Proyecto Fin de Carrera
Ingeniería de Tecnologías Industriales

# Aplicación de juegos hedónicos en grupos de robots

Autor: Celia Robles López
Tutor: Andrés Jiménez Losada

**Dpto. Matemática Aplicada II**
**Escuela Técnica Superior de Ingeniería**
**Universidad de Sevilla**

Sevilla, 2019

Proyecto Fin de Carrera
Ingeniería de Tecnologías Industriales

# Aplicación de juegos hedónicos en grupos de robots

Autor:
Celia Robles López

Tutor:
Ándres Jiménez Losada
Profesor titular

Dpto. de Matemática Aplicada II
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2019

Proyecto Fin de Carrera: Aplicación de juegos hedónicos en grupos de robots

Autor: Celia Robles López

Tutor: Andrés Jiménez Losada

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2019

El Secretario del Tribunal

*A mi familia*
*A mis maestros*
*A mis amigos*

# Agradecimientos

En primer lugar, me gustaría agradecer a mis padres todo el apoyo recibido a lo largo de este tiempo sin el que no hubiera sido posible llegar hasta aquí. De igual manera, agradezco a mis hermanos la fuerza y cariño incondicional que siempre me han mostrado.

Durante estos años de formación han sido muchas las personas que me han ayudado a cumplir mis objetivos y me han facilitado el camino. Por ello quiero agradecer a mis compañeros y profesores los conocimientos y competencias que he adquirido gracias a ellos.

Por último, quiero hacer una mención especial a mi tutor Andrés Jiménez Losada por su ayuda y animos durante la realización de este proyecto y por estar siempre a mi total disposición cuando lo he necesitado.

*Celia Robles López*
*Grado en Ingeniería de Tecnologías Industriales*
*Sevilla, 2019*

# Resumen

Muchos han sido los estudios realizados en los últimos años sobre sistemas multirobot debido al gran potencial que han demostrado en distintos ámbitos. Una de estas aplicaciones consiste en el desarrollo de tareas de seguridad y vigilancia en infraestructuras que necesitan ser supervisadas.

En este trabajo se propone un modelo donde varios robots inalámbricos recorren una serie de puntos de interés en un entorno valioso. Cada punto de interés representa una cola de paquetes que requieren la recogida y posterior transmisión inalámbrica de la información por parte de los robots a un servidor central. Este problema se modela como un juego hedónico de formación de coaliciones entre los robots y los puntos que serán visitados. El algoritmo propuesto permite a los robots tomar decisiones distribuidas para unirse o abandonar una coalición, basándose en el beneficio que estos obtienen por pertenecer a la coalición. Como resultado de estas decisiones los robots y los puntos de interés se distribuyen creando coaliciones independientes que forman una partición de red estable.

Se busca optimizar el tiempo en el que los robots desarrollan el trabajo de vigilancia, consiguiendo un modelo eficiente y preciso donde todos los puntos de interés establecidos serán atendidos por uno o varios robots en el menor tiempo posible.

# Abstract

Multirobot systems are revolutionizing various fields of applications, and many studies have been made over the past few years. Among many other areas, this technology is used to develop security and surveillance tasks in infrastructures which require supervision.

This study proposes a model where several wireless robots move through a series of points of interest in a particular environment. Each points of interest represents a queue of packets that demand the collection and subsequent wireless transmission by the robots to a central server. This case is modeled as a hedonic coalition formation game between the robots and the landmarks to be visited. The proposed algorithm allows robots to make distributed decisions to join or leave a coalition, based on the benefits they derive from belonging to a coalition. As a result of these decisions, the robots spread out across landmarks forming independent coalitions that establish a stable network partition.

The objective is to optimize the time duration, in which the robots carry out their surveillance work, achieving an efficient and precise model where all the established points of interest will be served by one or several robots in the minimum possible time.

# Índice

# ÍNDICE DE TABLAS

# ÍNDICE DE FIGURAS

# 1 INTRODUCCIÓN

El concepto de patrullaje consiste en recorrer una cierta zona (puntos de interés) con intención de protegerla y/o supervisarla. Cuando se trata de trabajar en un espacio amplio para realizar una misión, la utilización de robots coordinados permite aumentar la eficacia y eficiencia de la operación. Las soluciones de sistemas de seguridad que utilizan robots móviles en este tipo de aplicaciones tienen muchas ventajas ya que en comparación con otros tipos de vigilancia como por ejemplo el uso de cámaras, los robots móviles no tienen la necesidad de ser vigilados constantemente por un supervisor, por lo que no intervendrán factores humanos como el cansancio o la falta de concentración (no experimentan las limitaciones humanas). Por estas razones, los robots móviles pueden aplicarse para mejorar las soluciones de sistemas de seguridad.

Existen dos tipos de sistemas robóticos: sistemas single-robot (un solo robot) y sistemas multirobot (varios robots). Los sistemas multi robot se han convertido recientemente en un foco de considerable interés debido a su aplicabilidad en distintas áreas. Estos sistemas pueden ir desde simples sensores, adquiriendo y procesando datos, hasta complejas máquinas de tipo humano capaces de interactuar con el entorno de forma bastante compleja. Por eso, este trabajo se centra en desarrollar la idea de patrullaje multirobot aplicada a la supervisión y protección de las distintas instalaciones de la Escuela Técnica Superior de Ingeniería de Sevilla. Las ventajas de usar el sistema multirobot en vez de usar un solo robot son varias:

1. Mejora de la eficiencia ya que puede haber tareas demasiado complejas para que un solo robot obtenga buenos resultados.
2. Varios robots consiguen un sistema más robusto y fiable
3. Los sistemas multi robot mejoran el rendimiento en tareas complejas y distribuidas.
4. Puede ser más barato fabricar varios robots con funciones limitadas que uno muy potente.

Una buena solución al problema del patrullaje multirobot debe reducir el tiempo entre dos visitas al mismo punto de interés y evitar la previsibilidad. Este problema ha recibido mucha atención por parte de los grupos de investigación en los últimos años. Varias obras disponibles en la literatura señalan este interés. En especial, trabajos que describen modelos de patrullaje para coordinar la toma de decisiones entre robots.

Por ello el modelo propuesto en este trabajo busca métodos eficientes para la toma de decisiones en cuanto a la formación de conjuntos (coaliciones) de robots con los distintos puntos de interés que tienen que visitar, a los que llamaremos tareas y que ellos mismos sean capaces de decidir la ruta que seguirán para recorrer las distintas tareas. Por este motivo se ha considerado conveniente utilizar conceptos de Teoría de Juegos.

