la parte más importante de esta memoria, veremos una sección dedicada a los *Análisis de los experimentos* y a las *Conclusiones* obtenidas tras realizas las pruebas deseadas.
- Para finalizar, veremos algunas de las líneas sobre *Trabajo futuro* que han ido surgiendo a medida que se desarrollaba este Trabajo de Fin de Máster.

# 2. Trabajo relacionado

Con el avance tecnológico durante el último siglo, la asignación de recursos es un tema de preocupación en áreas como Ciencias de la Computación y Economía. Una de las primeras definiciones que surgieron sobre el problema de reparto clásico fueron los *problemas de bancarrota*. En este tipo de problemas interviene un conjunto de agentes $A = \{A_1, A_2, \ldots, A_n\}$ sobre los que debemos repartir una determinada cantidad $E \in IR_+$ de un bien divisible. Dichos agentes poseen unas preferencias que vienen representadas por el vector $\rho = \{\rho_1, \rho_2, \ldots, \rho_n\}$. Si para cada $i \in \{1, \ldots, n\}$, la cantidad $\rho_i$ representa la reclamación legítima del agente $Ai$ sobre la cantidad a repartir, y el total E no es suficiente para satisfacer a todos los agentes, esto es $\rho(A) = \sum_{i\in\{1,\ldots,n\}} \rho i > E$, entonces se trata de un *problema de bancarrota* [**7**]. En otras palabras: un problema de reparto clásico trata de determinar las cantidades que deben asignarse a cada agente de manera que la suma total sea la cantidad a repartir.

Como nombramos durante la **Introducción**, este campo de estudio suele denominarse como MARA. Existen excelentes trabajos y una amplia gama de aplicaciones que incluyen, entre otros, la contratación industrial [**8**], fabricación y programación [**9, 10, 11**], explotación eficiente de satélites de observación de la tierra [**12, 13**], gestión de tráfico aéreo [**14**], transporte público [**15**] y un largo etc. Cuando los problemas de asignación de recursos son definidos, en la mayoría de los casos no es suficiente con la formalización del problema clásico que hemos hecho anteriormente. En este aspecto, MARA podría definirse como "El proceso de distribuir un número de recursos entre un número de agentes". Sin embargo, es necesario matizar más esta definición: ¿Qué *tipo* de recursos se están distribuyendo? ¿Qué mecanismo de *asignación* estamos empleando? Y por último, ¿Cuáles son los objetivos de dicha asignación y cómo son determinados estos objetivos? [**4**].

## 2.1. Variaciones en Multiagent Resource Allocation

Existen una gran cantidad de variaciones que pueden ser consideradas a la hora de realizar un reparto de recursos. Algunas de estas variaciones dependen de factores tan importantes como, por ejemplo, la definición las características de los recursos y de los agentes, o el objetivo que se está persiguiendo a la hora de realizar el reparto [**4**].

### 2.1.1. Recursos

Si nos centramos en los recursos, han sido muchas las formas que se han estudiado para presentarlos. Por ejemplo, un recurso podría ser continuo (e.g. energía) o discreto (e.g. tornillos). Esta propiedad física influye directamente en la forma en la que el recurso es distribuido a los agentes. Otras características interesantes que han sido estudiadas son considerar si el recurso es divisible o no, si puede ser asignado a múltiples agentes a la vez o si es estático o dinámico (un recurso podría perecer con el tiempo o ser consumido por los agentes).

### 2.1.2. Agentes

En cuanto a los agentes, estos pueden participar en el reparto con diferentes criterios. Algunos estudios abordan el caso en el que ni siquiera existe la figura del repartidor, sino que este es eliminado, siendo los propios agentes quienes se encargan de la labor de repartir los recursos mediante interacciones locales entre sí [**16**]. Otra característica interesante podría ser la que nosotros proponemos: limitar mediante ciertas restricciones las preferencias que los agentes pueden establecer por los recursos disponibles.

### 2.1.3. Objetivo

El objetivo más típico en MARA es encontrar una asignación que sea óptima con respecto a una métrica que depende, de una manera u otra, de las preferencias de los agentes. Cuando se realiza el reparto, la suma de preferencias según la asignación de recursos puede modelarse utilizando típicamente nociones de Welfare Economics y Social Choice Theory. Existe mucha literatura acerca de todos estos conceptos, tal y como recoge M. Wooldridge haciendo una introducción a los sistemas multiagente [**17**] o T. W. Sandholm con un enfoque moderno a la Inteligencia Artificial Distribuida [**18**].

Por último, si vemos el problema del reparto desde una situación real, existen numerosas situaciones en nuestro día a día en las que podrían aplicarse los distintos modelos de repartos de recursos. Como hemos adelantado, para nosotros el objetivo que perseguiremos durante nuestros experimentos se centra en el modelo *Egalitarian*. Este modelo es comúnmente utilizado y podemos elaborar fácilmente una situación verídica en la que pueda ser aplicado.

Imaginemos que una catástrofe (e.g terremoto) azota el sur de Asia. Ante situaciones como esta, la Unión Europea dispone de un mecanismo de ayuda internacional denominado Mecanismo de Protección Civil. El objetivo es sencillo: minimizar el impacto del terremoto y enviar ayuda a las zonas afectadas. Si pensamos un momento sobre cómo deberíamos asignar la ayuda, enseguida nos damos cuenta de que se trata de un modelo *Egalitarian*, pues es necesario asignar un mayor número de recursos (personal sanitario y civil, comida, material sanitario, etc.) a las zonas más afectadas. De este modo, siempre perseguiremos realizar un reparto de recursos entre las zonas más afectadas, buscando que dicho reparto sea lo más eficiente posible (suponiendo que pudiéramos medir el grado de satisfacción que cada una de las zonas que reciben los recursos).

## 2.2. Algoritmos genéticos

Los algoritmos genéticos son métodos adaptativos que pueden usarse para resolver problemas de búsqueda y optimización. El fundamento de los mismos se basa en el proceso genético de los organismos vivos; a lo largo de las generaciones, las poblaciones evolucionan en la naturaleza acorde con los principios de selección natural y la supervivencia de los

mejor adaptados al medio. Por imitación a este proceso, los algoritmos genéticos son capaces de ir creando soluciones de tal forma que estas vayan hacia el valor óptimo.

Todo algoritmo genético necesita unos determinados ingredientes y pasar por ciertas fases para alcanzar esos valores óptimos de los que hemos hablado. A continuación, vamos a abordar qué es lo que vamos a tener en cuenta durante el desarrollo de nuestros algoritmos genéticos.

### 2.2.1. Funcionamiento básico

Este tipo de algoritmos trabajan con una población de individuos, cada uno de los cuales representa una solución factible a un problema dado. A cada individuo se le asigna un valor, relacionado con lo bueno que es ese individuo al ser tratado como solución. Este valor, a su vez, determina la probabilidad de que sea seleccionado para reproducirse, cruzando su "material genético" con otro individuo seleccionado de la misma forma. Este cruce producirá nuevos individuos descendientes de los anteriores (los cuales comparten algunas características de sus padres). De esta manera se produce una nueva población de posibles soluciones, la cual reemplaza a la anterior y verifica la interesante propiedad de que contiene una mayor proporción de buenas características en comparación con la población anterior. Así, a lo largo de las generaciones, las buenas características se propagan a través de la población. Si el algoritmo genético ha sido bien diseñado, la población convergerá hacia una solución razonablemente buena.

Observemos la **Ilustración 1**, en la que se muestran cuadrados divididos en cuatro partes. Imaginemos que cada cuadrado completo es un individuo y que cada una de sus cuatro partes se corresponde con un gen que lo conforma. Supongamos por un momento que para un caso concreto el objetivo es tener una población lo más variada posible, es decir: queremos obtener individuos cuyos genes tengan todas las tonalidades de azul distintas. Démonos cuenta de cómo a medida que la población evoluciona, el cruzamiento de la población hace que vayamos convergiendo hacia nuestro objetivo.

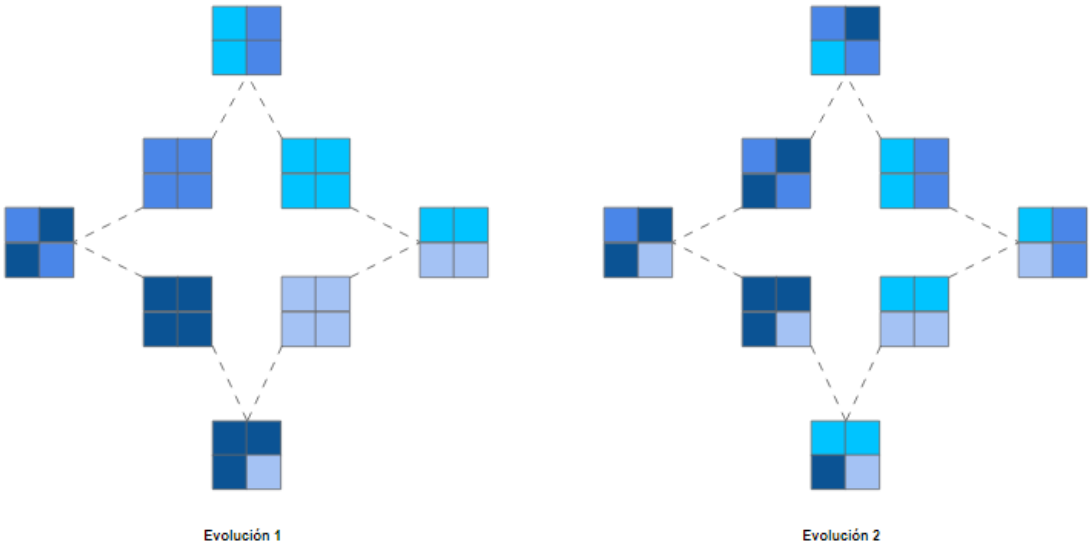


Ilustración 1: Evolución de un algoritmo genético

Durante la primera evolución, los hijos (cuadrados exteriores) ya tienen genes procedentes de sus dos padres. Esto seguirá ocurriendo cada vez que la población evolucione. Si prestamos atención a la segunda evolución de la población, observamos que uno de los individuos ya tiene tres de los cuatro colores que hay en total (individuo superior). Posiblemente este individuo, al mezclarse con el individuo de la derecha, obtenga en las próximas evoluciones los cuatro colores, logrando así el objetivo marcado.

Por último, conviene mencionar que el poder de los algoritmos genéticos proviene del hecho de que se trata de una técnica robusta, y pueden tratar con éxito una gran variedad de problemas provenientes de diferentes áreas, incluyendo aquellos en los que otros métodos encuentran dificultades.

### 2.2.2. Funcionamiento de los algoritmos genéticos

Vamos a ver brevemente qué metodología general siguen los algoritmos genéticos. En primer lugar, lo que se hace es crear una población inicial en lo que llamaremos fase de *Inicialización*. El tamaño de la población depende de la naturaleza del problema, pero normalmente contiene varias decenas, cientos o miles de posibles soluciones. A menudo, para dotar de gran variedad a la población, esta es generada aleatoriamente. Después viene la fase de computar la función de evaluación (en inglés denominada función *fitness*) de cada individuo. Como dijimos anteriormente, cada individuo necesita ser evaluado para saber cómo de bueno es respecto al resto de individuos de la población. A continuación, se procede a evolucionar la población, lo cual conlleva seis fases:

1. *Selección:* se seleccionan individuos de la población para hacer el cruce. Esta selección puede hacerse de manera aleatoria, por orden o como el implementador

crea conveniente. Es importante no dejar ningún individuo fuera de las opciones de selección, pues podría ser potencialmente valioso para la siguiente generación.

2. *Cruzamiento:* los individuos seleccionados se cruzan con cierta probabilidad (como vimos en **Ilustración 1**) y se crean nuevos individuos *hijos* que heredan características de sus padres.

3. *Cómputo de la función fitness:* se procede a calcular nuevamente cómo de valioso es cada individuo de la nueva población frente a los demás.

4. *Torneo:* una vez que todos los individuos de la nueva población ya tienen un valor fitness asociado, es hora de realizar el torneo en el que competirán con los individuos de la vieja población por parejas. Al final del torneo, los individuos de cada pareja con un mayor valor fitness serán los que sobrevivan a la siguiente generación.

5. *Mutación:* los hijos tienen una cierta probabilidad de mutar. Esto puede ser especialmente interesante porque, si un hijo ha tenido la mala suerte de que sus padres no eran "buenos", al mutar puede mejorar y llegar a ser competente en la siguiente evolución.

6. *Elitismo:* a pesar de ser opcional, la mayoría de las implementaciones suelen ser elitistas, lo que quiere decir que el mejor individuo de la población siempre sobrevivirá a la siguiente generación, lo cual supone una garantía importante y de ahí que se tenga en cuenta comúnmente. Al mejor individuo de una población suele denominársele *Fittest*.

En la mayoría de los casos, el implementador elige cómo se ejecutan estas seis fases, en las cuales podría haber pequeñas variaciones que, en principio, no deberían afectar a la obtención del individuo óptimo (como por ejemplo intercambiar las fases de *mutación* y *cruzamiento*).

```
BEGIN
    Inicialización de la población
    WHILE NOT Terminar DO
        FOR Tamaño de la población DO
            Selección
            Cruzamiento
            Fitness
            Torneo
            Mutación
        END FOR
        Elitismo
        IF la población ha convergido THEN
            Terminar := true
    END WHILE
END
```

Ilustración 2: Pseudo-código de un algoritmo genético canónico

Como puede verse en la **Ilustración 2**, la inicialización de la población tan solo se realiza al comenzar la ejecución del algoritmo. Después se llevan a cabo las diferentes fases que hemos visto, cuyas complejidades vienen determinadas por el número de individuos que tiene la población.

Conviene señalar que, en principio, el único factor que nos limita cuando queremos alcanzar el valor óptimo es el tiempo de ejecución, es decir, el tiempo que dejamos ejecutando el algoritmo genético. Asimismo, en nuestro caso el tiempo vendrá limitado por el número de veces que evoluciona la población, pudiéndose modificar este parámetro como deseemos. Cuando el algoritmo genético ha finalizado todas las iteraciones (o no, pero no queremos continuar ejecutándolo), tan solo debemos quedarnos con el mejor individuo de la población resultante que, como apuntamos con anterioridad, es nuestro individuo *fittest*.

Llegados a este punto ya tenemos una idea muy clara del flujo de ejecución que tendrán los algoritmos genéticos desarrollados en este TFM (aunque habrá pequeñas variaciones). De nuevo, téngase en cuenta que esta es la elección que hemos hecho, persiguiendo siempre la eficiencia en los experimentos que llevaremos a cabo.

### 2.2.3. Codificación

Pasamos al último de los apartados antes de entrar de lleno en la implementación que se ha hecho de los algoritmos: la codificación de un algoritmo genético. Los individuos en sí mismos no son más que su conjunto de características, es decir: no podríamos diferenciar dos individuos con las mismas características (aunque, como veremos, esto tiene un pequeño matiz). Cada una de estas características es lo que desde ahora en adelante denominaremos

como *gen*. Así pues, nuestra población no es más que un conjunto de individuos, los cuales están formados, a su vez, por una serie de genes.

Por otro lado, la función de *fitness* debe ser diseñada para cada problema de forma específica. Dado un individuo particular, la función normalmente suele asignar un número real que refleja el nivel de adaptación del individuo al problema. Además, durante la fase de cruzamiento, la selección debe hacerse considerando todos los individuos de la población y usando un procedimiento que favorezca a los mejores adaptados. En ningún momento debe existir un sesgo excesivo hacía los mejores individuos porque podría ocurrir que individuos que en conjunto no son relativamente buenos, contengan genes que sí lo son. Por eso no deben ser descartados a la ligera.

Una vez seleccionados dos individuos, sus genes se combinan para dar lugar a un nuevo hijo. La forma en la que se realiza el cruce es elegida por la persona que desarrolla el algoritmo, pero es común tener una probabilidad del 50% de escoger un gen del primer individuo y un 50% de escoger un gen del segundo individuo. En cualquier caso, existen otros métodos de cruzamiento como por ejemplo Shuffle crossover, Average crossover o Uniform crossover [**19**]. En este último, durante el cruce se cogen una serie de genes consecutivos (cuya longitud depende de un número aleatorio que indica los i-ésimos genes que formarán la serie) de ambos padres y se generan dos descendientes. Estos "pedazos" de genes serán incluidos en sus hijos de manera excluyente: si una serie de genes es asignada al hijo 1, entonces esos mismos genes no pueden ser asignados al hijo 2. En la **Ilustración 3** se muestra un ejemplo en el que se cogen series de dos genes de igual longitud.

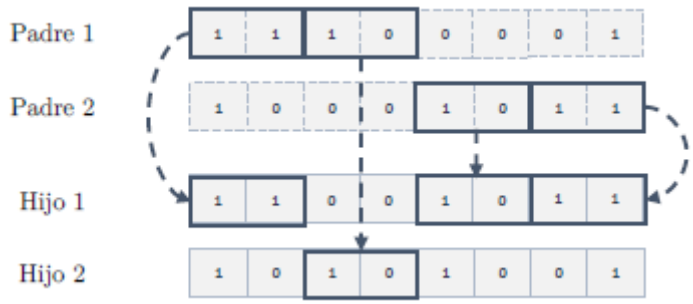


Ilustración 3: Ejemplo de Uniform crossover

Para finalizar con las fases vistas anteriormente, el operador de mutación se aplica a cada hijo de manera individual, y consiste en la alteración aleatoria (normalmente con probabilidad pequeña) de cada gen del individuo. Tras la mutación, el mejor individuo de la población es almacenado.

Si el algoritmo genético ha sido correctamente implementado, la población evolucionará a lo largo de las generaciones sucesivas de tal manera que la adaptación media extendida a

todos los individuos, así como la adaptación del mejor individuo, se irán incrementando hacia el óptimo global.

## 2.3. Distribución normal

Introducimos brevemente un tipo de distribución de probabilidad que utilizaremos en nuestros experimentos. Se llama distribución normal a una de las distribuciones de probabilidad de variable continua que más aparece en estadística y teoría de probabilidades. En esta distribución, la variable normal está completamente determinada por dos parámetros: su media y su desviación estándar, denotadas generalmente por $\mu$ y $\sigma$ respectivamente. Variando estos dos parámetros podremos generar, de manera aleatoria, valores cercanos a uno dado. Esto resultará especialmente interesante cuando queramos modificar la preferencia de un determinado agente de tal forma que los nuevos valores se mantengan "cercanos" al valor que tenía con anterioridad. E incluso, yendo un paso más allá, vamos a poder ir alejándonos de dicho valor tanto como queramos sin más que modificar la desviación estándar $\sigma$ de la distribución normal.

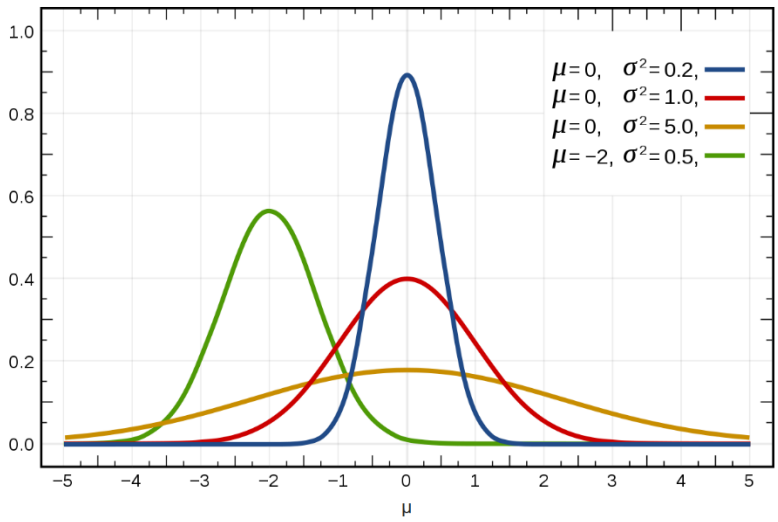


Ilustración 4: Variación de la media y desviación estándar

Fijémonos en que la media establece el punto medio desde el cual se calculan el resto de los valores de la distribución normal. Si modificamos la media, nuestra campana tomará un nuevo punto medio y la campana aparecerá desplazada hasta esa nueva media. Por otro lado, la desviación estándar nos dice la dispersión que tienen los valores que se encuentran dentro de la campana; cuanto menor desviación estándar, menor será la dispersión de valores. Tanto es así que, con una desviación estándar $\sigma = 0.0$, los valores no sufren ningún tipo de dispersión.

Pasemos a ver algunas de las características que más nos van a influir cuando trabajemos con ella durante los experimentos:

- La distribución normal es simétrica respecto de su media $\mu$.

- La moda y la mediana[2] son iguales a la media μ.
- Existen algunos intervalos interesantes en la distribución normal. Tan solo cabe destacar que dentro del intervalo [μ - 3σ, μ + 3σ] se encuentra comprendida, aproximadamente, el 99.74% de la distribución.

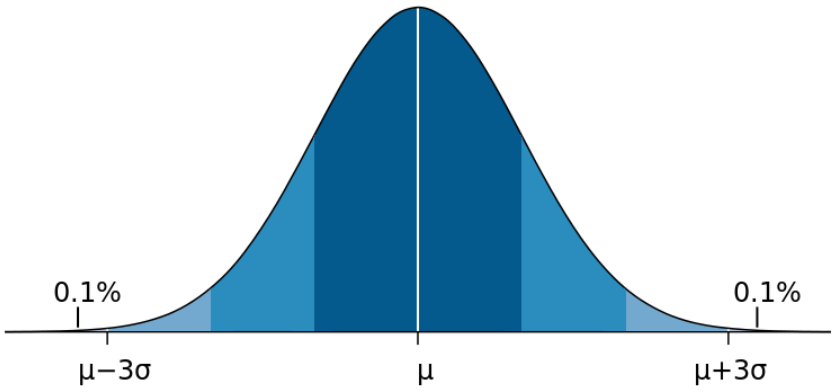


Ilustración 5: Intervalos en una distribución normal

- Aunque es obvio, conviene tener en cuenta que, al igual que todas las distribuciones no acotadas, la normal puede dar valores extremadamente pequeños y grandes con probabilidad prácticamente nula (aunque deberá tenerse en cuenta). Esto será especialmente importante en algunas pruebas que vamos a realizar.

[2] La moda es el número que se presenta con más frecuencia en un conjunto de datos. Por otro lado, la mediana se define como el valor que queda justo en medio cuando un conjunto de datos se ordena de menor a mayor.

# 3. Desarrollo de los proyectos

En las secciones anteriores hemos explicado en qué consiste el modelo *Egalitarian* sobre el que vamos a trabajar, e hicimos una aproximación a los algoritmos genéticos en la que vimos las características que definen cómo esta metodología de trabajo permite resolver determinados problemas obteniendo resultados considerablemente buenos. Si bien es cierto que hemos estudiado brevemente el esquema general que se suele seguir, también lo es el hecho de que, para cada problema concreto (hitos dentro de un mismo objetivo), una buena técnica es la de realizar un pequeño refinamiento al esquema general.

## 3.1. Proyectos utilizados con los algoritmos desarrollados

Nos gustaría hacer una sencilla aproximación y mostrar un esquema general de los proyectos Java implementados, y su relación con nuestros objetivos. En la **Sección 1.4** vimos que este Trabajo de Fin de Máster consta principalmente de tres objetivos. Para esquematizarlo de la mejor manera posible, hemos desarrollado tres proyectos Java (a pesar de que algunos tienen ciertas similitudes) para intentar cumplir esos tres objetivos.

Como hemos avanzado, el AGI (Algoritmo Genético Inferior) nos permitirá realizar experimentos para buscar repartos de recursos en los que se maximizará la utilidad del agente que menos utilidad reciba. Estos experimentos aportarán información sobre las estrategias que podría seguir el Agente 0 para, además, ver si existe alguna estrategia que siempre funcione independientemente de las preferencias que hayan establecido los demás agentes. El primero de los proyectos Java, llamado "Proyecto no aleatorizado", se encarga de esto: calcula las utilidades de todos los agentes cuando el Agente 0 ha mentido de diferentes formas.

Vistas las formas que tiene el Agente 0 de mentir, pasamos a perseguir un objetivo aún más ambicioso: hallar dinámicamente en cada caso la mentira óptima para el Agente 0 o, al menos, aproximarnos tanto como podamos. De nuevo, como adelantamos durante los objetivos, si el Agente 0 realiza una estimación de las preferencias de los demás agentes, ¿cuáles son las preferencias (en general, falsas) que deberá comunicar al repartidor para obtener la mayor utilidad posible? Encontrar estas preferencias es el objetivo de nuestro segundo proyecto, llamado "Proyecto AGS", el cual ejecuta el AGS (Algoritmo Genético Superior) que, a través de la evolución de su población, nos irá proporcionando cada vez mejores preferencias para el Agente 0.

Estamos viendo que los objetivos establecidos están estrechamente relacionados: primero vemos cómo el Agente 0 puede apañárselas para mentir y de qué formas predefinidas puede hacerlo. A continuación nos centramos en encontrar dinámicamente la mejor mentira contra unos rivales estimados por el Agente 0 y, por último, lo que haremos será corroborar cómo de buena es esa mentira. En particular, veremos qué ocurre cuando las preferencias del resto de los agentes son cada vez más diferentes a las que supuso el Agente 0. En este

caso, tal vez las preferencias que el AGS encontró para el Agente 0 no sean tan buenas porque han dejado de parecerse a las que él estimó. El último de los proyectos, llamado "Proyecto aleatorizado", es el que se encarga de, tomando como entrada las verdaderas preferencias del Agente 0, modificar las preferencias del resto de los agentes (con respecto a las estimadas por dicho agente) e ir calculando las utilidades verdaderas alcanzadas por el Agente 0 utilizando su supuesta mentira óptima, para comprobar cómo van variando. Aunque aún no hemos entrado en los detalles de cada algoritmo, todos los proyectos necesitan conocer qué reparto escogerá el repartidor para cada conjunto de preferencias recibidas, por lo tanto todos ellos harán uso del AGI.

Cabe destacar que el AGI puede trabajar tanto con preferencias no limitadas como con preferencias limitadas. De ahora en adelante esto es lo que se conocerá como Modo no limitado y Modo limitado. Luego como los tres proyectos hacen uso del AGI, podremos realizar experimentos en cualquiera de los dos modos. En la siguiente ilustración se muestran los tres proyectos Java de los que hemos hablado.

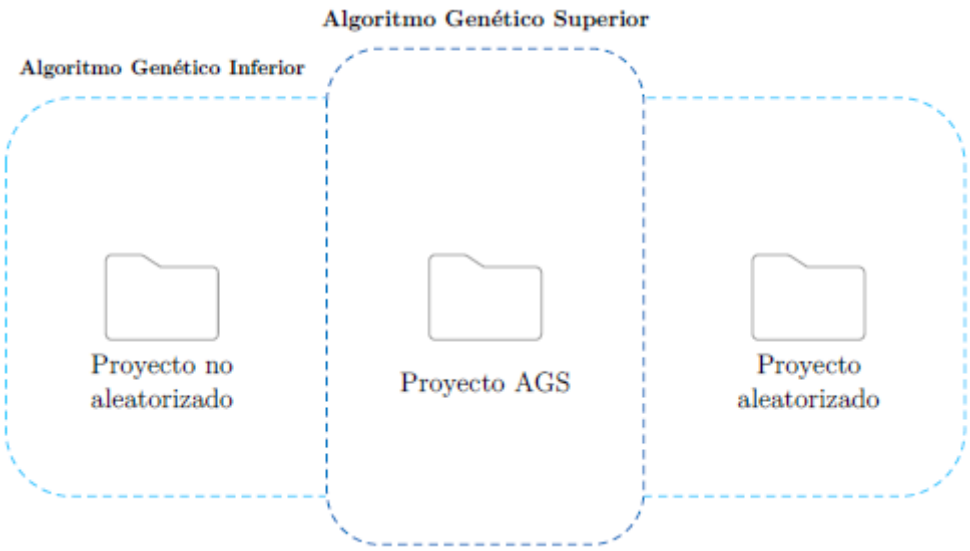


Ilustración 6: Estructura de proyectos Java

Nótese que, si queremos que el AGS encuentre la mejor mentira para el Agente 0, esa mentira tendrá una utilidad asociada, o mejor dicho, cada mentira que el AGS encuentre tendrá una utilidad asociada, que precisamente será la forma de cuantificar cómo de buena es. Por eso no es de extrañar que el AGS haga uso del AGI para que este le vaya proporcionando dichas utilidades.

### 3.1.1. Representación de la información

A continuación, pasamos a hablar sobre cómo vamos a representar toda la información que necesitan los algoritmos para su ejecución.

Para nosotros, cada individuo es únicamente un vector y cada posición del vector es un gen. Todos estos individuos forman la población con la que irá iterando el algoritmo genético en cuestión. Continuando con esta idea, debemos apuntar que, aunque en un primer momento la preferencia de los agentes y los individuos se almacenan en ficheros, cuando el programa carga los datos y comienza a ejecutarse, dicha información se vuelca en un vector de vectores (e.d. se carga en memoria). De esta forma disponemos de una sola estructura de datos en la que tenemos un fácil acceso a cada individuo de la población y, a su vez, a cada gen.

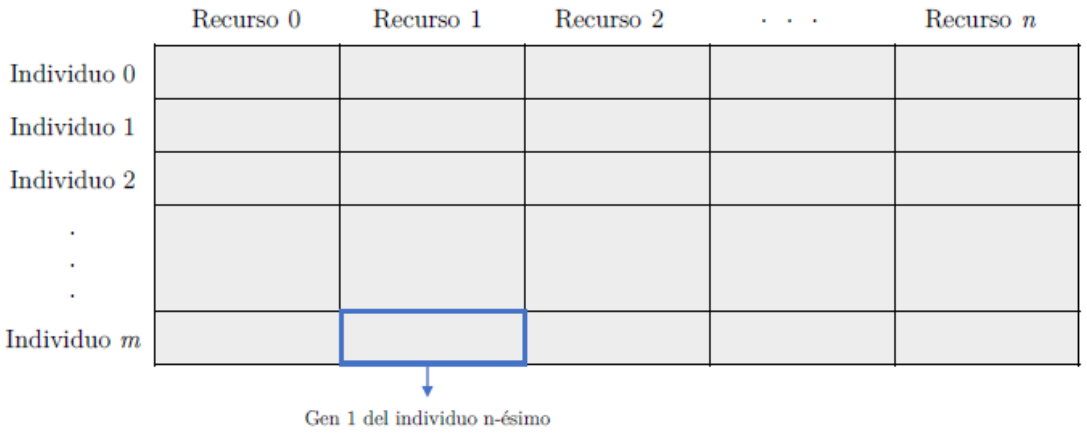


Tabla 5: Representación en ejecución de los individuos del AGI

Nótese que, en el AGI, cada individuo representará un posible reparto de recursos entre los agentes. Por el contrario, los individuos del AGS serán posibles preferencias (en general falsas) del Agente 0 hacia los recursos.

La representación de las preferencias de los agentes es totalmente análoga a la representación de los individuos; el programador que decida realizar pruebas puede elegir cuántos agentes desea que haya. Dicha información también será almacenada en ficheros para ser posteriormente volcada en un vector de vectores. Nótese que la longitud de los vectores de los individuos siempre será igual a la longitud de los vectores de las preferencias, pues los individuos son soluciones de $n$ recursos y las preferencias son establecidas también sobre esos mismos $n$ recursos.

| | Preferencia por recurso 0 | Preferencia por recurso 1 | Preferencia por recurso 2 | . . . | Preferencia por recurso $n$ |
|---|---|---|---|---|---|
| Agente 0 | | | | | |
| Agente 1 | | | | | |
| Agente 2 | | | | | |
| . . . | | | | | |
| Agente $m$ | | | | | |

Tabla 6: Representación en ejecución de las preferencias de los agentes

### 3.1.2. Fases de los algoritmos genéticos desarrollados

Profundicemos un poco más sobre cómo se llevan a cabo las diferentes fases por las que pasan los algoritmos. Si bien es cierto que son diferentes, los pasos que siguen durante su ejecución son similares (aunque el objetivo que se esté persiguiendo sea ligeramente distinto). Aun así, vamos a ver las diferencias y cómo plasmamos todo eso visualmente.

Partimos de la situación en la que la población ha sido inicializada y se ha hecho un primer cálculo del fitness asociado a cada individuo. Asimismo, se ha almacenado el *fittest*, ya que se trata de una implementación elitista.

1. *Selección*: se comienza realizando una selección aleatoria de dos individuos entre toda la población actual. Esta selección se realizará tantas veces como el tamaño de población que tengamos, para que en cada evolución de la población se conserve el número de individuos.