La teoría de juegos es un área de la matemática aplicada que utiliza modelos para estudiar las interacciones que surgen entre unos agentes en base a unos incentivos (los llamados «juegos») para finalmente tomar una decisión por lo general, óptima. La teoría de juegos se ha convertido en una herramienta sumamente importante para la teoría económica y ha contribuido a comprender más adecuadamente la conducta humana frente a la toma de decisiones. Sus investigadores estudian las estratégias óptimas, así como el comportamiento previsto y observado de individuos en juegos.

En concreto en este trabajo se va a hacer uso de los juegos hedónicos. Un juego hedónico [5] (también conocido como juego de formación de coaliciones hedónicas) es un juego que modela la formación de coaliciones (grupos) de jugadores cuando los jugadores tienen preferencias sobre el grupo al que pertenecen. Un juego hedónico se define como un conjunto finito de jugadores y cada jugador tiene una clasificación de preferencia sobre todas las coaliciones (subconjuntos) de jugadores a los que el jugador pertenece. El resultado de un juego hedónico consiste en una partición de los jugadores en coaliciones desarticuladas, es decir, a cada jugador se le asigna un grupo único. Estas particiones se denominan a menudo estructuras de coalición.

En este trabajo se propone un modelo de formación de coaliciones de robots y tareas basado en conceptos de teoría de juegos y más en concreto haciendo uso de la teoría de juegos hedónicos, que consiste en que los distintos robots vayan recorriendo las tareas de su coalición y supervisando así el área que se quiere vigilar, según las coaliciones formadas todas las tareas tendrán un robot que las supervise. Una vez se llega a cada tarea, transmiten la información de esta a un receptor central y continúan con la siguiente.

# 2 TEORÍA DE JUEGOS

## 2.1 Introducción a la teoría de juegos

En el lenguaje ordinario, la palabra juego hace referencia a divertimento y también a actividad en que los participantes, sometidos a reglas que hay que cumplir, intentan ganar, pero pueden perder. Aunque la palabra "juego" tiene connotaciones lúdicas y relativas al azar, la teoría de juegos trata de entender en qué situaciones puede surgir la necesidad de un comportamiento estratégico o bajo qué tipo de condiciones podría establecerse la cooperación entre individuos. En estos juegos, cada jugador intenta conseguir el mejor resultado posible (maximizar su utilidad), pero teniendo en cuenta que el resultado del juego no depende solo de sus acciones, sino también de las acciones de los otros jugadores. Es esta característica de los juegos -tomar decisiones que más convengan para ganar teniendo que cumplir las reglas del juego y sabiendo que los demás jugadores también influyen en los resultados con sus decisiones- la que más valor tiene para su estudio sistemático, ya que muchas situaciones de interés para la economía y para otras ciencias (como biología, sociología, ingeniería...) comparten con ellos esa característica.

Se estudian a menudo situaciones de decisión individual, en las que el agente intenta optimizar su utilidad, sin importar lo que hagan los otros. Por ejemplo, la elección de cantidades de materiales a utilizar o producir para el desarrollo de un proyecto.

Sin embargo, hay muchas otras situaciones en que la utilidad del resultado final no depende solo de la acción del agente, sino también de los otros agentes. En el ejemplo anterior, si el mismo proyecto va a ser realizado por diversos agentes compitiendo entre si.

El planteamiento según el cual no importa lo que hagan otros agentes es por tanto una simplificación de la realidad.

Algunos aspectos básicos de teoría de juegos son los siguientes:

- Jugadores: Son los participantes en el juego que toman decisiones con el fin de maximizar su utilidad.
- Acciones de cada jugador: Son las decisiones que puede tomar cada jugador en cada momento en que le toque jugar. El conjunto de acciones de un jugador en cada momento del juego puede ser finito o infinito.
- Resultados del juego: Son los distintos modos en que puede concluir un juego. Cada resultado lleva aparejadas unas consecuencias para cada jugador

- Pagos: Cada jugador recibe un pago al acabar el juego, que depende de cuál haya sido el resultado del juego. El significado de dicho pago es la utilidad que cada jugador atribuye a dicho resultado, es decir, la valoración que para el jugador tienen las consecuencias de alcanzar un determinado resultado en el juego.
- Estrategias: Una estrategia de un jugador es un plan completo de acciones con las que éste podría proponerse participar en dicho juego. Un perfil de estrategias sería el conjunto de estrategias, una por cada jugador.
- Coalición: Jugadores que forman grupos que realizan acciones y estrategias conjuntas.

La Teoría de Juegos fue creada por Von Neumann y Morgenstern, y descripta en su libro clásico *The Theory of Games Behavior* [2], publicado en 1944. Otros habían anticipado algunas ideas. Los economistas Cournot y Edgeworth fueron particularmente innovadores en el siglo XIX. Otras contribuciones posteriores mencionadas fueron hechas por los matemáticos Borel y Zermelo. El mismo Von Neumann ya había puesto los fundamentos en el artículo publicado en 1928. Sin embargo, no fue hasta que apareció el libro de Von Neumann y Morgenstern que el mundo comprendió cuán potente era el instrumento descubierto para estudiar las relaciones humanas.

Von Neumann y Morgenstern investigaron dos planteamientos distintos de la Teoría de Juegos. El primero de ellos el planteamiento estratégico o no cooperativo. Von Neumann y Morgenstern resolvieron este problema en el caso particular de juegos con dos jugadores cuyos intereses son diametralmente opuestos. A estos juegos se les llama estrictamente competitivos, o de suma cero, porque cualquier ganancia para un jugador siempre se equilibra exactamente con una pérdida correspondiente para el otro jugador. El ajedrez, el backgammon y el póquer son juegos tratados habitualmente como juegos de suma cero.

En el segundo de ellos desarrollaron el planteamiento coalicional o cooperativo, en el que buscaron describir la conducta óptima en juegos con muchos jugadores. Puesto que éste es un problema mucho más difícil, no es de sorprender que sus resultados fueran mucho menos precisos que los alcanzados para el caso de suma cero y dos jugadores.

Entre los juegos no cooperativos [1] caben dos distinciones básicas: los juegos estáticos o dinámicos, y juegos con o sin información completa. En los juegos estáticos los jugadores toman sus decisiones simultáneamente (cada jugador decide sin saber que han decidido los demás), mientras que en los dinámicos puede darse el caso de que un jugador conozca ya las decisiones de otro antes de decidir. En los juegos con información completa todos los jugadores conocen las consecuencias, para sí mismos y para los demás, del conjunto de decisiones tomadas, mientras que en los juegos con información incompleta, algún jugador desconoce alguna de las consecuencias.

En el enfoque cooperativo se analizan las posibilidades de que algunos o todos los jugadores lleguen a un acuerdo sobre qué decisiones tomaría cada jugador en ausencia de acuerdo previo. Dos aspectos son vitales en dicho análisis: qué coaliciones se formarán y como se repartirán los beneficios obtenidos. Los juegos cooperativos de utilidad transferible responden a la segunda pregunta mientras que los juegos hedónicos describen un método para responder a la segunda. El orden lógico para la resolución de problemas sea primero encontrar los grupos de jugadores que cooperan y luego repartir los beneficios, pero en muchas ocasiones sólo estamos interesados en agrupar de manera efectiva a los agentes. Sin embargo, para la resolución del primer problema también se utilizan métodos de juegos de utilidad transferible.

El trabajo que se realiza a continuación aplicará la teoría de juegos cooperativos para encontrar qué grupos de tareas y robots deben relacionarse en busca de la eficiencia., es decir nescitaremos ciertos conocimientos sobre juegos de utilidad transferible y sobre juegos hedónicos.

## 2.2 Juegos cooperativos

### 2.3.1. Concepto de juego cooperativo

Como ya comentamos en el apartado anterior, los Juegos Cooperativos se caracterizan por el hecho de que los jugadores pueden cooperar entre ellos para buscar un beneficio común. Una cuestión importante en la Teoría de Juegos Cooperativos es que, en el momento en que varios jugadores deciden cooperar en algún sentido, debe formarse una coalición entre estos jugadores. Los jugadores de esta coalición, desde ese instante, actuarán buscando el máximo beneficio posible para la coalición. Una coalición puede estar formada por cualquier grupo de jugadores de cualquier tamaño. El pago de esta coalición, esto es, los beneficios que la coalición obtendrá del juego, será un valor numérico, y deberá ser repartido al finalizar el juego entre los jugadores que forman la coalición. Estos pagos vendrán dados por una función $v(N)$ a lo que denominamos función característica. Cuando cualquier reparto del pago entre los jugadores es posible, hablamos de un juego con Utilidad Transferible. Algunas definiciones que usaremos de los juegos son las siguientes.

**Definición 2.1.** Un juego en forma coalicional o en forma de función característica con utilidades trasferibles, **juego UT** para abreviar, consiste en:

- Un conjunto finito de jugadores $N = \{1, 2, \dots, n\}$.
- Una función característica, $v: P(N) \rightarrow \mathbb{R}$ siendo $P(N)$ el conjunto de condiciones en $N$ (o conjunto de poder) que asocia a cada subconjunto $S$ de N (o coalición) un número real v($S$) (valor de la coalición), siendo v($\varnothing$) = 0.

Por tanto, denotamos a un juego UT como $(N, v)$ donde tanto $N$ como $v$ deben estar especificados. Como vemos, la única restricción que se le impone a la función característica es que a la coalición formada por un conjunto vacío de jugadores le asigne un pago nulo.

**Definición 2.2.** Se dice que un juego $(N, v)$ es **monótono** si $\forall\, S, T \subseteq N$ *con* $S \subseteq T$, se verifica que

$$v(S) \leq v(T).$$

Es decir, el juego es monótono cuando el beneficio se incrementa al construirse mayores coaliciones.

**Definición 2.3.** Se dice que un juego $(N, v)$ es **superaditivo** si $\forall\, S, T \subseteq N$, con $S \cap T = \varnothing$, se verifica que:

$$v(S) + v(T) \leq v(S \cap T).$$

Esto significa que la cooperación entre dos coaliciones espera recibir más beneficio que por separado. Si la desigualdad de la definición anterior se da en sentido opuesto se dice que el juego es subaditivo. Algunos autores, como Shapley [4] o Owen [3], consideran que los juegos tienen incluida la condición de ser superaditivos. En el juego superaditivo los jugadores tienen un interés claro en formar la gran coalición $N$.

**Definición 2.4.** Se dice que un juego $(N, v)$ es **convexo** si $\forall\, S, T \subseteq N$, se verifica que

$$v(S) + v(T) \leq v(S \cup T) + v(S \cap T).$$

Si la desigualdad se da en sentido opuesto se dice que el juego es **cóncavo.** Es decir, un juego es cóncavo si

$$v(S \cup T) \leq v(S) + v(T) - v(S \cap T).$$

**Definición 2.5.** Se dice que un juego $(N, v)$ es **0-normalizado** si se verifica que

$$v(\{i\}) = 0, \forall\, i \in N.$$

Nótese que en los juegos 0-normalizados, los jugadores están obligados a cooperar entre ellos, porque solos obtendrán un beneficio nulo.

**Definición 2.6.** Se dice que un juego $(N, v)$ es **(0,1)-normalizado** si se verifica que

$$v(\{i\}) = 0, \forall i \in N \text{ y } v(N) = 1.$$

### 2.3.2. Ejemplos de juegos cooperativos

*El juego de las patentes*

Dos centros de investigación, que llamaremos como jugadores 1 y 2 respectivamente, han obtenido de manera independiente fórmulas muy parecidas de un nuevo fármaco para aliviar la jaqueca. El centro 1 ha patentado y homologado su fórmula para el conjunto de los países asiáticos y países de la Unión Europea. El centro 2 tiene homologada su patente para los países asiáticos y para Estados Unidos.

La comercialización de uno de los fármacos producirá unos beneficios de 7 billones de dólares en el mercado asiático, de 3 billones en el europeo y de 3 billones en Estados Unidos. Si se comercializan a la vez los dos fármacos en un mismo mercado, los beneficios se repartirían a partes iguales.

Por otra parte, supongamos que hay dos empresas, jugadores 3 y 4, que tienen los factores y la tecnología necesarios para la fabricación de los fármacos, pero que cada una de ellas tiene una limitación en la producción ya que sólo puede abastecer al mercado asiático y a uno de los otros mercados (europeo o americano).

La posesión de la fórmula magistral o de los medios de producción, por separado, no aporta valor. Cualquiera de las dos empresas puede fabricar y comercializar cualquiera de los dos fármacos pero cada uno de los centro que posee una de las fórmulas sólo puede conceder su licencia a una de las empresas, y cada empresa solo puede obtener una licencia.

La representación del juego en forma coalicional es la siguiente:

$$N = \{1, 2, 3, 4\}$$

en donde los jugadores están especificados en el enunciado.