2. *Cruzamiento*: a continuación, se crea un nuevo individuo hijo[3] y, para cada uno de sus genes, existe un 50% de probabilidad de que dicho gen sea heredado del primer individuo y un 50% de ser heredado del segundo individuo (individuos padres). Una vez creado el nuevo individuo, es almacenado para usarse durante la siguiente evolución.

[3] Llamaremos así al individuo resultante del cruzamiento entre dos individuos de la "antigua" población.

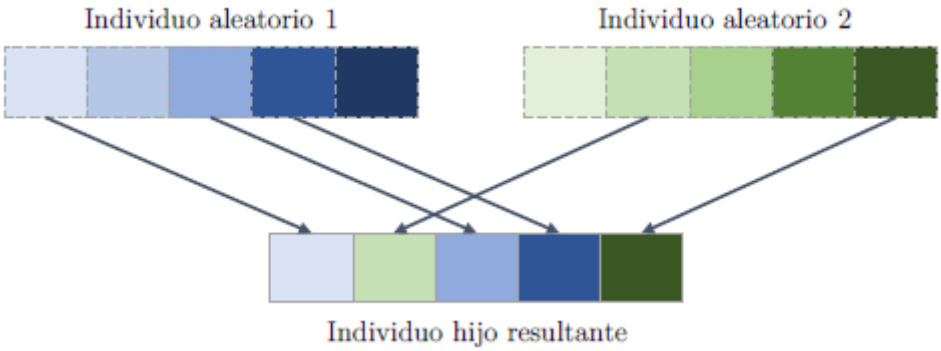


Ilustración 7: Fase de cruzamiento

3. *Cómputo de la función fitness*: habiendo creado todos los individuos de la nueva población, es necesario calcular sus valores fitness para poder evaluarlos y que sean comparables al resto de individuos de la población.

4. *Torneo*: con las dos poblaciones listas (una formada por los padres y otra por los hijos), es hora de realizar el torneo en el que competirán para tratar de sobrevivir hasta, al menos, la siguiente generación. El torneo planteado escoge parejas (por orden) entre la vieja y la nueva población, es decir: compite mejor individuo de la vieja población contra el mejor individuo de la nueva población. Después se hace exactamente lo mismo para el resto de los individuos, sobreviviendo únicamente el mejor de cada pareja. Podría pensarse que esto crea un sesgo en la población, pero en realidad no; aunque en este torneo se elijan las parejas de manera *ad hoc*, en la fase de cruzamiento la elección se ha hecho de forma totalmente aleatoria (así como los genes heredados por cada uno de los hijos).

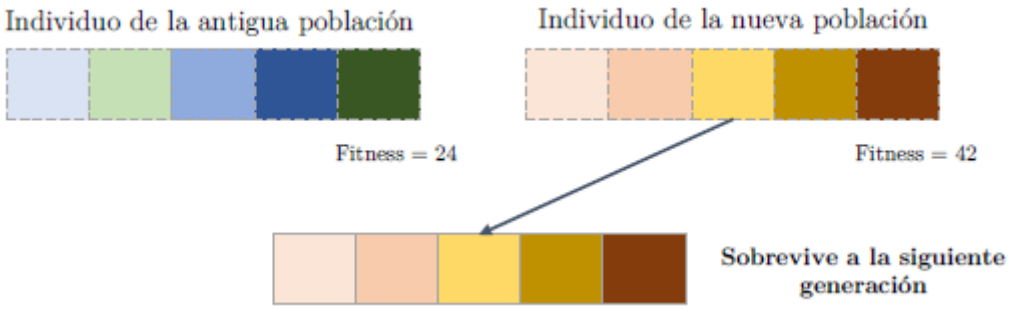


Ilustración 8: Fase de torneo

Supongamos que, tal y como vemos en la ilustración, cogemos el individuo resultante del cruzamiento anterior. En la fase de torneo éste competirá con individuos de la nueva población, sobreviviendo únicamente aquel que tenga un *fitness* mayor.

5. *Mutación*: esta fase se lleva a cabo recorriendo todos los genes de todos los individuos y mutándolos con una cierta probabilidad. Es importante que dicha

probabilidad no sea muy alta porque los individuos podrían cambiar demasiado, lo cual echaría por tierra el trabajo realizado por las anteriores fases. Al mismo tiempo, tampoco debe ser prácticamente cero porque ligaría mucho a los hijos con sus padres, sin posibilidad de que haya más variedad durante las generaciones. La forma en la que muta un gen viene denotada por sumar (o restar) un porcentaje al valor que tuviese. Dicho porcentaje se establece dentro de un intervalo (predefinido) y es elegido de manera aleatoria. De esta manera podemos elegir cuánto queremos que cambie un gen (respecto a su valor original) sin más que ampliar el intervalo.


Ilustración 9: Fase de mutación

En esta ilustración se refleja la forma en la que un gen muta. Nótese que el nuevo valor, como hemos dicho, es un valor generado aleatoriamente al calcular un porcentaje aleatorio dentro de un intervalo. Puede variar poco, tal y como ilustra el último gen, o mucho, como en los genes en la posición 2 y 4 (de ahí que tengan un color completamente distinto).

6. *Elitismo*: por último, se procede a almacenar el individuo *fittest* (aquel con el *fitness* más alto) en el primer lugar de la población.

Hasta aquí hemos visto los rasgos comunes que presentan los dos algoritmos. Como dijimos anteriormente, los dos algoritmos, aunque relacionados, tienen objetivos completamente diferentes. Pasemos a ver en detalle en qué consisten cada uno de estos objetivos y cómo los algoritmos han sido diseñados e implementados.

## 3.2. Algoritmo Genético Inferior

Para llevar a cabo el desarrollo del AGI, que fue el primero en ser diseñado e implementado, fueron necesarias algunas pruebas que tomaron las primeras semanas del tiempo dedicado a este TFM. Téngase en cuenta que no solo se trata de diseñar e implementar un algoritmo genético acorde al problema de maximizar el beneficio del agente con menos utilidad, sino que, una vez implementado, hay que realizar numerosas pruebas (algunas ellas tomaron varios días en ser ejecutadas) para corroborar que el algoritmo está funcionando correctamente. En otras palabras: es necesario testearlo y lograr que sea lo

más eficiente posible. Eso es lo que se busca; que, si el valor devuelto por el algoritmo no es el óptimo, al menos nos hayamos aproximado todo lo posible con el tiempo del que dispusimos.

### 3.2.1. Objetivo

El problema que resuelve este algoritmo es el de, dadas las preferencias de $n$ agentes por una cantidad $m$ de recursos, encontrar una distribución de recursos tal que la menor utilidad recibida (considerando las de todos los agentes) sea máxima. Este es precisamente el objetivo del modelo *Egalitarian* que nombramos al principio.

Teniendo claro el objetivo, las posibles distribuciones de recursos son soluciones al problema y, a su vez, individuos de la población que maneja el AGI. Recordemos que en nuestro modelo los recursos son indivisibles, es decir, no podemos asignar medio recurso a un agente y el resto a otro agente distinto.

### 3.2.2. Persiguiendo el Fittest en el AGI

¿Cómo logra realmente el algoritmo alcanzar valores cada vez mejores y converger hacia el valor óptimo? Como vimos en las secciones anteriores, cada vez que la población del algoritmo evoluciona, este almacena el individuo con el mayor *fitness* alcanzado (mejor solución encontrada hasta el momento). ¿Bastaría con devolver dicha solución tras, digamos, $n$ evoluciones del algoritmo? Tal vez, pero podemos hacerlo mejor. Debido a la aleatoriedad implícita de cualquier algoritmo genético, si tras esas $n$ evoluciones de la población simplemente nos quedamos con el *fittest* sobreviviente, ¿tenemos certeza de que ese fittest es verdaderamente el óptimo? Es más, ¿tenemos siquiera certeza de que es un valor cercano al óptimo? Ciertamente no; no tenemos ninguna certeza. La solución propuesta es hacer que el AGI se ejecute 2.000 veces y quedarnos con el mejor resultado obtenido de esas 2.000 ejecuciones independientes.

Como avanzamos en los objetivos de este Trabajo de Fin de Máster, nos encontramos con dos tipos de reparto sobre los que queremos realizar pruebas. En el primer caso queremos averiguar qué ocurre cuando los agentes pueden establecer una preferencia por cada recurso que, aun estando dentro de un intervalo, puede ser la que cada agente considere. Y el segundo caso, obligamos a los agentes a que todas las preferencias establecidas deban sumar una determinada constante. En la siguiente ilustración puede apreciarse un pequeño ejemplo en el que se ven las preferencias de un determinado agente sobre tres recursos. En la primera fila no nos importa la suma de las preferencias, tan solo que estén comprendidas dentro del intervalo abierto (0,100) (hemos decido establecer el intervalo abierto para que las preferencias de cada agente por algún recurso sean siempre mayores que 0). En cambio, si nos fijamos en la segunda fila, podemos ver cómo, además de estar cada valor dentro del mismo intervalo, también se cumple que la suma total de las tres preferencias es igual a 100.

| | Preferencia por recurso 0 | Preferencia por recurso 1 | Preferencia por recurso 2 |
|---|---|---|---|
| Agente | 42.0 | 21.3 | 96.4 |
| Agente | 30.5 | 50.3 | 19.2 |

Ilustración 10: Comparación de preferencias entre el modo no limitado y limitado

Hemos dicho que se han implementado dos algoritmos en total, lo cual es cierto. Pero para el AGI hay dos versiones que, si bien podrían haberse implementado todas en un solo proyecto Java, hemos optado por separarlas. ¿La razón? Aunque hay mucho código en común, cada versión persigue un objetivo distinto y, por una mayor comodidad, decidimos separar las dos versiones en dos proyectos distintos (los cuales se corresponden con los que vimos en las secciones anteriores). No queríamos mezclar los datos de las pruebas realizadas y que estuvieran todas en un solo proyecto.

### 3.2.3. No aleatorizado

Esta versión es la que únicamente se encarga de cumplir el objetivo del modelo *Egalitarian*. El AGI simplemente calcula las utilidades de todos los agentes y, cada vez que evoluciona la población, trata de buscar soluciones al reparto de recursos aún mejores (para el agente menos beneficiado). En nuestras pruebas, las preferencias que tienen los agentes por cada uno de los recursos son generadas de manera aleatoria. Cabe destacar que, puesto que las preferencias son leídas desde ficheros de texto (y posteriormente cargadas en memoria), un programador que desee realizar experimentos con unas determinadas preferencias podrá hacerlo sin más que modificar los ficheros desde los que se hace la lectura.

### 3.2.4. Aleatorizado

Como segunda versión tenemos el AGI aleatorizado. La diferencia es la manera en la que se modifican las preferencias. Recordemos que este proyecto se utiliza una vez que ya hemos ejecutado el AGS para el Agente 0 y así, teniendo una mentira considerablemente aceptable, queremos comprobar qué ocurre cuando las preferencias del resto de los agentes difieren cada vez más de las que el Agente 0 estimó. Para ir realizando esta modificación en las preferencias de los demás agentes, lo que hace esta versión es aplicar una distribución normal a la preferencia que cada agente diferente al Agente 0 tiene por cada recurso, tomando como media la normal la preferencia que el Agente 0 estimó que el agente correspondiente tendría por el recurso). De esta forma podemos ir aplicando distribuciones normales en las que la desviación sea cada vez más grande e ir analizando qué ocurre con la utilidad que va obteniendo el Agente 0. Así, podremos observar si las mentiras obtenidas por el AGS, pretendidamente óptimas para el Agente 0 cuando este tiene información

perfecta sobre las preferencias de sus rivales, es más o menos robusta ante la imprecisión de dicha información.

A modo de ejemplo, la siguiente ilustración muestra un caso real preparado en el que se han establecido unas determinadas preferencias para un agente genérico. Las filas superiores muestran las preferencias de dicho agente que podrían haber sido estimadas por el Agente 0, mientras que las inferiores representan las preferencias después de aplicar la distribución normal, para representar la imprecisión en la información del Agente 0. Si nos fijamos en la parte izquierda, vemos que los valores apenas cambian, pues es relativamente probable que los nuevos valores generados estén muy cerca de los originales, ya que la desviación es apenas de 1.0 en ese caso. Por el contrario, en los resultados mostrados en la parte derecha observamos una mayor dispersión de los nuevos valores obtenidos, donde la desviación es 10.

| | Preferencia por recurso 0 | Preferencia por recurso 1 | Preferencia por recurso 2 |
|---|---|---|---|
| Agente | 42.0 | 21.3 | 96.4 |
| Agente | 42.4 | 21.8 | 95.7 |

**Desviación σ = 1.0**

| | Preferencia por recurso 0 | Preferencia por recurso 1 | Preferencia por recurso 2 |
|---|---|---|---|
| Agente | 42.0 | 21.3 | 96.4 |
| Agente | 34.2 | 23.6 | 86.8 |

**Desviación σ = 10.0**

Ilustración 11: Comparación de valores normales para los dos tipos de preferencias

Queremos dar importancia a un hecho que no debe pasar desapercibido y del que es fácil percatarse, ¿qué ocurre si los nuevos valores generados por la distribución normal hacen que la suma en el modo limitado no sea exactamente la constante que se estableció? Y por otro lado: ¿qué ocurre si, por ejemplo, la desviación es muy alta y se genera un valor de preferencia que se sale del rango establecido? Este proyecto tiene programados algunos métodos para controlar este tipo de anomalías.

En el primer caso, el programa se asegura de que las preferencias modificadas sigan sumando la constante establecida normalizando el vector de preferencias de cada agente, es decir: nos aseguramos de que todas las preferencias para cada agente sigan sumando lo mismo pero habiéndose incrementado/decrementado cada una de ellas de manera relativa. Por otro lado, para evitar que los valores generados por la distribución se salgan del rango establecido, lo que se hace es, cada vez que se genera un valor, se comprueba si está dentro del rango. Si no es así, se vuelve a generar el valor tantas veces como sea necesario hasta que caiga dentro del intervalo. Se ha verificado que estas comprobaciones ni siquiera ralentizan de forma notable el tiempo de ejecución del algoritmo.

Por último, queremos mencionar que estas dos versiones del AGI son capaces de trabajar tanto en el modo de preferencias no limitadas como en el modo de preferencias limitadas. Existen dos métodos encargados de generar las preferencias de todos los agentes y

escribirlas en ficheros de texto. Con lo cual, dependiendo del modo en el que se esté trabajando se llamará a un método u otro antes de comenzar con la ejecución del AGI en sí.

## 3.3. Algoritmo Genético Superior

Pasemos a estudiar el que posiblemente sea el segundo de los pilares de este TFM: el Algoritmo Genético Superior. Recordemos que este algoritmo tan solo es utilizado en el *Proyecto AGS* (ver **Ilustración 6**). Aun así, el uso que se le da es muy alto, sobre todo en los experimentos realizados haciendo uso de la distribución normal. Al igual que ocurría con el AGI, el diseño e implementación de este algoritmo no fue tarea fácil. Incluso nos atreveríamos a decir que fue bastante más complejo que programar el AGI por sí solo (por razones que veremos enseguida).

Antes de entrar de lleno en el objetivo del AGS, quisiéramos destacar que un cambio a tener en cuenta respecto al AGI es la representación de los individuos (de nuevo, posibles soluciones al problema). Estudiamos cuando introdujimos el AGI que los individuos de la población representan soluciones al problema de la distribución de recursos que debe hacer el subastador. Ahora, en el AGS, cada individuo representa una posible asignación de preferencias de cierto agente (el Agente 0) hacia los recursos de los que se dispone. Es decir: los genes que forman cada uno de los individuos de la población ya no son números enteros que representan a los agentes, sino que ahora son números reales que representan las preferencias que tiene cierto agente por cada uno de los recursos. En la siguiente ilustración hacemos una comparativa entre los individuos que forman la población en el AGI y los individuos de la población del AGS, todos ellos con cuatro genes de longitud.