Obtengamos la función característica: cada coalición en la que no haya al menos un centro de investigación y una empresa no puede obtener ningún beneficio. Por tanto,

$$v(\varnothing) = 0,$$

$$v(\{1\}) = 0,\ v(\{2\}) = 0,\ v(\{3\}) = 0,\ v(\{4\}) = 0,$$

$$v(\{1, 2\}) = 0,\ v(\{3, 4\}) = 0$$

Cuando un solo centro de investigación llega a un acuerdo de cooperación con una sola de las empresas, la coalición que se forma de centro con empresa se garantiza un beneficio de 3.5 (la mitad del beneficio del mercado asiático) más 3 (correspondiente al mercado europeo o al americano). Por tanto,

$$v(\{1, 3\}) = v(\{1, 4\}) = v(\{2, 3\}) = v(\{2, 4\}) = 6.5$$

Cuando cooperan los dos centros de investigación con sólo una de las empresas (cualquiera que sea), alcanzan conjuntamente un beneficio de 7 (mercado asiático) más 3 (uno de los mercados europeos o americano. Por tanto,

$$v(\{1, 2, 3\}) = v(\{1, 2, 4\}) = 10$$

Cuando se forma una coalición formada por las empresas y un solo centro de investigación (cualquiera que sea), alcanzan conjuntamente un beneficio de 7 (mercado asiático) mas 3 (mercado europeo si el centro 1 está en la coalición, mercado americano si el centro de investigación 2 está en la coalición). Por tanto,

$$v(\{1, 3, 4\}) = v(\{2, 3, 4\}) = 10$$

Por último, si deciden trabajar conjuntamente los dos centros de investigación y las dos empresas, alcanzarán conjuntamente un beneficio de 7 (mercado asiático) más 3 (mercado europeo) más 3 (mercado americano). Es decir,

$$v(\{1, 2, 3, 4\}) = 13$$

Por tanto, la representación del juego en forma coalicional es:

$$N = \{1, 2, 3, 4\}$$

$$v: P(N) \rightarrow \mathbb{R} \text{ con}$$

$$v(\varnothing) = 0,$$

$$v(\{1\}) = 0,\ v(\{2\}) = 0,\ v(\{3\}) = 0,\ v(\{4\}) = 0,$$

$$v(\{1, 2\}) = 0,\ v(\{1, 3\}) = 6.5,\ v(\{1, 4\}) = 6.5,\ v(\{2, 3\}) = 6.5\ v(\{2, 4\}) = 6.5,$$

$$v(\{3, 4\}) = 0,\ v(\{1, 2, 3\}) = 10,\ v(\{1, 2, 4\}) = 10\ v(\{1, 3, 4\}) = 10,$$

$$v(\{2, 3, 4\}) = 10,\ v(\{1, 2, 3, 4\}) = 13$$

*Planta de reciclaje*

Tres pueblos de la misma provincia han decidido hacer una planta de reciclaje, para no tener que transportar los residuos a otras provincias. Los tres ayuntamientos proponen como posibilidad no realizar tres plantas, sino cooperar entre ellos. Tras un estudio de costes, esta es la situación

$$v(\{1\}) = 100,\ v(\{2\}) = 150,\ v(\{3\}) = 100$$

$$v(\{1, 2\}) = 150,\ v(\{1, 3\}) = 160,\ v(\{2, 3\}) = 180$$

$$v(\{1, 2, 3\}) = 200.$$

Hemos tomado $N = \{1,2,3\}$. El valor de la función característica del juego para cada coalicón viene determinado por el costo presupuestado para la construcción de una planta con capacidad suficiente como para dar servicio a los pueblos de dicha coalición.

### 2.3.3. El conjunto de imputaciones y el core

Sea ($N$, $v$) un juego UT superaditivo en donde $N = \{1, 2, \ldots, n\}$ es el conjunto de jugadores y $v$ es la función característica. Si los jugadores deciden trabajar conjuntamente, es decir cooperar, el problema que se presenta consiste en cómo repartir el valor v($N$) entre los $n$ jugadores. Supondremos también que se trata de un juego de beneficios.

Sea el vector de distribución de pagos $x = (x_1, x_2, ..., x_n)$, en donde para cada $i = 1, 2, \ldots,$ n, $x_i$ representa el pago que recibe el jugador $i$. Para cualquier coalición S $\subseteq N$, se utilizará la siguiente notación:

$$x(S) = \sum_{i \in S} x_i$$

Además, se define $x(\varnothing) = 0$, de forma técnica.

A la hora de usar los juegos cooperativos UT para modelar situaciones de la vida real, existen una serie de restricciones lógicas para el vector de pagos:

Suponiendo que todos los jugadores llegan a un acuerdo, formando la gran coalición $N$, el beneficio total de esa gran coalición viene representado por v($N$). Si al finalizar el juego reciben el vector de pagos $x = (x_1, x_2, ..., x_n)$, este vector de pagos satisface el **principio de eficiencia** cuando

$$\sum_{i \in N} x_i = v(N)$$

Este principio impone que, si se forma la gran coalición $N$, el beneficio de la misma será repartido en su totalidad por lo miembros que la forman.

Para que los jugadores acepten la distribución de beneficios propuesta por el vector de pagos, tienen que recibir un pago superior al que recibirían si jugasen solos. Este es el llamado **principio de individualidad racional**:

$$x_i \geq v(\{i\}),\ \forall\ i = 1, 2, \dots n.$$

Los vectores $x \in \mathbb{R}^n$ que cumplen con el principio de eficiencia son llamados vectores de pagos eficientes o **preimputaciones** para el juego (*N, v*). Podemos por tanto definir el conjunto de preimputaciones de un juego (*N, v*) como el conjunto de vectores de distribución de pagos:

$$PI(N, v) = \{x = (x_1,\ x_2, \dots, x_n) \in\ \mathbb{R}^n : x(N) = v(N)\}$$

Como vemos, el conjunto de preimputaciones no es más que el conjunto de todos los vectores de pagos que cumplen el principio de eficiencia. Si además de este principio, imponemos que los vectores de pagos cumplan el principio de individualidad racional, obtenemos el conjunto de imputaciones de un juego (*N, v*).

**Definición 2.7.** Una **imputación** es un vector de pagos $x \in \mathbb{R}^n$ que es eficiente y verifica el principio de racionalidad individual.

$$I(N, v) = \{x = (x_1, x_2, \dots, x_n) \in PI(N, v): x_i\ \geq v(\{i\}), para\ i = 1, 2, \dots, n\} = \ \{x = ((x_1, x_2, \dots, x_n) \in\ \mathbb{R}^n : x(N) = v(N), x_i\ \geq v(\{i\}), para\ i = 1, 2, \dots, n\}$$

Se dice que el juego (*N, v*) es **esencial** si se verifica que I (*N, v*) $\neq \varnothing$

Si consideramos el ejemplo anterior, en la siguiente figura se muestra el conjunto de imputaciones para el juego cooperativo *de las patentes*.

En el caso de 4 jugadores, el conjunto de imputaciones se suele representar como un tetraedro en el plano, guardando analogía con la representación para 3 jugadores que vendría representada por un triangulo.

Para dicho juego se tiene que

$$I(J, v) = \{(\{x_1, x_2, x_3, x_4\}) \in\ \mathbb{R}^4 : x_1 +\ x_2 + x_3 + x_4 = 13,\ x_1 \geq 0,\ x_2 \geq 0,\ x_3 \geq 0,\ x_4 \geq 0\}$$

En la siguiente figura podemos ver el resultado gráficamente:

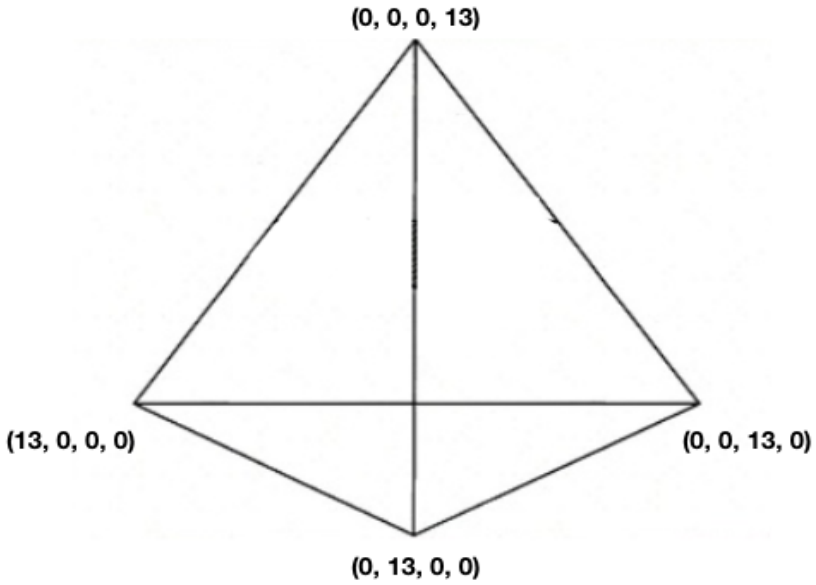


Figura 2.1. Conjunto de imputaciones para 4 jugadores.

Dentro de los múltiples conceptos de solución en la Teoría de Juegos, uno de los más importantes es el core. Si tenemos un juego cooperativo $(N, v)$ donde $N = \{1, 2, \ldots, n\}$, y $v$ la función característica que describe el juego, se desea extraer un subconjunto (del conjunto de preimputaciones) de vectores de pago que los jugadores estén dispuestos a aceptar. Si, además de exigirle que cumplan el principio de eficiencia, le exigimos a los vectores de pago que cumplan el principio de racionalidad individual, hablamos de extraer un subconjunto de vectores del conjunto de imputaciones. Podemos ampliar el principio de racionalidad individual mediante el principio de racionalidad coalicional, llegando entonces al concepto de core de un juego cooperativo.

Una coalición o conjunto de jugadores que pudiese obtener un pago cooperando, también exigirá de un vector de pagos un beneficio mayor al que obtendría formando la coalición. De manera análoga al principio de individualidad racional, tenemos la condición de racionalidad de grupo o, también llamada, condición de **optimalidad de Pareto**.

$$\sum_{i \in S} x_i = x(S) \geq v(S)$$

**Definición 2.8.** El **core** de un juego $(N, v)$ es el siguiente conjunto de vectores de pagos:

$$C(N, v) = \{x = ((x_1, x_2, \ldots, x_n) \in \mathbb{R}^n : x(N) = v(N), x(S) \geq v(S), para\ todo\ S \in P(N)\}$$

En definitiva, podemos decir que el core es el conjunto de vectores de pago que ofrece a cada coalición que puede formarse sobre *N* un beneficio, al menos, igual que el que esta coalición puede conseguir por sí misma. Por tanto, los elementos del core son aceptables para todas las coaliciones S $\subseteq$ *N*. Un reparto de pagos que satisface a todos los jugadores y a todas las posibles coaliciones.

Sea (*N*, *v*) un juego cooperativo. El conjunto C (*N*, *v*) es cerrado, acotado y convexo. No obstante, pueden darse casos donde el core puede ser muy grande o incluso estar vacio. Si es grande, habrá que plantearse cual de sus vectores de pago escoger. Para los juegos en los que el core es vacio no se podría obtener un vector de pagos con el que todos los jugadores o posibles coaliciones se vieran beneficiados. Sin embargo, hay clases de juegos cooperativos de utilidad transferible para los que el core es no vacio.

Dado que el core nos da una solución para un juego cooperativo, y que existen juegos con el core vacío, es un objetivo importante de la Teoría de Juegos Cooperativos caracterizar los juegos cooperativos con el core no vacío. A cerca de lo cual, Shapley (1967) introdujo el concepto de coaliciones equilibradas y de juego equilibrado.

**Definición 2.9.** Una familia $\{S_1, S_2, \dots, S_m\}$ de subconjuntos de *N*, distintos y no vacíos, es **equilibrada** sobre *N* si existen números positivos $\propto_1, \propto_2, \dots, \propto_m$, denominados pesos, tales que para todo *i* = 1, 2, …, *n* verifican:

$$\sum_{\{j:i\in S_j\}} \propto_j = 1$$

Se dice que el juego (*N*, *v*) es equilibrado si, para cualquier colección equilibrada sobre *N*, se verifica que:

$$\sum_{j=1}^{m} \propto_j v(S_j) \leq v(N)$$

Bondareva (1963) y Shapley (1967) demostraron que la clase de juegos equilibrados coincide con la clase de juegos con core no vacío.