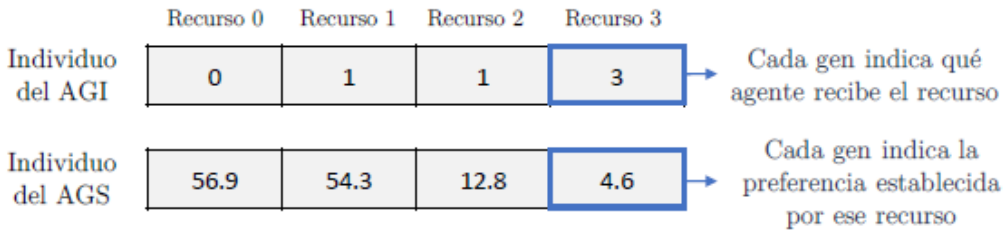


Ilustración 12: Diferencia entre individuos del AGI y AGS

### 3.3.1. Objetivo

En contraste al objetivo perseguido por el AGI, el cual tenía como misión encontrar una distribución de recursos tal que se maximice la menor utilidad recibida por los agentes, el objetivo esta vez es encontrar las mejores preferencias que deberían ser enviadas al repartidor para que, a la hora de realizar el reparto, la utilidad obtenida por el Agente 0 (con sus preferencias reales) sea lo más alta posible. En otras palabras: no es más que responder a aquella pregunta que nos hicimos al principio sobre cuál era la mejor manera de mentir que tiene el Agente 0 tal que su beneficio sea el mayor posible. Démonos cuenta

de que esto es un caso que en principio no tiene nada que ver con nuestro mecanismo de reparto, en el sentido de que, durante el reparto, un agente tan solo conoce sus preferencias y ninguna otra (por lo tanto, lo único que puede hacer el Agente 0 es estimar cuáles serán las preferencias de los demás agentes).

Para tratar de encontrar unas preferencias tales que, después de haber realizado el reparto, nos reporten una mayor utilidad a cierto agente $k$ (idealmente, la mayor de todas), debemos tener en cuenta las preferencias de todos los agentes, o al menos una estimación de ellas. Y con dicha información, tan solo debemos ajustar las preferencias (probablemente falsas) que el agente $k$ comunicará al repartidor para obtener, en el hipotético reparto que se comunicase, utilidades cada vez mayores. Encontrar esas preferencias, como hemos dicho, es el objetivo del AGS.

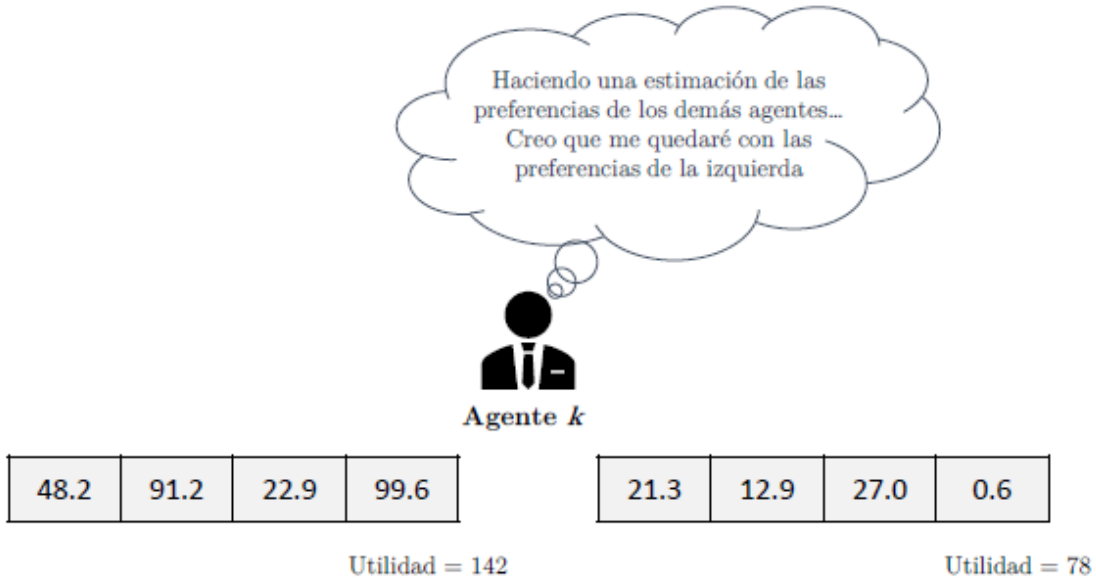


Ilustración 13: Buscando las mejores preferencias con el AGS

Pero este objetivo no será una tarea sencilla para el AGS. No lo será en tiempo ni en la heurística que debe seguir para ir mejorando las preferencias con cada una de las evoluciones de la población.

### 3.3.2. Persiguiendo el Fittest en el AGS

Como ya sabemos, lo primero que hace cualquiera de los algoritmos que manejamos es generar unas determinadas preferencias para los agentes (las preferencias comunicadas al repartidor) y la población. Tal y como hemos avanzado, en este caso cada individuo de la población es, a su vez, una configuración de preferencias generalmente falsas del Agente 0 por los recursos disponibles. Recordemos que esta población se genera de manera aleatoria,

pero poco a poco, con cada una de las evoluciones, los individuos tenderán a parecerse cada vez más a las preferencias óptimas[4], tal y como vimos en la **Ilustración 1**.

Cuando el AGS calcula los valores *fitness* de sus individuos, lo que hace es tener en cuenta la utilidad que tendrían cada uno de ellos cuando se realiza el reparto de recursos. Es decir: por cada individuo de la población (nuevas preferencias del Agente 0 a evaluar), el AGS debe llamar al AGI para que este le proporcione el resultado que supondría resolver nuestra subasta bajo el modelo *Egalitarian* usando dichas preferencias. En efecto, estamos hablando de que calcular la función *fitness* de cada individuo AGS supone invocar el propio AGI (de ahí la complejidad de lograr nuestro objetivo).

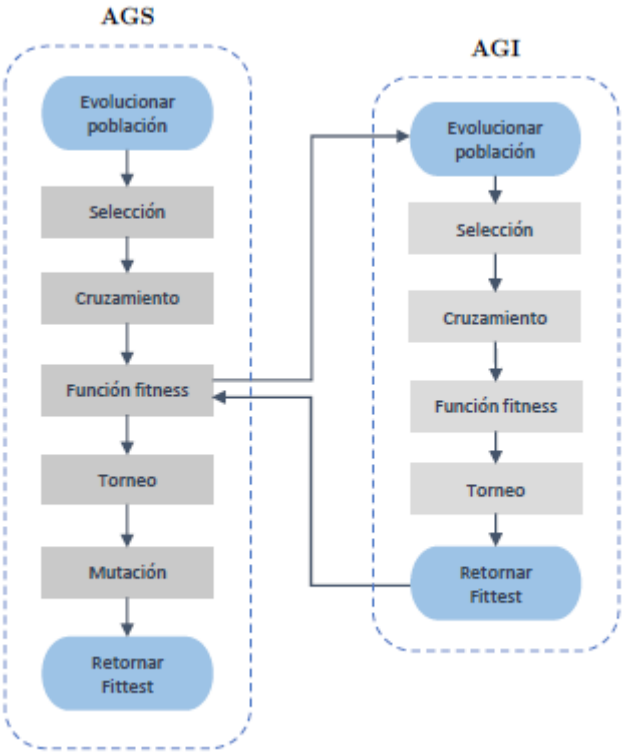


Ilustración 14: Flujo de ejecución del Proyecto 2

Cabe destacar que al AGS no le afecta el modo en el que estemos trabajando, con respecto a, si las preferencias están limitadas o sin limitar. Lo único que le interesa es generar preferencias, evolucionarlas e ir modificándolas (pasando por las fases ya vistas) para así descubrir conjuntos de preferencias que den utilidades cada vez mayores. En este TFM se analiza qué ocurre cuando un solo agente (el Agente 0) sigue la estrategia y trata de "engañar" al resto de agentes para aumentar su utilidad. Realizar un estudio para ver qué ocurre cuando $n$ agentes tratan de seguir la estrategia de mentir, frente a los que no la siguen, es una de las líneas que queda fuera del Trabajo de Fin de Máster pero que se estudiará en futuros trabajos.

[4] Denotamos así a las preferencias con las que un agente puede llegar a conseguir la mayor utilidad posible una vez que el subastador realiza la distribución de recursos.

# 4. Análisis de los experimentos

Durante esta sección vamos a proceder a analizar los experimentos que se han llevado a cabo. El objetivo de estos experimentos es, precisamente, ofrecer respuestas a las preguntas que surgieron en la sección **Objetivos**, al comienzo de esta memoria. Cabe destacar que no solo se han hecho los experimentos descritos aquí; hay ciertas pruebas que optaremos por dejar fuera de esta memoria ya que prácticamente no aportan información acerca de los análisis hechos. Tanto las pruebas restantes como las utilidades logradas por todos los agentes pueden ser consultadas[5].

De ahora en adelante, los experimentos se centrarán únicamente en el Agente 0, que es el agente sobre el que queremos maximizar la utilidad obtenida. La forma en la que los hemos estructurado es muy sencilla: acorde con los objetivos propuestos, en una primera sección a la que llamaremos "Análisis de las pruebas AGI", analizaremos algunas de las formas que tiene el Agente 0 para mentir, tanto en el modo no limitado como en el modo limitado. Durante la segunda sección, llamada "Buscando la preferencia óptima", intentaremos encontrar unas preferencias que permitan al Agente 0, haciendo una previa estimación de las preferencias del resto de los agentes, encontrar la mejor mentira posible, es decir, aquellas preferencias por los recursos que al ser comunicadas al repartidor le aporten una mayor utilidad conforme a sus preferencias verdaderas (de nuevo, esto lo haremos tanto en el modo no limitado como en el modo limitado). Por último, en la tercera sección llamada "Análisis aplicando una distribución normal", veremos qué ocurre cuando las estimaciones que en un principio hizo el Agente 0 sobre las preferencias de los demás agentes, no son exactamente las que él pensó. Además, estudiaremos qué ocurre con la utilidad del Agente 0 cuando dicha estimación difiere cada vez más de las preferencias que realmente establecieron el resto de los agentes.

## 4.1. Análisis de las pruebas AGI

Pasamos a las pruebas en las que únicamente interviene el AGI. El objetivo de estos análisis es crear algunas formas sencillas en las que el Agente 0 podría mentir, es decir: vamos a modificar las preferencias del Agente 0 (sin tener en cuenta las preferencias de los demás agentes) de diferentes maneras para tratar de buscar mentiras lo más eficientes posibles. ¿Cómo afectarán estas mentiras a la distribución de recursos hecha por el repartidor? La siguiente tabla recoge las distintas pruebas que se han llevado a cabo.

| | |
|---|---|
| **P1** | Alza los cuatro primeros recursos |
| **P2** | Baja los cuatro primeros recursos |
| **P3** | Alza los ocho primeros recursos |
| **P4** | Baja los ocho primeros recursos |
| **P5** | Alza los cuatro recursos más valorados entre el resto de los agentes |
| **P6** | Baja los cuatro recursos más valorados entre el resto de los agentes |

[5] Ver el documento adjunto a la memoria.

| | |
|---|---|
| **P7** | Alza los cuatro recursos menos valorados entre el resto de los agentes |
| **P8** | Baja los cuatro recursos menos valorados entre el resto de los agentes |
| **P9** | P5 con los ocho recursos más valorados entre el resto de los agentes |
| **P10** | P6 con los ocho recursos más valorados entre el resto de los agentes |
| **P11** | P7 con los ocho recursos menos valorados entre el resto de los agentes |
| **P12** | P8 con los ocho recursos menos valorados entre el resto de los agentes |
| **P13** | Alza los cuatro recursos que más valora él |
| **P14** | Baja los cuatro recursos que más valora él |
| **P15** | Alza los cuatro recursos que menos valora él |
| **P16** | Baja los cuatro recursos que menos valora él |
| **P17** | P13 con los ocho recursos que más valora él |
| **P18** | P14 con los ocho recursos que más valora él |
| **P19** | P15 con los ocho recursos que menos valora él |
| **P20** | P16 con los ocho recursos que menos valora él |

Tabla 7: Experimentos hechos solo con el AGI

Como hemos dicho, la manera de abordar las pruebas será ejecutando repetidamente el AGI a la vez que vamos modificando las preferencias del Agente 0 conforme al plan de la prueba correspondiente. Estas preferencias irán aumentando o disminuyendo un determinado porcentaje establecido previamente. Así, cada vez que las preferencias se modifican, resolvemos el problema del reparto y obtenemos la utilidad que le corresponde al Agente 0 con la solución encontrada al ejecutar el AGI.

Antes de pasar a ver los resultados, nos gustaría explicar cómo vamos a organizar los análisis. En primer lugar, recordemos que estas pruebas se centrarán exclusivamente en el AGI, y veremos el caso en el que consideramos preferencias no limitadas y limitadas. Asimismo, las pruebas se han llevado a cabo en dos escenarios distintos (tanto en el modo no limitado como en el modo limitado), variando el número de agentes disponibles. Luego, además de ver algunas de las pruebas de la **Tabla 7** en cada uno de los escenarios, también contrastaremos los resultados más interesantes entre los distintos escenarios. La siguiente tabla recoge estos dos escenarios y sus diferencias.

| Escenario | Número de agentes | Número de recursos |
|---|---|---|
| Uno | 4 | 10 |
| Dos | 8 | 10 |

Tabla 8: Variación de agentes disponibles en cada escenario

Para cada uno de estos dos escenarios, salvo el número de agentes, el resto de los parámetros se han mantenido siempre igual en cada una de las veinte pruebas. En la siguiente tabla se recogen los valores de estos parámetros. Queremos volver a recordar un detalle muy importante: el AGI, como cualquier otro algoritmo genético, cada vez que se ejecuta puede encontrar una utilidad distinta (aunque parecida) para el Agente 0. Esta aleatoriedad está intrínseca en el algoritmo, pues depende de todas las fases que atraviesa cada vez que la población evoluciona y, de hecho, esta capacidad de encontrar distintos

caminos también es una parte importante del éxito de los algoritmos genéticos. Dicho esto, pensamos que ejecutar el AGI una sola vez cada vez que realizamos alguna modificación de las preferencias podría dar lugar a resultados que no sean del todo fiables a la hora de sacar conclusiones, así que optamos por ejecutar el AGI 2.000 veces. A continuación se coge la mejor solución obtenida (la solución que más maximice la utilidad del agente que recibe una menor utilidad) y, con dicha solución (reparto de recursos), se calcula la ventaja que proporciona al Agente 0 mentir. Este cálculo se realiza restando la utilidad que recibe el Agente 0 considerando un reparto basado en sus preferencias reales a la utilidad que recibe el Agente 0 considerando un reparto basado en sus preferencias modificadas (nótese que tan solo se consideran unas preferencias u otras a la hora de realizar el reparto, no cuando se debe calcular la utilidad que le corresponde, donde siempre se utilizan las preferencias reales). La **Ilustración 15** aclara la manera en la que se realiza el cálculo de utilidad.

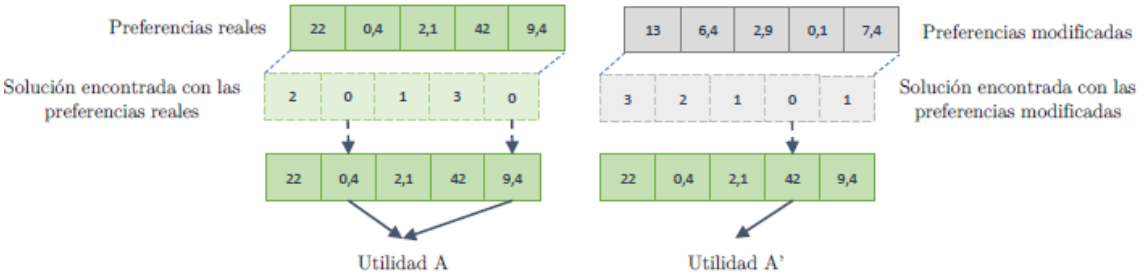


Ilustración 15: Cálculo de la utilidad en las pruebas AGI

Como vemos en este sencillo ejemplo, el Agente 0 ha obtenido una utilidad A, con sus preferencias reales, de 9,8 puntos, mientras que usando sus preferencias modificadas obtuvo una utilidad A' de 42 puntos. En nuestros experimentos, el valor que aparecería sería 42 – 9,8 = 32,2. Esto nos indica que fue beneficioso para el Agente 0 el hecho de haber modificado sus preferencias reales.

Puesto que la utilidad que se resta podría ser mayor que la obtenida al llevar a cabo una mala estrategia, no es de extrañar que en las siguientes gráficas aparezcan utilidades con valores negativos. Esto no quiere decir que la utilidad obtenida sea negativa, únicamente refleja que la utilidad alcanzada por una determinada estrategia es peor que la utilidad lograda al usar las preferencias reales (es decir, sin mentir).