Considerando el ejemplo realizado anteriormente vamos a obtener el core del *juego de las patentes*:

Se trata de un juego con 4 jugadores. Las restricciones que deben cumplir los elementos del core, en este caso, son las siguientes:

Principio de eficiencia:

$$x_1 + x_2 + x_3 + x_4 = 13$$

Racionalidad individual:

$$x_1 \geq 0,\ \ x_2 \geq 0,\ \ x_3 \geq 0,\ \ x_4 \geq 0$$

Racionalidad de las coaliciones formadas por dos jugadores:

$$x_1 + x_2 \ \geq 0,\ \ x_1 + x_3 \ \geq 6.5,\ \ x_1 + x_4 \geq 6.5,$$

$$x_2 + x_3 \ \geq 6.5,\ \ x_2 + x_4 \geq 6.5,\ \ x_3 + x_4 \geq 0$$

Racionalidad de las coaliciones formadas por tres jugadores:

$$x_1 + x_2 + x_3 \ \geq 10,\ \ x_1 + x_2 \ + x_4 \ \geq 10,$$

$$x_1 + x_3 + x_4 \ \geq 10,\ \ x_2 + x_3 \ + x_4 \ \geq 10,$$

Utilizando la igualdad del principio de eficiencia en las desigualdades correspondientes a la racionalidad de coaliciones de tres jugadores, queda:

$$x_1 + x_2 + x_3 \ \geq 10 \iff 13 - \ x_4 \ \geq 10 \iff x_4 \ \leq 3$$

$$x_1 + x_2 + x_4 \ \geq 10 \iff 13 - \ x_3 \ \geq 10 \iff x_3 \ \leq 3$$

$$x_1 + x_3 + x_4 \ \geq 10 \iff 13 - \ x_2 \ \geq 10 \iff x_2 \ \leq 3$$

$$x_2 + x_3 + x_4 \ \geq 10 \iff 13 - \ x_1 \ \geq 10 \iff x_1 \ \leq 3$$

Es imposible que $x_1 \leq 3, x_3 \leq 3$ y a la vez $x_1 + \ x_3 \geq 6.5$. Por tanto, el core de este juego es el conjunto vacio.

### 2.3.4. Valores

Un **valor** para juegos UT es una aplicación *f* que define un vector de distribución de pagos para cada juego (*N, v*), que denotamos por *f* (*N, v*). Es decir, es una regla que determina como repartir el beneficio en cada juego.

La regla más sencilla que puede ocurrirse es denominada la regla igualitaria.

**Definición 2.10.** La **regla igualitaria** es una aplicación que le hace corresponder a cada juego UT (*N, v*) el vector de pagos $\mathcal{E}(N, v)$ dado por:

$$\mathcal{E}_i(N, v) = \frac{v(N)}{n} \ con\ |N| = n$$

Dentro de los juegos cooperativos UT, el valor de Shapley, es uno de los conceptos de solución más utilizados. Shapley analizó los juegos cooperativos con la finalidad de, dada la función característica de un juego, establecer el pago esperado para un jugador determinado. Se trata de buscar un reparto de pagos único que cumpla una serie de propiedades o axiomas previamente establecidos. Shapley partió de cuatro axiomas o suposiciones que, según él, debería cumplir el reparto de pagos óptimo, y demostró que sólo una asignación de pagos cumplía todos los axiomas, siendo esta asignación el valor de Shapley. Es importante destacar que el valor de Shapley es un concepto de solución independiente del core, y al no exigirle que cumpla el principio de racionalidad coalicional, no siempre es una solución que pertenezca al core. Sin embargo, para los juegos convexos, el valor de Shapley sí pertenece al core del juego.

**Definición 2.11.** El **valor de Shapley** es la aplicación $\phi$ que asigna a cada juego $(N, v)$ el vector de pagos $\phi(N, v)$ dado por

$$\phi_i(N, v) = \sum_{S \in P(N)} q^N(S)\,[v(S) - v(S - \{i\})]$$

en donde

$$q^N(S) = \frac{(s-1)\,!\,(n-s)!}{n!}$$

siendo s = $|S|$, el número de jugadores que hay en la coalición S.

Si para un valor f exigimos los siguientes axiomas:

Axioma 1.

**Eficiencia**. La función de asignación $\phi(N, v)$ debe distribuir el pago total del juego. Es decir, debe ser

$$\sum_{i=1}^{n} f_i(N, v) = v(N)$$

Axioma 2.

**Simetría.** Para cualquier par de jugadores que realicen aportaciones equivalentes para cada coalición, es decir, tales que cumplan que

$$v(S \cup \{i\}) = v(S \cup \{N\}), para\ todo\ S \in P(N), con\ i, j \notin S$$

debe ser

$$f_i(N, v) = f_j(N, v)$$

Axioma 3.

**Tratamiento del jugador pasivo.** Si un jugador no aporta ningún beneficio adicional al resto de jugadores no debe recibir ningún pago adicional. Es decir, para cada jugador $i \in J$, para el cual se verifica que

$$v(s) = v(S - \{i\}) + v(\{i\}), para\ toda\ coalición\ S\ con\ i \in S$$

debe ser

$$f_i(N, v) = v(\{i\})$$

Axioma 4.

**Aditividad.** La función de asignación ϕ debe ser invariante a cualquier descomposición arbitraria del juego. Formalmente, dados dos juegos cualesquiera $(N, v_1)$ y $(N, v_2)$ debe ser

$$f(N, v_1 + v_2) = f(N, v_1) + f(N, v_2)$$

El único valor que verifica 1, 2, 3 y 4 es el valor de Shapley.

Se puede observar que el valor de Shapley está determinado, de forma exclusiva y a priori, por la función característica del juego.

El valor de Shapley tiene distintas interpretaciones. Puede interpretarse como la contribución marginal esperada de cada jugador al entrar en una coalición al azar. En efecto, el factor $v(S) - v(S - \{i\})$ es la contribución marginal efectiva de i al incorporarse a $S - \{i\}$, mientras que el factor $q(s)$ es la probabilidad de que a i le toque incorporarse precisamente a $S - \{i\}$ y no a otra coalición.

Otra manera de interpretar el valor de Shapley es suponer que los jugadores forman la gran coalición incorporándose de uno en uno, en un orden aleatorio. De esta forma, cada jugador consigue la cantidad con la que él contribuye a la coalición ya formada cuando se incorpora. El valor de Shapley distribuye a cada jugador la cantidad esperada que él obtiene por este procedimiento, suponiendo que la gran coalición de n jugadores puede formarse, de manera equiprobable, en todos los órdenes posibles.