La siguiente tabla recoge los parámetros más importantes que se han mantenido inalterados durante las pruebas en cada uno de los escenarios y para cada uno de los dos modos de funcionamiento.

| Parámetro | Valor |
|---|---|
| Tamaño de la población | 100 |
| Número de veces que evoluciona la población en cada ejecución del AGI | 50 |
| Número de veces que se ejecuta el AGI por cada modificación de preferencias | 2.000 |

Tabla 9: Configuración de los parámetros generales del AGI

Una última aclaración de cara a las gráficas que veremos a continuación es que las pruebas, según sean al alza o a la baja, tienen un color determinado. Hemos establecido usar colores en tonos naranjas para las pruebas que modifican las preferencias al alza y colores en tonos azules para las pruebas que modifican los valores a la baja. Y a su vez, tonos claros para las pruebas que únicamente modifican cuatro recursos y tonos oscuros para las pruebas que modifican ocho recursos.

### 4.1.1. Modo no limitado

Si pasamos directamente al caso en el que estamos considerando preferencias no limitadas, el Agente 0 podrá mentir sobre ellas con la única condición de que el valor asignado a cada preferencia por cada recurso se encuentre dentro del intervalo (0, 100). Para este caso, hemos considerado que, en cada muestra[6], las preferencias que se estén modificando, tanto al alza como a la baja, aumenten/disminuyan un 2% del valor que les quedan por crecer/decrecer, es decir, del intervalo que falta hasta llegar a 100 (en el caso de que estemos incrementándolas) o del intervalo que falta hasta llegar a 0 (en el caso de que estemos decrementándolas). De este modo, podemos ir acercándonos a 100 tanto como queramos. Al mismo tiempo, el número de muestras que se han recogido es de cien (e.d. se han modificado cien veces los recursos en cada una de las pruebas). Después de realizar algunas pruebas previas, creemos que 2% es un valor aceptable con el que ir modificando los valores de las preferencias.

En la **Tabla 10** se recogen las preferencias que han sido establecidas para los agentes de cara a cada uno de los diez recursos dentro del Escenario 1. La última fila, "Media", recoge el valor medio de las preferencias para los agentes 1, 2 y 3 (obviando las del Agente 0).

| Preferencias en el Escenario 1 | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Recurso** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| **Agente 0** | 42,364 | 19,185 | 75,3776 | 42,327 | 60,208 | 68,986 | 85,308 | 20,668 | 31,671 | 60,415 |
| **Agente 1** | 53,703 | 96,730 | 70,873 | 68,473 | 86,954 | 51,349 | 3,0645 | 30,017 | 55,039 | 45,423 |
| **Agente 2** | 88,492 | 17,881 | 70,7305 | 98,714 | 30,422 | 40,718 | 98,415 | 85,195 | 42,088 | 57,059 |
| **Agente 3** | 95,089 | 50,577 | 69,314 | 26,018 | 56,030 | 64,098 | 55,086 | 6,623 | 49,242 | 31,690 |
| **Media** | 79,095 | 55,069 | 70,306 | 64,402 | 57,802 | 52,055 | 52,188 | 40,612 | 48,790 | 44,724 |

Tabla 10: Preferencias en modo no limitado, Escenario 1

[6] Denotamos muestra a cada uno de los valores (utilidad del Agente 0) que obtenemos cada vez que modificamos sus preferencias y ejecutamos el AGI 2.000 veces, quedándonos con el valor más alto retornado.

Si nos fijamos en la **Ilustración 16**, observamos la utilidad obtenida por el Agente 0 durante las pruebas 1, 2, 3 y 4 (pruebas en las que únicamente se modifican los recursos de manera secuencial). Como vemos, se han tomado un total de 100 muestras, tal y como indica el eje horizontal. A medida que avanzamos en las muestras incrementando o disminuyendo el valor de las preferencias por los recursos correspondientes en cada prueba, la utilidad obtenida varía. Esta variación se vuelve más notable en las pruebas 3 y 4 debido a que en estas dos pruebas se modifican ocho de los diez recursos disponibles. Si observamos la utilidad de las pruebas parece claro que, en estos primeros experimentos, al Agente 0 no le conviene mentir alzando el valor de sus preferencias; la utilidad cae considerablemente a medida que avanzan las muestras en las pruebas 1 y 3 (en mayor medida según tomamos más recursos y según disminuimos cada vez más el valor de las preferencias modificadas). Por otro lado, las pruebas en las que el Agente 0 modifica sus preferencias a la baja sí parecen ser una buena estrategia: si bien es cierto que la Prueba 2 no aporta gran información, si nos fijamos en su análoga con ocho recursos observaremos que la utilidad comienza a crecer a medida que avanzamos en las muestras.

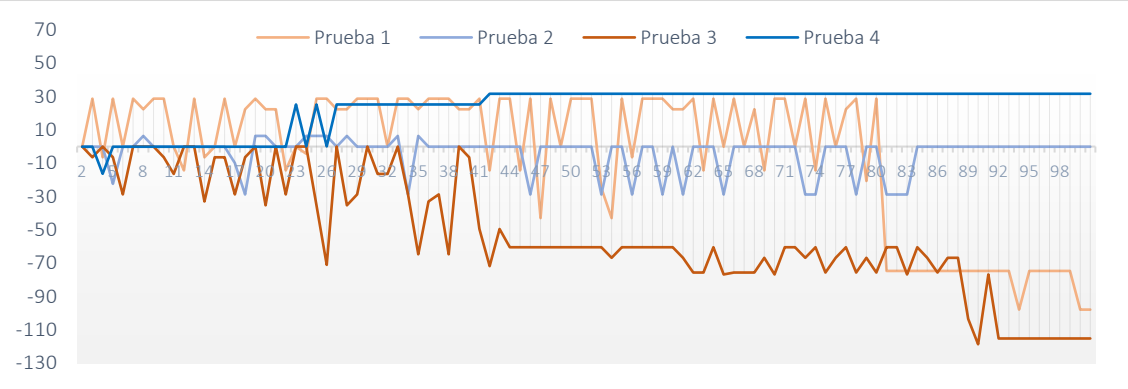


Ilustración 16: Pruebas 1, 2, 3 y 4. Escenario 1. Modo no limitado

Si pasamos a ver qué ocurre cuando el Agente 0 modifica las preferencias de los cuatro recursos más/menos valorados entre los demás agentes, volvemos a observar que incrementar el valor de las preferencias no le conviene en ninguno de los casos, sino que más bien mentir a la baja sigue siendo la mejor opción. Aunque en estas cuatro pruebas únicamente estemos considerando cuatro recursos, llama la atención que se obtienen valores más extremos cuando consideramos los recursos menos valorados entre el resto de los agentes (pruebas 7 y 8) que cuando consideramos los recursos más valorados entre el resto de los agentes (pruebas 5 y 6).

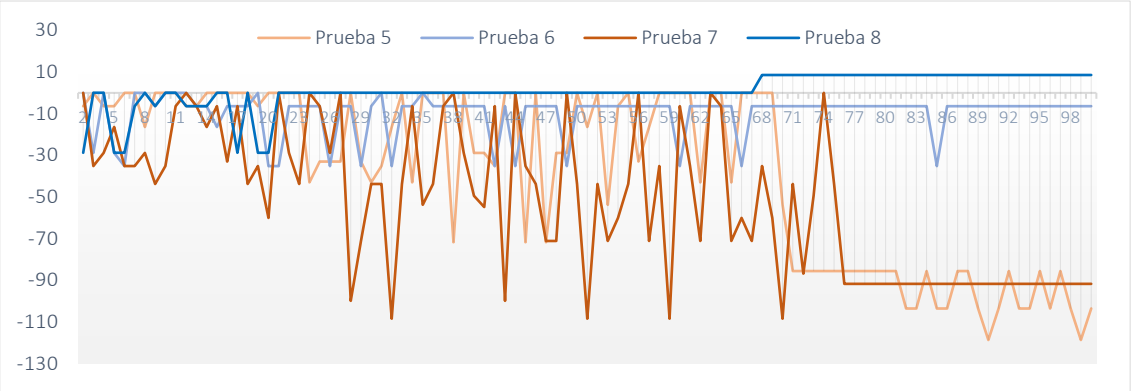


Ilustración 17: Pruebas 5, 6, 7 y 8. Escenario 1. Modo no limitado

Las conclusiones obtenidas durante estas pruebas se ven notablemente reforzadas cuando consideramos ocho recursos en vez de solo cuatro. Las pruebas 9, 10, 11 y 12[7] indican que, por el momento, la mejor estrategia a seguir por el Agente 0 es mentir haciendo ver a los demás que no le interesan los recursos que ellos valoran en menor medida.

Continuando con las pruebas, vemos en la **Ilustración 18** modificados los cuatro recursos que más/menos valora el Agente 0. De nuevo, en cualquiera de los dos casos, mentir al alza es una mala estrategia a seguir. Sin embargo, vemos que la utilidad siguiendo la estrategia de disminuir sus preferencias tampoco parece funcionar durante estas pruebas, únicamente al comienzo de la Prueba 14, aunque después de eso, la utilidad cae considerablemente.

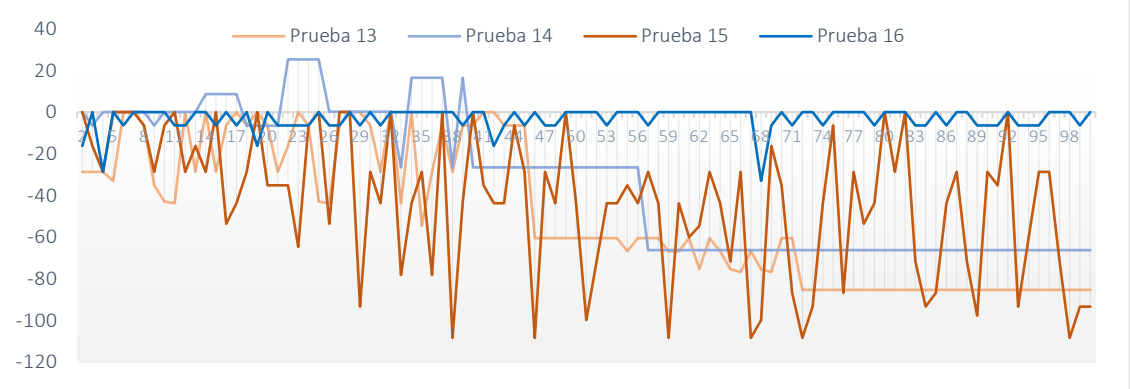


Ilustración 18: Pruebas 13, 14, 15, y 16. Escenario 1. Modo no limitado

Cabe destacar la caída sufrida en la Prueba 14 y analizarla en detalle.

En general, tras ver algunas de las estrategias para mentir que podría llevar a cabo el Agente 0, podemos afirmar con cierta confianza que considerar al alza las preferencias por los recursos (en cualquiera de las pruebas) no es una buena estrategia a seguir. Incrementar sus preferencias por los recursos le ha supuesto ir perdiendo cada vez más utilidad a medida que avanzaba el número de muestras. Así que, consecuentemente, mentir al alza sobre un

[7] Los resultados de estas y el resto de las pruebas pueden ser consultados en el enlace proporcionado al comienzo de la sección **Análisis de los experimentos**.

menor número de recursos sería lo más "beneficioso" para él. Además, mentir al alza sobre los recursos más valorados (tanto por él mismo como por el resto de los agentes) parece ser, dentro de las estrategias al alza, la menos mala de todas ellas.

Por el contrario, en cualquiera de las pruebas en las que se han ido decrementado las preferencias por los recursos, el Agente 0 ha obtenido una mayor utilidad (o, al menos, su utilidad no se ha visto perjudicada). Parece que, cuantos más recursos considere para mentir a la baja sobre sus preferencias, una mayor utilidad obtendrá. Asimismo, como estamos en el caso de preferencias no limitadas, el Agente 0 puede disminuirlas a su antojo siempre que el valor final sea mayor a cero. Si pensamos por un momento en estos resultados, disminuir las preferencias parece la mejor estrategia a seguir; si el Agente 0 hace ver al repartidor que todos los recursos disponibles le interesan muy poco, entonces el repartidor, para cumplir con su objetivo (maximizar la utilidad del agente que menos utilidad recibe) deberá asignar prácticamente todos los recursos al Agente 0.

Por último, es importante destacar que al repartidor no le conviene asignar todos los recursos disponibles al Agente 0. Si así lo hiciese, el resto de los agentes tendrían una utilidad de 0, luego pasarían a ser los más desfavorecidos (y, por lo tanto, los nuevos objetivos del repartidor para maximizar sus utilidades). Dicho esto, es lógico pensar que para cualquiera de las soluciones que encuentre el AGI, en todas ellas al menos algún recurso será asignado a cada uno del resto de los agentes, incluso cuando el Agente 0 utiliza las estrategias vistas anteriormente.

### 4.1.2. Modo limitado

Pasemos al caso en el que el Agente 0 ya no podrá disminuir el valor de sus preferencias tanto como desee. ¿Qué ocurrirá ahora? ¿Seguirá siendo beneficioso disminuir solo las preferencias por ciertos recursos a cambio de aumentar las de otros para conservar la restricción en el modo limitado? Las pruebas hechas en el modo limitado son las mismas (la elección de los recursos a modificar se mantiene).

Esta vez, algunas de las pruebas no llegarán a las cien muestras. Nótese que si estamos modificando las preferencias al alza para, por ejemplo, los primeros $m$ recursos, para compensar debemos decrementar el valor de los siguientes $n–m$ recursos. En teoría, al igual que ocurría en el modo no limitado, siempre podemos ir aumentando el valor de esos primeros $m$ recursos y acercarnos a 100 tanto como queramos. Ahora bien, la restricción añadida impide que podamos tener valores negativos en los $n–m$ recursos. Por esa razón, durante algunas pruebas observaremos que el número de muestras es considerablemente pequeño (debido a que el AGI se detiene cuando ya no podemos continuar decrementando los $n–m$ recursos).

Análogamente, presentamos las preferencias del Escenario 1 que han sido generadas para todos los agentes al comienzo de la ejecución.

| Preferencias en el Escenario 1 | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Recurso** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| **Agente 0** | 17,677 | 12,586 | 4,354 | 12,006 | 5,472 | 0,771 | 14,571 | 3,499 | 16,556 | 12,504 |
| **Agente 1** | 1,927 | 6,099 | 19,666 | 18,177 | 10,515 | 10,757 | 2,425 | 3,310 | 23,046 | 4,073 |
| **Agente 2** | 16,957 | 12,247 | 11,647 | 7,330 | 7,636 | 12,574 | 9,878 | 8,735 | 11,331 | 1,661 |
| **Agente 3** | 14,415 | 2,203 | 16,789 | 1,032 | 13,158 | 9,702 | 5,182 | 5,468 | 17,193 | 14,853 |
| **Media** | 11,100 | 6,850 | 16,034 | 8,847 | 10,436 | 11,011 | 5,828 | 5,838 | 17,190 | 6,862 |

Ilustración 19: Preferencias en el modo limitado, Escenario 1

Si nos fijamos en los valores generados, al tener la restricción de que la suma de las preferencias de cada agente debe ser 100, es lógico que ahora los valores sean relativamente menores que en el caso anterior. Esto tiene un impacto directo en el porcentaje que modificamos en las preferencias de los recursos cada vez que las incrementamos/decrementamos. Habiendo hecho algunas pruebas previas, hemos optado por modificar el porcentaje que teníamos en el modo no limitado. Esta vez, al trabajar en el modo limitado, hemos establecido que cada preferencia se modifique 1% al alza y 3% a la baja. El razonamiento es muy sencillo: al generarse las preferencias con un valor más bajo, cuando las estemos incrementando, queremos que lo hagan lentamente (pues el intervalo es mayor). Por consiguiente, cuando estemos decrementando su valor, el intervalo será más pequeño, por eso hemos aumentado el porcentaje de decremento respecto al alza.

Si pasamos a las primeras cuatro pruebas, las cuales se muestras en la **Ilustración 20**, ninguna de las formas de mentir que funcionaban en el modo no limitado funciona. Es más, parece que mentir haciendo creer al repartidor que los cuatro primeros recursos no le interesan al Agente 0, ha pasado a ser una mala estrategia a seguir, pues su utilidad cae considerablemente. Incrementar las preferencias tampoco aporta una mejora en la utilidad del Agente 0.

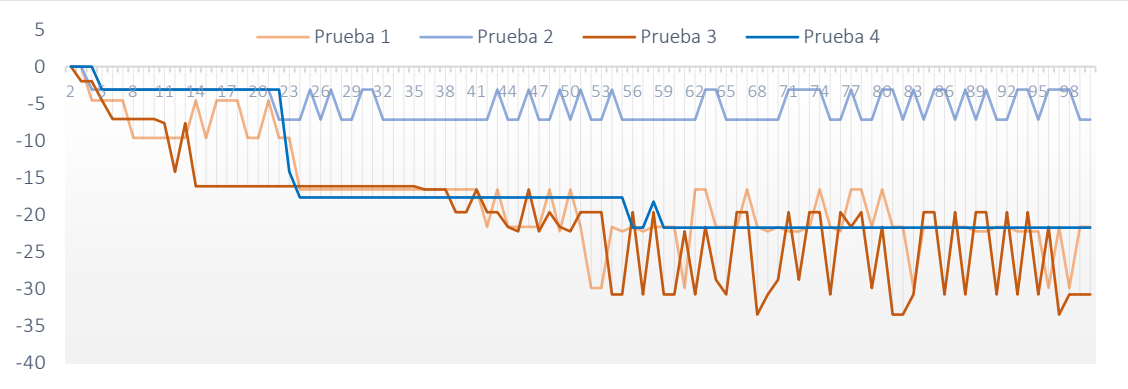


Ilustración 20: Pruebas 1, 2, 3 y 4. Escenario 1. Modo limitado

Durante las Pruebas 5, 6, 7 y 8, tampoco parece funcionar ninguna de las estrategias. En todos los casos, la utilidad obtenida a la larga cae (caída que se ve mucho más pronunciada durante las pruebas 9, 10, 11 y 12, cuando consideramos ocho recursos en vez de cuatro). Quizá, por resaltar una pequeña anomalía, si nos fijamos en la Prueba 6 podemos ver que la utilidad sube ligeramente durante las primeras muestras. Pequeñas variaciones, por qué no, podrían incluso hacer que la utilidad asignada al Agente 0 mejore. Sin embargo, cuando estas variaciones son llevadas al extremo (lo que podríamos considerar una estrategia propiamente dicha), vemos un resultado mucho más evidente.

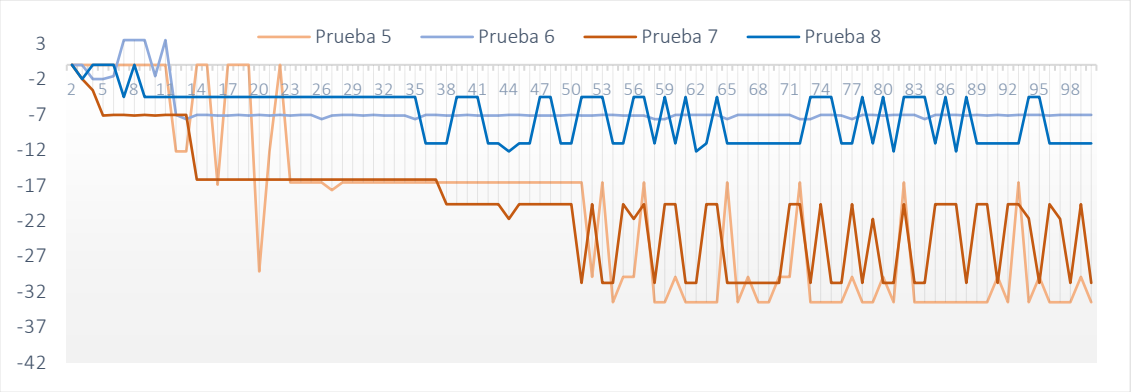


Ilustración 21: Pruebas 5, 6, 7 y 8. Escenario 1. Modo limitado

En la última gráfica, al igual que ocurría en la Prueba 5, la Prueba 15 presenta una ligera mejoría al comienzo de las muestras, pero a medida que estas avanzan, el resultado muestra una clara evidencia de que esta estrategia tampoco es buena. En general, la estrategia de modificar los recursos que más/menos valora el Agente 0, y que tan bien funcionaba cuando estábamos trabajando con el modo no limitado, ahora tampoco sirve.

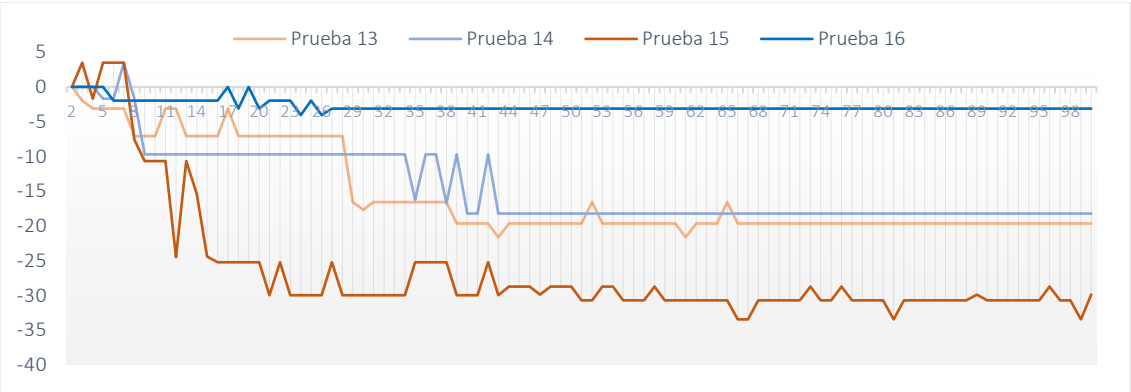


Ilustración 22: Pruebas 13, 14, 15 y 16. Escenario 1. Modo limitado

Hemos visto que, cuando estamos en el caso de preferencias limitadas, parece no haber ninguna estrategia (al menos de las que aquí hemos considerado) que funcione. Ninguna de las estrategias estudiadas aporta beneficio al Agente 0, para el cual, la mejor opción es simplemente no mentir al comunicar sus preferencias al repartidor.

¿Estos resultados quieren decir que en el modo limitado no existe ninguna estrategia que de manera general permita al Agente 0 obtener una mejor utilidad? No. Estos resultados indican que ninguna de las estrategias consideradas funciona, tan solo eso. Para contestar a la pregunta que acabamos de lanzar es necesario dar un paso más allá. De alguna manera, necesitamos encontrar mentiras que cada vez sean mejores, pero llevar a cabo esta tarea no parece que vaya a ser sencillo. Si nos damos cuenta, existen muchas maneras de modificar las preferencias, así como el porcentaje de modificación establecido para cada una de las muestras. Por esta razón, haremos que el AGS intente encontrar (o aproximarnos tanto como podamos) las mejores preferencias para el Agente 0 en cada caso, es decir: aquellas con las que obtenga una mayor utilidad.

## 4.2. Buscando la preferencia óptima

En la sección anterior hemos tratado algunas de las estrategias predefinidas que puede seguir el Agente 0 para mentir engañando al resto de los agentes e intentar lograr una mayor utilidad. Algunas de estas estrategias son relativamente buenas y otras no tanto, sobre todo cuando nos encontramos en el modo limitado. Ahora bien, yendo un paso más allá, ¿qué valores de preferencia deberá comunicar el Agente 0 para lograr la mayor utilidad posible en cada caso? Si recordamos, este es el segundo de los objetivos que estamos persiguiendo.

Pasemos a realizar los análisis de los experimentos hechos con el AGS (Algoritmo Genético Superior). Recordemos que la misión del AGS es encontrar una serie de preferencias (en nuestro caso las preferencias correspondientes al Agente 0) que le permitan obtener una mayor utilidad cuando el Agente 0 comunica al repartidor tales preferencias, asumiendo que las preferencias que comunicarán los demás agentes son ciertas preferencias estimadas para ellos por el propio Agente 0. Antes de continuar conviene saber de dónde proviene esta utilidad del Agente 0, es decir, cómo se calcula.

Cuando hablamos de que la misión del AGS es encontrar una serie de preferencias que proporcionen una mayor utilidad al Agente 0, esta utilidad no se calcula basándonos en la valoración que otorgarían, precisamente, dichas preferencias óptimas[8]. Las preferencias óptimas únicamente se utilizan para lograr una mejor distribución de recursos para el Agente 0, pero al final, la utilidad del Agente 0 se calcula teniendo en cuenta sus preferencias reales.

[8] Denotaremos así a las preferencias encontradas por el AGS, las cuales reportan una utilidad mayor o igual al Agente 0 que sus preferencias reales, es decir, las preferencias que hubiese comunicado sin haber realizado ningún tipo de estimación sobre los demás agentes.

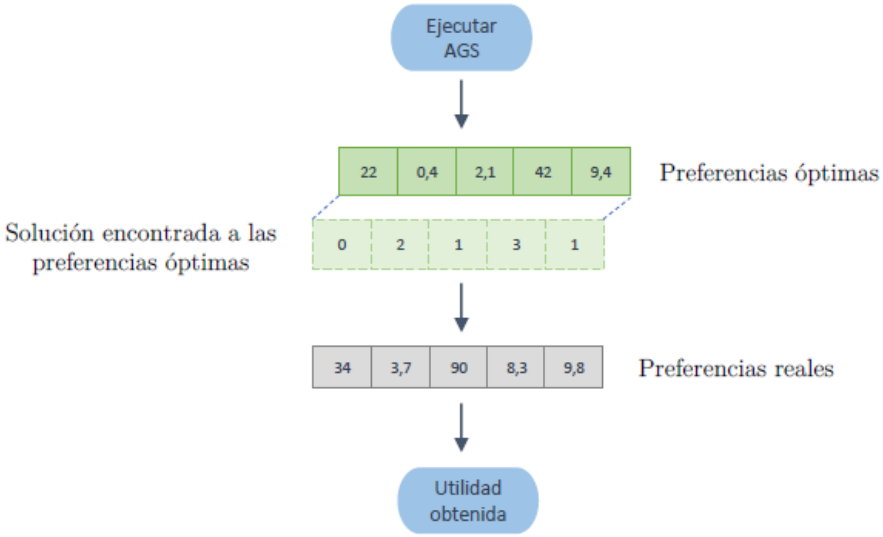


Ilustración 23: Cálculo de la utilidad con el AGS

Si nos fijamos en la **Ilustración 23**, al ejecutar el AGS, este nos proporciona una serie de preferencias óptimas para los recursos disponibles. Estas preferencias tienen asociada (como es natural) una solución, es decir, se corresponden con una distribución de reparto de recursos. Con esta distribución y las preferencias *reales* del Agente 0, lo que se hace es calcular su utilidad correspondiente. Insistimos en que no se usan las preferencias óptimas para realizar el cálculo; estas solo sirven para encontrar una mejor distribución del reparto de recursos (siempre desde el punto de vista de que dicha distribución sea la más beneficiosa para el Agente 0 con sus verdaderas preferencias).

Como ya sabemos, el tiempo que emplea el AGS en encontrar preferencias que proporcionen al Agente 0 una mejor utilidad es considerablemente alto debido a que, para cada uno de los individuos de su población, el AGS debe hacer una llamada al AGI (que se ejecutará completamente) para que este le retorne el valor *fitness* de dicho individuo. Durante el estudio de los resultados con el AGI, vimos que para cada muestra ejecutábamos el AGI 2.000 veces y, de todas esas ejecuciones, al final nos quedábamos con el resultado más alto retornado. Tras probar con diversos experimentos en los que hemos ido variando los parámetros del AGS y del AGI, se ha comprobado que esta vez logran alcanzarse resultados satisfactorios con "tan solo" 400 ejecuciones completas del AGI para el cálculo del valor fitness de cada individuo del AGS. Esto implica que, si tenemos una población en el AGS de, por ejemplo, 50 individuos, para que dicha población evolucione una sola vez será necesario llamar al AGI 50 veces (lo que conlleva 20.000 ejecuciones completas del AGI).

El resto de los parámetros del AGI son los que vimos en la **Tabla 9**. Por otro lado, los parámetros del AGS para el Escenario 1 son los que observamos en la siguiente tabla:

| Parámetro | Valor |
|---|---|
| Número de agentes | 4 |
| Número de recursos disponibles | 10 |
| Tamaño de la población del AGS | 50 |
| Número de veces que evoluciona la población del AGS | indefinido |

Tabla 11: Configuración de los parámetros generales del AGS

Con estos parámetros, cada una de las evoluciones de la población del AGS tomaba en torno a 4 minutos. El número de veces que evoluciona la población está marcado como *indefinido* porque no tiene sentido acotarlo demasiado, aunque para los experimentos hemos tomado 800 evoluciones. Es cierto que lo hemos dejado en un valor arbitrariamente alto, pero pensemos por un momento que siempre que dispongamos de tiempo suficiente, podremos dejar el AGS ejecutándose para que cada vez continúe mejorando su población, lo que proporcionaría unas mejores preferencias para el Agente 0. Estas 800 evoluciones han tomado en torno a 40 horas de ejecución.

A continuación, vamos a ver cómo para el AGS es considerablemente más difícil encontrar unas preferencias que aporten una mayor utilidad cuando estamos trabajando en el modo limitado (lo cual era de esperar), ya que el Agente 0 no puede simplemente disminuir la preferencia de cada recurso a su antojo.

### 4.2.1. Modo no limitado

En primer lugar, partimos de la utilidad que el Agente 0 alcanzó cuando comunicó sus preferencias reales (es decir, no mentir) al realizar el reparto de recursos, que fue de 221,101 (esto es, simplemente ejecutar el AGI con sus preferencias reales). Tanto estas preferencias como las de los demás agentes pueden observarse en la siguiente tabla.

| | Recurso 0 | Recurso 1 | Recurso 2 | Recurso 3 | Recurso 4 | Recurso 5 | Recurso 6 | Recurso 7 | Recurso 8 | Recurso 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Óptimo** | 1,236106107 | 4,30185429 | 7,7246844 | 21,4005234 | 27,5123777 | 7,65814814 | 13,5606997 | 1,39471891 | 7,85270647 | 3,63572758 |
| **Agente 0** | 42,364775 | 19,185787 | 75,377697 | 42,327327 | 60,208991 | 68,986514 | 85,308145 | 20,668722 | 31,67128 | 60,415511 |
| **Agente 1** | 53,703774 | 96,730491 | 70,873802 | 68,473613 | 86,954077 | 51,349186 | 3,0645272 | 30,017284 | 55,039713 | 45,423049 |
| **Agente 2** | 88,492241 | 17,88123 | 70,730566 | 98,714908 | 30,42223 | 40,718859 | 98,415448 | 85,195807 | 42,088602 | 57,059793 |
| **Agente 3** | 95,089597 | 50,57722 | 69,314898 | 26,018095 | 56,030841 | 64,098374 | 55,086135 | 6,6232624 | 49,242516 | 31,690532 |

Tabla 12: Preferencias en el modo no limitado para el AGS

La fila *Óptimo* indica las preferencias que ha encontrado el AGS después de evolucionar su población 800 veces. Si nos fijamos, llama la atención que en todos los recursos disponibles el AGS ha optado por disminuir el valor de cada preferencia respecto a las preferencias reales que tenía el Agente 0. Recordemos que durante las pruebas con el AGI parecía razonable afirmar que una buena estrategia a seguir en el modo no limitado es sencillamente disminuir las preferencias del Agente 0 tanto como se pueda. Así, es lógico pensar que los valores de las preferencias óptimas tiendan a ser más bajos que los valores de las preferencias reales.

La siguiente gráfica recoge cómo ha ido mejorando la utilidad obtenida[9] (eje vertical) durante las 800 evoluciones de la población del AGS (eje horizontal), partiendo de las 221 unidades logradas por las preferencias reales.

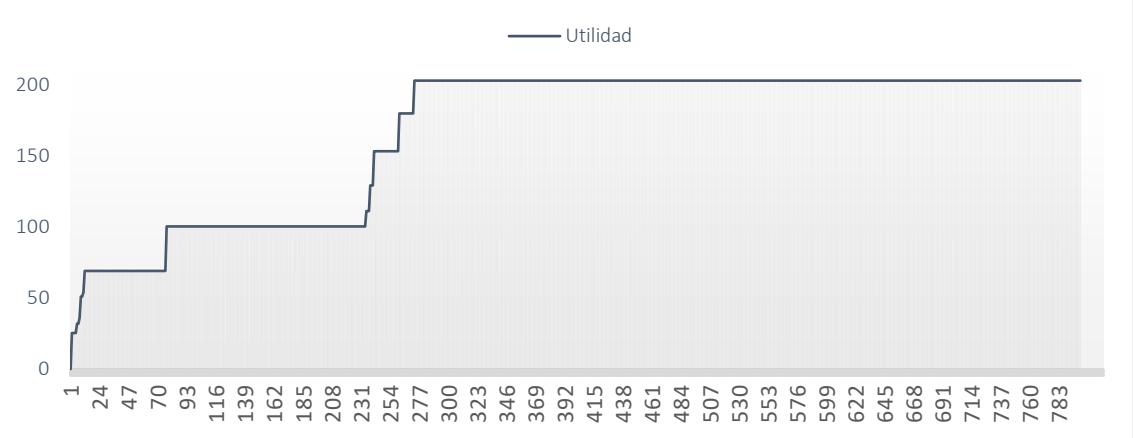


Ilustración 24: Utilidad total en el modo no limitado del AGS. Escenario 1

Se puede observar como el incremento de utilidad cada vez es menos frecuente debido a que hallar unas preferencias mejores resulta cada vez más difícil, aunque es cierto que pequeños cambios en las preferencias pueden desembocar en nuevos caminos a explorar que reporten aún más utilidad, tal y como ocurre en la muestra 235. Puede parecer que obtener una utilidad máxima de 424 con las preferencias óptimas no es mucho, pero de nuevo, hallar individuos que mejoren cada vez más la utilidad no es una tarea sencilla para el AGS. A pesar de ello, con esas 40 horas de ejecución, ambos algoritmos han logrado encontrar una solución óptima[10] en la que siete de los diez recursos disponibles son asignados al Agente 0. La solución óptima en cuestión es: 3100000200 (recordemos que cada dígito indica a qué agente es asignado el recurso de la posición correspondiente). Esta solución, como hemos dicho, proporciona una utilidad de 424 al Agente 0.

En primer lugar, es razonable afirmar que ambos algoritmos han hecho un gran trabajo puesto que se han asignado $n$-3 recursos al Agente 0. Esto es un resultado considerablemente bueno puesto que es el máximo número de recursos que puede recibir sin que alguno de los demás agentes obtenga una utilidad de 0 (caso que no deseamos porque entonces pasaría a ser el agente menos beneficiado). Si entramos en detalle sobre los recursos que han sido asignados al Agente 0 en la solución óptima, rápidamente nos damos cuenta de que esos siete recursos (todos salvo el 0, 1 y 7) se quedan muy cerca del óptimo teórico (es decir, del mejor resultado que el AGS podría haber alcanzado). Si nos fijamos en las preferencias de la **Tabla 12**, los siete recursos que más valora el Agente 0 (los recursos con los que podría haber alcanzado la mayor utilidad posible) son todos salvo

[9] Esta utilidad es el valor absoluto obtenido. Aquí no se ha realizado ninguna resta para calcular el beneficio del Agente 0 tan y como hicimos durante los primeros experimentos cuando solo utilizábamos el AGI.

[10] Denotaremos así a la solución encontrada al haber usado las preferencias óptimas. Por el contrario, denotaremos como "solución real" a la solución hallada tras haber usado las preferencias reales.

el 1, 7 y 8. En otras palabras: únicamente con un cambio de recurso en nuestra solución podríamos haber logrado la mayor utilidad posible. En cualquier caso, creemos que haberos quedado a tan solo 10,69 puntos del óptimo teórico es un resultado notable.

### 4.2.2. Modo limitado

Como dijimos al comienzo del análisis del AGS, en este caso encontrar una serie de preferencias que cada vez aporten mayor utilidad al Agente 0 no es tan fácil (a pesar de que antes tampoco lo era) pues, si el AGS considera que deben disminuirse las preferencias por, digamos, dos de los recursos, esa disminución debe ser sumada a otros recursos para que así se conserve la restricción de limitación. Para este caso, el resultado es que la utilidad obtenida para el Agente 0 considerando sus preferencias reales es de 34,233[11].

De nuevo, en la siguiente tabla podemos ver el valor de las mejores preferencias que ha encontrado el AGS durante las 500 evoluciones de la población.

| | Recurso 0 | Recurso 1 | Recurso 2 | Recurso 3 | Recurso 4 | Recurso 5 | Recurso 6 | Recurso 7 | Recurso 8 | Recurso 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Óptimo** | 11,3590498 | 10,0530664 | 17,9787742 | 7,97992408 | 1,6432262 | 8,11148303 | 13,7607059 | 5,38636778 | 10,5097884 | 13,2176142 |
| **Agente 0** | 17,677023 | 12,586434 | 4,3543668 | 12,006179 | 5,4727495 | 0,7713936 | 14,571267 | 3,4997866 | 16,556633 | 12,504167 |
| **Agente 1** | 1,9274997 | 6,0993133 | 19,66611 | 18,177965 | 10,515567 | 10,757488 | 2,4250987 | 3,3107353 | 23,04639 | 4,0738346 |
| **Agente 2** | 16,957018 | 12,2476 | 11,647019 | 7,330271 | 7,6367753 | 12,5746 | 9,8788009 | 8,7351097 | 11,331465 | 1,6613409 |
| **Agente 3** | 14,415688 | 2,2033675 | 16,789505 | 1,0328801 | 13,158384 | 9,7028421 | 5,182228 | 5,4689635 | 17,193138 | 14,853004 |

Tabla 13: Preferencias en el modo limitado para el AGS

En esta ocasión, el intervalo en el que el AGS modifica los valores de las preferencias es el [-2, 2] para cada una de las evoluciones de su población en la fase de mutación. Hemos considerado que, por razones obvias, el intervalo debe ser menor (pues esta vez debemos afinar más a la hora de realizar las modificaciones).

En la gráfica de utilidad que se presenta a continuación podemos ver que la mayor mejora que se ha alcanzado es de 47,21, lo que es un resultado notablemente peor que en el caso limitado si tenemos en cuenta la utilidad relativa (en porcentaje de crecimiento) alcanzada con las preferencias reales.

[11] En ningún momento debe compararse este dato con el obtenido en el modo limitado. Estos resultados pertenecen a instancias totalmente diferentes, cada una de ellas con sus propias preferencias.

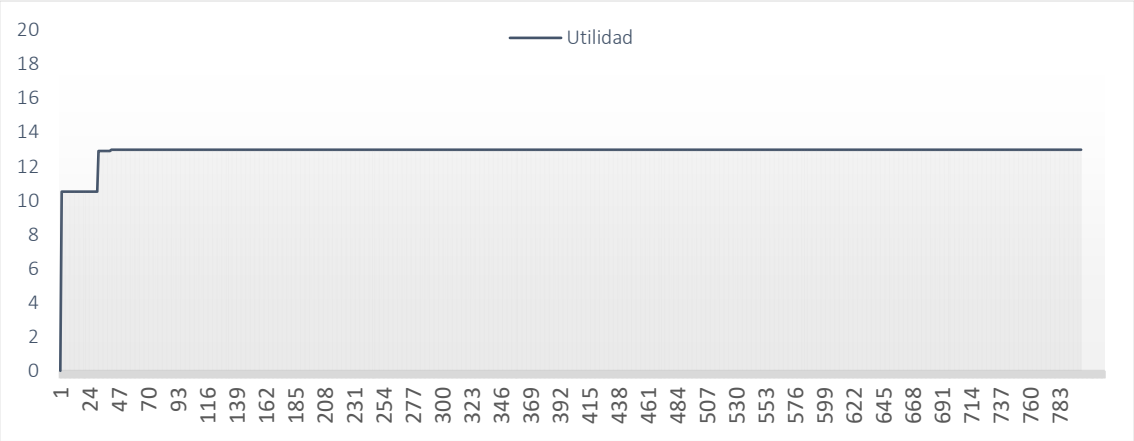


Ilustración 25: Utilidad total en el modo limitado del AGS. Escenario 1

Esta mejora de tan solo 12,88 puntos refleja la enorme dificultad a la que se enfrenta el AGS en el modo limitado ya que es necesaria una mayor precisión a la hora de realizar el aumento y disminución de las preferencias por los recursos.

Para este modo conviene analizar las dos soluciones halladas y ver las diferencias entre los repartos resultantes. El reparto encontrado con las preferencias reales del Agente 0 fue 211323203, mientras que la solución óptima fue 2031120003. Si nos fijamos en las dos soluciones, observamos que la primera solución asigna únicamente el recurso 8 al Agente 0 y que, por otro lado, la solución óptima asigna los recursos 1, 6, 7 y 8. Volvamos a la **Tabla 13**, donde se muestran las preferencias en el modo limitado. Si observamos el reparto con preferencia óptima (usando la mentira), vemos que el AGI ha logrado encontrar un reparto en el que se le han asignado al Agente 0 tres de los cuatro recursos que él más valora, lo cual no está nada mal (teniendo en cuenta que hay cuatro agentes y diez recursos disponibles). Esta solución, a pesar de no incrementar considerablemente la utilidad del Agente 0, indica que el AGS junto con el AGI han "luchado" por tratar de encontrar repartos en los que los seis recursos que menos valora el Agente 0 no estén involucrados (e.d. que no hayan sido asignados al Agente 0).

### 4.3. Análisis aplicando una distribución normal

En el último de los análisis, vamos a ver qué ocurre cuando, habiendo ejecutado el Agente 0 el AGS (e.d. habiendo buscado sus preferencias basándose en una estimación de las preferencias que el resto de los agentes ha comunicado), los demás agentes resultan tener unas preferencias diferentes de las estimadas. ¿Continuará siendo igual de buena la estrategia del Agente 0 por mucho que el resto de los agentes muestren otras preferencias? Si no es así, ¿hasta qué punto merecerá la pena utilizar las preferencias óptimas que obtuvo? La intuición nos dice que, a medida que las preferencias de los demás agentes se van alejando de los valores estimados por el Agente 0, la utilidad obtenida por él irá siendo cada vez peor (pues las preferencias óptimas que logró encontrar el AGS se basaron en su estimación inicial).

Antes de entrar en detalles, nos gustaría comentar algunos aspectos generales que son comunes en el modo no limitado y limitado. Para ir modificando las preferencias de los agentes, lo que haremos será aplicar una distribución normal que, según la desviación estándar establecida, proporcionará valores de preferencias más o menos distantes a los valores que inicialmente el Agente 0 estimó (salvo, por supuesto, los propios valores del Agente 0). Además, es importante conocer algunos conceptos que nos ayudarán a entender mejor los experimentos.

- Ua: denotaremos así a la utilidad obtenida por el Agente 0 después de haber ejecutado el AGS, es decir: la utilidad resultante cuando se maximiza su mentira.
- Ub: es la utilidad que obtiene el Agente 0 con sus preferencias reales. O lo que es lo mismo: sin haber ejecutado el AGS.
- Ua': esta será la utilidad obtenida por el Agente 0 después de haber ejecutado el AGS y tras haber aplicado una desviación estándar a las preferencias del resto de los agentes, para simular la imprecisión de su estimación.
- Ub': análogo al anterior pero considerando que esta vez el Agente 0 no ejecuta el AGS.

Se espera que estas cuatro utilidades que hemos definido alcancen valores algo diferentes en el caso de tener preferencias no limitadas, pues es mucho más fácil llevar a cabo la estrategia de mentir. Los valores para la desviación estándar $\sigma$ que se han tenido en cuenta son: 1.0, 2.0, 4.0, 8.0, 16.0, 32.0, 64.0 y 95.0. Nótese que en estos experimentos el único algoritmo que se ejecuta es el AGI, puesto que ya no es necesario encontrar las preferencias falsas óptimas (ya fueron obtenidas previamente por el AGS). Ahora, lo único que nos interesa es aplicar una cierta desviación estándar a las preferencias del resto de los agentes y ver cómo varía la utilidad lograda por el Agente 0. Luego solamente estamos interesados en hallar soluciones a medida que vamos modificando dichas preferencias. Asimismo, por esta razón los parámetros que se usaron en el AGI se mantienen exactamente iguales.

Cabe destacar que, para cada desviación estándar, se ha hecho una media de resultados totalmente independientes. Aplicar una desviación a las preferencias puede generar soluciones que no reflejen con exactitud el impacto que está teniendo dicha desviación, ya que los nuevos valores son generados de manera aleatoria (tomando como media el valor de preferencia que se esté considerando en cada momento) tal y como se explicó brevemente en la **Sección 2.3**. Es por eso que, para saber con mayor precisión cómo afecta aplicar una determinada desviación a las preferencias, se ha optado por realizar una media entre 100 pruebas para cada una de las desviaciones consideradas. Es decir: cada vez que modificamos la desviación, dicha desviación se aplica 100 veces (cada una de las veces tomando como base de las preferencias óptimas y reales, de manera que la aplicación no es acumulativa) y después, simplemente se procede a calcular la media de las 100 pruebas ejecutadas.

Para terminar, otro resultado interesante que vamos a tener en cuenta será la resta entre las utilidades obtenidas cuando el Agente 0 hace uso de sus preferencias óptimas y cuando simplemente utiliza sus preferencias reales. Vamos a manejar en total los resultados de tres restas:

I. Resta sin desviación: es la resta resultante al hacer la operación Ua – Ub.
II. Resta con desviación: es la resta resultante al hacer la operación Ua' – Ub'.
III. Resta total: es la operación que consiste en restar el resultado de la segunda de las restas (Ua' – Ub') al resultado de la primera (Ua – Ub).

### 4.3.1. Modo no limitado

Como ya vimos en las secciones anteriores, las preferencias no limitadas permiten al Agente 0 mentir de manera eficiente y simple; únicamente debe disminuir su preferencia por cada recurso para aparentar ser poco conformista y entonces el repartidor deberá asignarle más recursos para tratar de cumplir con el objetivo del modelo *Egalitarian*.

Pasemos a ver los resultados obtenidos a medida que incrementamos la desviación sobre las preferencias de todos los agentes salvo el Agente 0. Recordemos que el objetivo de aplicar la desviación es corroborar si de verdad las preferencias encontradas por el Agente 0 al usar el AGS siguen siendo buenas (siguen aportando utilidades relativamente altas) aun cuando las preferencias de los demás agentes difieran cada vez más en la estimación que hizo el Agente 0 justo antes de ejecutar el AGS para hallar su mentira óptima. Como se dijo durante la introducción de esta sección, se ha aplicado cada una de las desviaciones tanto a Ua' como a Ub' (nótese que Ua y Ub siempre permanecen fijas ya que en ambos casos no estamos aplicando ningún tipo de desviación). Recordemos que para cada una de estas pruebas se aplicó 100 veces la misma desviación con el objetivo de promediar cómo afecta cada una de las desviaciones en la utilidad y tener una mayor precisión en los resultados para inferir conclusiones lo más robustas posibles.

La siguiente gráfica muestra cómo evoluciona la utilidad Ua' y Ub' a medida que vamos aplicando una desviación cada vez mayor. En el eje vertical encontramos la utilidad que percibe el Agente 0 y, en el eje horizontal, la desviación aplicada en cada experimento.

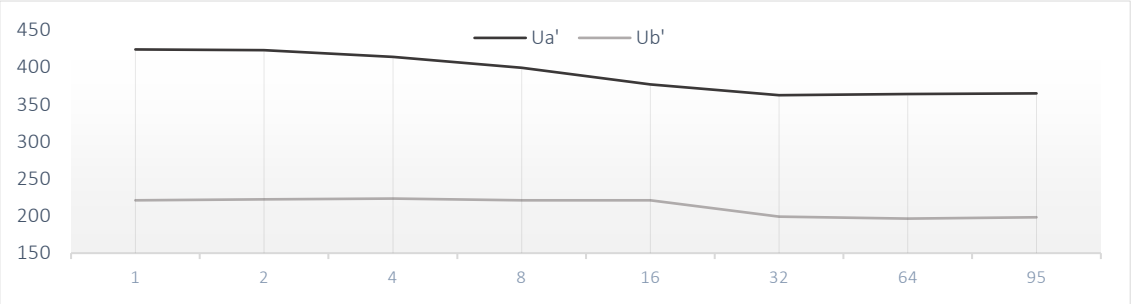


Ilustración 26: Evolución de la utilidad al aplicar desviaciones en modo no limitado

Según los resultados, parece razonable afirmar que, a medida que la desviación crece, la utilidad que percibe el Agente 0 es menor. En otras palabras: cuanto más difieren las preferencias del resto de los agentes respecto a las que el Agente 0 estimó antes de ejecutar el AGS, menos eficiente se vuelven las preferencias óptimas. Por otro lado, esta caída afecta más a Ua' que a Ub', pues como se observa en la gráfica, aunque ambas líneas caen, Ua' sufre una mayor caída en términos relativos. Fijémonos también en que, a partir de una desviación superior a $\sigma = 32.0$, las preferencias se modifican tanto que resulta difícil sacar conclusiones robustas e incluso, para $\sigma = 95.0$, llega a crecer ligeramente. Cabe destacar que, incluso aplicando una desviación final de $\sigma = 95.0$, la utilidad que obtiene el Agente 0 sigue siendo relativamente superior a cuando utiliza sus preferencias reales. Esto es una muestra más de la eficiencia que tienen las mentiras halladas durante la ejecución del AGS.

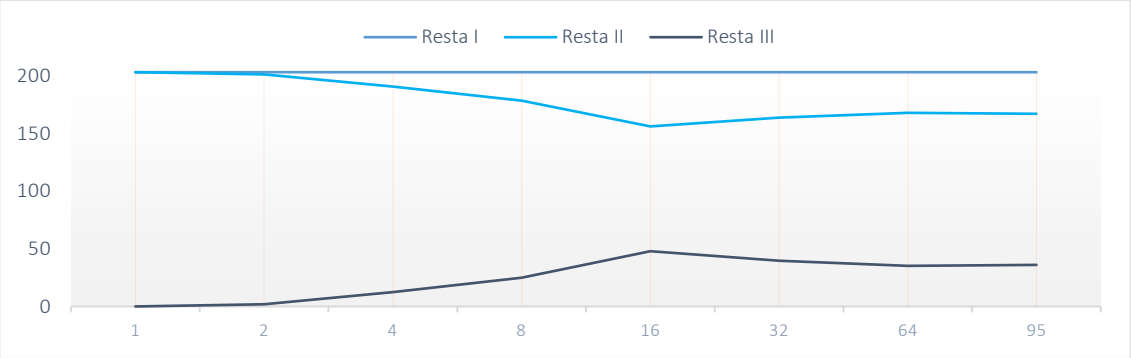


Ilustración 27: Resultado de las tres restas en modo no limitado

Si pasamos a los resultados de las restas que mencionamos anteriormente, vamos a encontrar que, por un lado, Resta I siempre mantiene un valor constante (pues se trata de los resultados sin aplicar ningún tipo de desviación). Y por otro lado, encontramos que Resta II y Resta III serán simétricas para cualquier resultado: cuando Resta II disminuya, Resta III aumentará exactamente las mismas unidades y viceversa. Con estos resultados, parece haber un punto en $\sigma = 16.0$ en el cual la diferencia entre Ua' y Ub' se hace menor para después volver a aumentar. Parece que tendencia general es que, a medida que aumentamos la desviación, Resta II es cada vez menor. Tal y como vimos en la **Ilustración 26**, este efecto es producido porque la desviación afecta más (en términos relativos) a Ua'. De nuevo, los valores obtenidos con desviaciones superiores a $\sigma = 16.0$ no permiten obtener

conclusiones robustas debido a la aleatoriedad de la distribución normal cuando esta es aplicada.

### 4.3.2. Modo limitado

Después de ver los resultados de ir aplicando desviaciones cada vez más altas en el modo no limitado, pasamos al modo limitado. La intuición nos dice que esta vez las preferencias serán considerablemente más sensibles a las variaciones producidas por la desviación, ya que para lograr las preferencias óptimas el AGS tuvo que hacer una labor mucho más fina. Tanto la representación de las gráficas como su significado se mantiene de la misma forma que vimos en el modo no limitado. Dicho esto, pasemos a ver algunos resultados iniciales. En este modo, ¿afectará de la misma forma la desviación a los resultados de utilidad retornados? ¿Se mantendrá ese cierto paralelismo entre Ua' y Ub'?

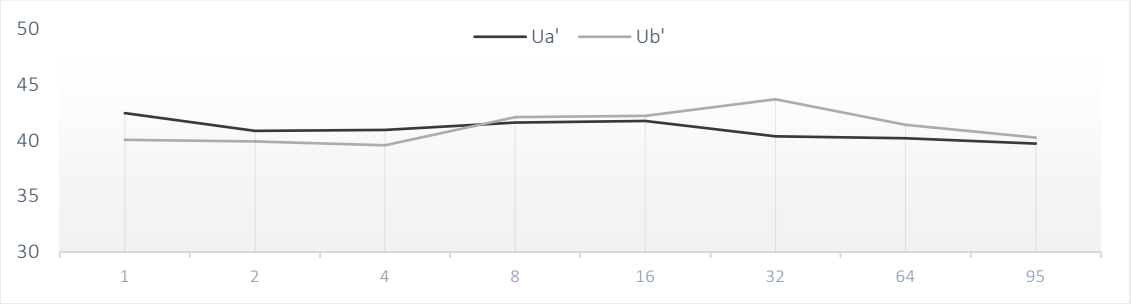


Ilustración 28: Evolución de la utilidad al aplicar desviaciones en modo limitado

Si observamos la **Ilustración 28**, durante las primeras pruebas en las que la desviación es menor, la utilidad de Ua' se ve bastante afectada por las pequeñas variaciones que sufren las preferencias del resto de los agentes. Este impacto en la utilidad, además, se mantiene en caída a medida que la desviación crece. Por otro lado, si observamos la gráfica de Ub', variaciones hasta $\sigma = 4.0$ desembocan en una utilidad sin cambios para el Agente 0, aunque después comienza a aumentar a medida que se aplican desviaciones mayores. Estos dos resultados no son extraños; pensemos por un momento que el AGS ha logrado unas preferencias óptimas basándose en una estimación previa, por lo que, en principio, cualquier pequeña variación en dichas preferencias óptimas dará como resultado una disminución de la utilidad. Asimismo, como las preferencias reales no han sido sometidas a una evaluación para tratar de ser mejoradas y puesto que nos encontramos en el modo limitado, es razonable pensar que pequeñas variaciones supongan obtener una utilidad mayor (recordemos que las preferencias reales fueron generadas totalmente al azar). Con desviaciones superiores a $\sigma = 16.0$ no pueden obtenerse conclusiones notables sobre la evolución de la utilidad.

Si pasamos a ver la **Ilustración 29**, en la que se muestran los resultados de las tres restas en el modo limitado, observamos que la Resta II cae a medida que aumentamos la desviación como consecuencia de los valores obtenidos en Ua' y Ub'. De manera general, es

razonable pensar que cuando se tienen preferencias limitadas sin haber ejecutado el AGS, pequeñas variaciones pueden hacer que se logre una utilidad ligeramente mayor. Por el contrario, pequeñas variaciones en unas preferencias óptimas limitadas hacen que la utilidad caiga considerablemente. De nuevo, no podemos sacar conclusiones robustas a partir de una desviación superior a $\sigma = 16.0$.

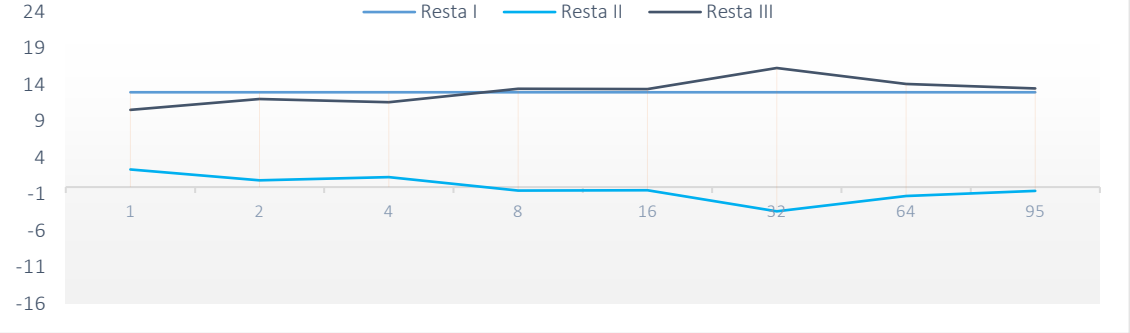


Ilustración 29: Resultado de las tres restas en modo no limitado

# 5. Conclusiones

A lo largo de los experimentos realizados en los cuatro escenarios estudiados en la sección anterior, hemos visto variaciones que afectan en mayor o menor medida al Agente 0, el cual ha sido protagonista durante los tres objetivos que marcamos para este Trabajo de Fin de Master. De manera general y sin entrar en los detalles vistos durante los análisis de las pruebas, pasemos a ver qué conclusiones podemos obtener respecto a los objetivos que fueron marcados en un principio.

Quizá el escenario que se ha visto en mayor profundidad haya sido el Escenario 1, en el que considerábamos diez recursos disponibles a repartir entre cuatro agentes. Los primeros experimentos realizados muestran algunas de las maneras típicas que tiene el Agente 0 para intentar encontrar una manera de beneficiarse cuando se realice el reparto de recursos. Los resultados obtenidos hacen ver que, en el modo no limitado, la mejor estrategia a seguir es sencillamente disminuir la preferencia por los recursos tanto como sea posible, dejando libres los $n$-$m$ recursos que él (el Agente 0) menos valora, donde $n$ es el número de recursos y $m$ es el número de agentes restantes. Esta es, por lo tanto, la estrategia a seguir en el modo no limitado. Por otro lado, las pruebas realizadas en el modo limitado nos permiten concluir que, en este caso, ninguna de las estrategias consideradas funciona.

Una vez visto que a priori la mejor estrategia en el modo no limitado es disminuir las preferencias, no es de extrañar que, durante el segundo de nuestros objetivos, el AGS haya confirmado dicha estrategia. Sin embargo, en el modo limitado, las preferencias encontradas no son fruto de ninguna estrategia como tal, sino más bien de ir probando en cada caso combinaciones sobre las preferencias por los recursos, disminuyendo y aumentando cada una de ellas y calculando, para cada variación, la utilidad lograda para el Agente 0.

Por último, una vez encontradas unas preferencias óptimas en cada caso[12] y tras haber aplicado diferentes desviaciones a las preferencias del resto de los agentes, podemos afirmar que los resultados están fuertemente ligados a la estimación que realizó el Agente 0 justo antes de lanzar el AGS durante el objetivo anterior. Las variaciones provocadas por la desviación han hecho que su utilidad sufra una caída considerable en ambos casos.

[12] Recordemos que así es como nosotros llamados a las preferencias encontradas por el AGS, pero estas preferencias no son las óptimas en términos de lo eficientes que son respecto a otras (o a la teórica óptima que el AGS podría llegar a encontrar).

# 6. Trabajo futuro

Hemos abordado y analizado un gran número de pruebas que nos han permitido conocer un poco más la naturaleza de nuestro juego del reparto bajo el modelo *Egalitarian*. Durante estos meses de trabajo y elaboración del TFM, también hemos identificado una gran variedad de caminos que pueden tomar los trabajos futuros para continuar profundizando en esta área.

No consideramos que los puntos estudiados, así como los objetivos que nos propusimos al principio, tengan una importancia significativamente mayor que los que a continuación se tratan. Simplemente eran la entrada natural en este campo que es relativamente nuevo para nosotros. Tal y como vamos a ver, existen un sinfín de variaciones en el reparto, cada una de ellas muy interesante, que sin duda abordaremos en trabajos futuros. Nos alegra que hayan surgido tantos nuevos caminos que explorar, porque es indicativo de que este es un área que está viva y en el que aún queda mucho trabajo por hacer. Algunas de las líneas de trabajo futuro son las siguientes:

- Estudiar cómo varían las pruebas cuando modificamos el número de recursos disponibles.
- Estudiar cómo varía el escenario general del reparto cuando se consideran utilidades no aditivas y definidas por *packs* de recursos obtenidos. En este caso, la utilidad obtenida por llevarse unos determinados recursos no es solo la suma de las preferencias que había hacia ellos, sino que existe un plus de utilidad añadido según qué combinaciones de recursos se lleven los agentes.
- Estudiar si el resto de los agentes pueden llegar a tener una estrategia eficiente para poder bloquear la mentira del Agente 0 bajo el modelo *Egalitarian* (tanto en modo no limitado como en limitado) o, al menos, de poderla bloquear parcialmente.
- En relación con el punto anterior, estudiar qué ocurre cuando varios agentes (no solo uno) tratan de utilizar a la vez sus respectivas mentiras óptimas.
- Estudiar la complejidad computacional de encontrar la mentira óptima.

# 7. Referencias


[**1**] M. Stonebraker, P. M. Aoki, W. Litwin, A. Pfeffer, A. Sah, J. Sidell, C. Staelin, and A. Yu, "Mariposa: a wide-area distributed database system", *The International Journal on Very Large Data Bases*, vol. 5, no. 1, pp. 048–063, 1996.

[**2**] R. Buyya, D. Abramson, and S. Venugopal, "The grid economy", Proceeding of the IEEE, vol. 93, no. 3, pp 698–714, 2005.

[**3**] X. León, T. A. Trinh, and L. Navarro, "Using economic regulation to prevent resource congestion in large-scale shared infrastructures", Future Generation Computer System, vol. 26, no. 4, pp. 599–607, 2010.

[**4**] Yann Chevaleyre, "Issues in Multiagent Resource Allocation". *May 12, 2005.*

[**5**] Ulle Endriss, Nicolas Maudet, Fariba Sadri and Francesca Toni, "Negotiating Socially Optimal Allocations of Resources". *Journal of Artificial Intelligence Research 25 (2006) 315–348.*

[**5**] Nhan-Tam Nguyen, Trung Thanh Nguyen, Magnus Roos and Jörg Rothe, "Complexity and Approximability of Social Welfare Optimization in Multiagent Resource Allocation". *11th International Joint Conference on Autonomous Agents and Multiagent Systems.*

[**6**] M. Roos and J. Rothe. "Complexity of social welfare optimization in multiagent resource allocation". *In proceedings of the 9th International Joint Conference on Autonomous Agents and Multiagent Systems*, page 641–648. IFAAMAS, May 2010.

[**7**] Aumann, R.J. y Maschler, M. (1985): Game theoretic analysis of a bankruptcy problem from the Talmud. *Journal of Economic Theory*, 36, 195-213.

[**8**] A. Giovannucci, J. A. Rodríguez-Aguilar, A. Reyes, F. X. Noria, and J. Cerquides. Towards automated procurement via agent-aware negotiation support. In *Proc. 3rd International Joint Conference on Autonomous Agents and Multiagent Systems* (*AAMAS-2004*), pages 244–253. ACM Press, 2004.

[**9**] H. van Brussel, J. Wyns, P. Valckenaers, L. Bongaerts, and P. Peeters. Reference architecture for holonic manufacturing systems: PROSA. *Computers in Industry*, 37(3):255–274, 1998.

[**10**] H. Van Dyke Parunak. Applications of distributed artificial intelligence in industry. In G. M. P. O'Hare and N. R. Jennings, editors, *Foundations of Distributed Artificial Intelligence*. John Wiley and Sons, 1996.

[**11**] P. Sousa, C. Ramos, and J. Neves. The Fabricare scheduling prototype suite: Agent interaction and knowledge base. *Journal of Intelligent Manufacturing*, 14(5):441–455, 2003.

[**12**] M. Lemaître, G. Verfaillie, and N. Bataille. Exploiting a common property resource under a fairness constraint: A case study. In *Proc. 16th International Joint Conference on Artificial Intelligence* (*IJCAI-1999*), pages 206–211. Morgan Kaufmann, 1999.

[**13**] M. Lemaître, G. Verfaillie, H. Fargier, J. Lang, N. Bataille, and J.-M. Lachiver. Equitable allocation of earth observing satellites resources. In *Proc. 5th ONERA-DLR Aerospace Symposium* (*ODAS-2003*), 2003.

[**14**] G. M. Jonker, J.-J. Meyer, and F. Dignum. Market mechanisms in airport traffic control. In *Proc. 2nd European Workshop on Multiagent Systems* (*EUMAS-2004*), 2004.

[**15**] E. Cantillon and M. Pesendorfer. Auctioning bus routes: The London experience. In P. Cramton et al., editors, *Combinatorial Auctions*. MIT Press, 2006. To appear.

[**16**] U. Endriss, N. Maudet, F. Sadri, and F. Toni, "Negotiating socially optimal allocations of resources," *CoRR*, vol. abs/1109.6340, 2011.

[**17**] M. Wooldridge. *An Introduction to Multiagent Systems*. Jhon Wiley and Sons, 2002.

[**18**] T. W. Sandholm. Distributed rational decision making. In G. Weiß, editor, *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999.

[**19**] A.J. Umbarkar and P.D. Sheth, "Crossover operators in genetic algorithms: a review," *Ictact journal on soft computing, October 2015, Volume: 06, Issue: 01.*