Vamos a calcular el valor de Shapley en el ejemplo *de las patentes:*

Se trata de un juego de 4 jugadores por lo que

$$\phi(N,v) = (\phi_1(v), \phi_2(v), \phi_3(v), \phi_4(v))$$

en donde siguiendo el teorema anterior:

$$\begin{aligned}\phi_1(N,v) = {} & \frac{1}{4}\, v(\{1\}) + \frac{1}{12}[\, v(\{1,2\}) - v(\{2\})] + \frac{1}{12}[\, v(\{1,3\}) - v(\{3\})] + \frac{1}{12}[\, v(\{1,4\}) - v(\{4\})] \\ & + \frac{1}{12}[\, v(\{1,2,3\}) - v(\{2,3\})] + \frac{1}{12}[\, v(\{1,2,4\}) - v(\{2,4\})] \\ & + \frac{1}{12}[\, v(\{1,3,4\}) - v(\{3,4\})] + \frac{1}{4}[\, v(\{1,2,3,4\}) - v(\{2,3,4\})] \\ = {} & \frac{1}{4}\,[0] + \frac{1}{12}\,[0-0] + \frac{1}{12}\,[6.5-0] + \frac{1}{12}\,[6.5-0] + \frac{1}{12}\,[10-6.5] + \frac{1}{12}\,[10-6.5] \\ & + \frac{1}{12}\,[10-0] + \frac{1}{4}\,[13-10] = \frac{13}{4}\end{aligned}$$

$$\begin{aligned}\phi_2(N,v) = {} & \frac{1}{4}\, v(\{2\}) + \frac{1}{12}[\, v(\{1,2\}) - v(\{1\})] + \frac{1}{12}[\, v(\{2,3\}) - v(\{3\})] + \frac{1}{12}[\, v(\{2,4\}) - v(\{4\})] \\ & + \frac{1}{12}[\, v(\{1,2,3\}) - v(\{1,3\})] + \frac{1}{12}[\, v(\{1,2,4\}) - v(\{1,4\})] \\ & + \frac{1}{12}[\, v(\{2,3,4\}) - v(\{3,4\})] + \frac{1}{4}[\, v(\{1,2,3,4\}) - v(\{1,3,4\})] \\ = {} & \frac{1}{4}\,[0] + \frac{1}{12}\,[0-0] + \frac{1}{12}\,[6.5-0] + \frac{1}{12}\,[6.5-0] + \frac{1}{12}\,[10-6.5] + \frac{1}{12}\,[10-6.5] \\ & + \frac{1}{12}\,[10-0] + \frac{1}{4}\,[13-10] = \frac{13}{4}\end{aligned}$$

$$\begin{aligned}\phi_3(N,v) = {} & \frac{1}{4}\, v(\{3\}) + \frac{1}{12}[\, v(\{1,3\}) - v(\{1\})] + \frac{1}{12}[\, v(\{2,3\}) - v(\{2\})] + \frac{1}{12}[\, v(\{3,4\}) - v(\{4\})] \\ & + \frac{1}{12}[\, v(\{1,2,3\}) - v(\{1,2\})] + \frac{1}{12}[\, v(\{1,3,4\}) - v(\{1,4\})] \\ & + \frac{1}{12}[\, v(\{2,3,4\}) - v(\{2,4\})] + \frac{1}{4}[\, v(\{1,2,3,4\}) - v(\{1,2,4\})] \\ = {} & \frac{1}{4}\,[0] + \frac{1}{12}\,[0-0] + \frac{1}{12}\,[6.5-0] + \frac{1}{12}\,[0-0] + \frac{1}{12}\,[10-0] + \frac{1}{12}\,[10-6.5] \\ & + \frac{1}{12}\,[10-6.5] + \frac{1}{4}\,[13-10] = \frac{13}{4}\end{aligned}$$

$$\phi_4(N,v) = \frac{1}{4}\, v(\{4\}) + \frac{1}{12}[\, v(\{1,4\}) - v(\{1\})] + \frac{1}{12}[\, v(\{2,4\}) - v(\{2\})] + \frac{1}{12}[\, v(\{3,4\}) - v(\{3\})]$$
$$+ \frac{1}{12}[\, v(\{1,2,4\}) - v(\{1,2\})] + \frac{1}{12}[\, v(\{1,3,4\}) - v(\{1,3\})]$$
$$+ \frac{1}{12}[\, v(\{2,3,4\}) - v(\{2,3\})] + \frac{1}{4}[\, v(\{1,2,3,4\}) - v(\{1,2,3\})]$$
$$= \frac{1}{4}\,[0] + \frac{1}{12}\,[6.5-0] + \frac{1}{12}\,[6.5-0] + \frac{1}{12}\,[0-0] + \frac{1}{12}\,[10-0] + \frac{1}{12}\,[10-6.5]$$
$$+ \frac{1}{12}\,[10-6.5] + \frac{1}{4}\,[13-10] = \frac{13}{4}$$

Por tanto, el valor de Shapley del juego es

$$\phi(N,v) = \left(\frac{13}{4}, \frac{13}{4}, \frac{13}{4}, \frac{13}{4}\right)$$

Todos los conceptos de solución que se han comentado tomaban a la coalición total N como la formada, sin embargo, eso no tiene porque ser. En caso que se forme una partición pueden aplicarse por separado. En el ejemplo de las *patentes* y con la regla igualitaria:

La representación del juego según las funciones características de las posibles coaliciones es

$v(\{1\}) = 0,\ v(\{2\}) = 0,\ v(\{3\}) = 0,\ v(\{4\}) = 0,$
$v(\{1, 2\}) = 0,\ v(\{3, 4\}) = 0$

$v(\{1, 3\}) = v(\{1, 4\}) = v(\{2, 3\}) = v(\{2, 4\}) = 6.5$

$v(\{1, 2, 3\}) = v(\{1, 2, 4\}) = 10$
$v(\{1, 3, 4\}) = v(\{2, 3, 4\}) = 10$

$v(\{1, 2, 3, 4\}) = 13$

Si las coaliciones que se forman son {1}, {2}, {3}, {1, 2}, {3, 4}, al ser sus funciones características 0, el pago que cada jugador de cada coalición recibe es igualmente 0.

$$\mathcal{E}_i(S,v) = \frac{v(S)}{|S|} = 0$$

Si las coaliciones que se forman son {1, 3}, {1, 4}, {2, 3}, {2, 4}, como sus funciones características valen 6.5 y el número de jugadores en cada coalición es $|S| = 2$ , el pago que cada jugador de cada coalición recibirá es:

$$\mathcal{E}_i(S, v) = \frac{v(S)}{|S|} = \frac{13}{4}$$

Si las coaliciones que se forman son {1, 2, 3}, {1, 2, 4}, {1, 3, 4}, {2, 3, 4}, como sus funciones características valen 10 y el número de jugadores en cada coalición es $|S| = 3$ , el pago que cada jugador de cada coalición recibirá es:

$$\mathcal{E}_i(S, v) = \frac{v(S)}{|S|} = \frac{10}{3}$$

Si la coalición que se forman es {1, 2, 3, 4} como su función característica vale 13 y el número de jugadores en cada coalición es $|S| = 4$ , el pago que cada jugador de cada coalición recibirá es:

$$\mathcal{E}_i(S, v) = \frac{v(S)}{|S|} = \frac{13}{4}$$

Según este método de distribución de los beneficios obtenidos por las posibles coaliciones, las coaliciones en las que cada jugador obtendría un mayor pago serían las siguientes:

{1, 2, 3}, {1, 2, 4}, {1, 3, 4}, {2, 3, 4}

## 2.3 Juegos Hedónicos

### 2.3.5. Introducción a juegos hedónicos

Como se comentó en la introducción del capítulo la resolución de un juego conlleva dos respuestas: qué coalición se forma y cómo se reparte el pago de las coaliciones formadas. En la teoría de juegos cooperativos, un juego hedónico (también conocido como juego de formación de coaliciones hedónicas) es un juego que modela la formación de coaliciones (grupos) de jugadores cuando los jugadores tienen preferencias sobre el grupo al que pertenecen. Un juego hedónico se especifica dando un conjunto finito de jugadores y, para cada jugador, una clasificación de preferencia sobre todas las coaliciones (subconjuntos) de jugadores a los que el jugador pertenece. El resultado de un juego hedónico consiste en una partición de los jugadores en coaliciones disociadas, es decir, a cada jugador se le asigna un grupo único. Estas particiones se denominan a menudo estructuras de coalición.

Los juegos hedónicos propuestos por Drèze y Greenberg [5] son un tipo de juego cooperativo de utilidad intransferible. Un juego de utilidad intransferible describe las posibilidades que tiene una coalición como un conjunto de resultados, en el que cada resultado especifica el pago a cada agente de la coalición. Estos posibles resultados pueden ser considerados como diferentes formas de organización de la coalición, lo que a su vez puede derivar en diferentes beneficios para los miembros de la coalición. Un resultado se representa como un vector de pago que asigna un pago a cada miembro de la coalición.

La característica distintiva de las preferencias hedónicas es que a cada agente sólo le importa qué agentes están en su coalición, pero no le importa cómo se agrupan los agentes de otras coaliciones. Por lo tanto, las preferencias hedónicas ignoran completamente las dependencias intercoalicionales.

La descripción de un juego hedónico puede ser exponencialmente grande en el número de agentes, ya que para cada agente debe describirse las preferencias de este agente sobre todas las posibles coaliciones a las que pueda pertenecer.

**Definición 2.12.** Sea $N = \{1, \ldots, n\}$ un conjunto de agentes o jugadores y $N_i = \{ S \subseteq N : i \in S\}$ el conjunto de todas las coaliciones que contienen al jugador $i \in N$. Un **juego hedónico** de formación de coaliciones es una pareja $(N, (\succsim_i)_{i \in N})$, donde $\succsim_i$ es un perfil de preferencia que especifica para el agente $i \in N$ una relación binaria reflexiva, completa y transitiva, $\succsim_i$ sobre $N_i$. Lamamos $\succsim_i$ a una relación de preferncia.

**Definición 2.13.** La solución de un juego hedónico $(N, (\succsim_i)_{i \in N})$, es una **estructura de coalición** $\pi$, esto una partición del conjunto *N* en coaliciones distintas 2 a 2.

**Definición 2.14.** Dada la estructura de una coalición, una **desviación** de un solo agente es una maniobra de este agente para abandonar su coalición actual y unirse a una coalición diferente o quedarse solo. Decimos que un agente tiene un incentivo para desviarse si existe una desviación de este agente a una nueva coalición o quedarse solo que prefiere a su antigua coalición; llamamos a tal desviación una desviación rentable.

### 2.3.6. Ejemplos de juegos hedónicos

En el ejemplo se va a utilizar la notación siguiente: Para dos coaliciones $S, T \in N_i$ que contienen al agente *i*, usamos $S \succ_i T$ para indicar que *i* prefiere estrictamente S sobre T, $S \sim_i T$ para indicar que *i* es indiferente entre *S* y *T*, y $S \succsim_i T$ para indicar tanto $S \succ_i T$ como $S \sim_i T$ son validos. Dada una estructura de coalición π usamos π(i) para denotar la (única) coalición de π que incluye al agente i.

*Dos es compañía, tres son multitud*

Siendo las preferencias individuales:

$$\{1, 2\} \succ_1 \{1, 3\} \succ_1 \{1\} \succ_1 \{1, 2, 3\}$$

$$\{2, 3\} \succ_2 \{1, 2\} \succ_2 \{2\} \succ_2 \{1, 2, 3\}$$

$$\{1, 3\} \succ_3 \{1, 2\} \succ_3 \{3\} \succ_3 \{1, 2, 3\}$$

Estas preferencias tienen un ciclo: el primer jugador prefiere al segundo jugador antes que al tercero, el segundo prefiere al tercer jugador antes que al primero y el tercer jugador prefiere al primero antes que al segundo. Todos los jugadores prefieren estar en pareja antes que estar todos juntos y estar solos es el peor resultado.

*El juego del invitado no deseado*

Siendo las preferencias individuales:

$$\{1, 2\} \succ_1 \{1\} \succ_1 \{1, 2, 3\} \succ_1 \{1, 3\}$$

$$\{1, 2\} \succ_2 \{2\} \succ_2 \{1, 2, 3\} \succ_2 \{2, 3\}$$

$$\{1, 2, 3\} \succ_3 \{2, 3\} \succ_3 \{1, 3\} \succ_3 \{3\}$$

A partir de estas preferencias, podemos ver que 1 y 2 quieren juntarse, pero no les gusta la presencia del jugador 3.

### 2.3.7. Estabilidad

Los jugadores en los juegos hedónicos son típicamente entendidos como auto-interesados, y por lo tanto los juegos hedónicos son usualmente analizados en términos de la estabilidad de las estructuras de coalición, donde se usan varias nociones de estabilidad, incluyendo el core y la estabilidad de Nash. Los juegos hedónicos se estudian tanto en economía, donde la atención se centra en identificar las condiciones suficientes para la existencia de resultados estables, como en sistemas multiagente, donde la atención se centra en identificar representaciones concisas de los juegos hedónicos y en la complejidad computacional de encontrar resultados estables.

Al igual que en otras áreas de la teoría de juegos, los resultados de los juegos hedónicos se evalúan utilizando conceptos de solución. Muchos de estos conceptos se refieren a una noción de estabilidad teórica del juego: un resultado es estable si ningún jugador (o posiblemente ninguna coalición de jugadores) puede desviarse del resultado para alcanzar un resultado subjetivamente mejor. La estabilidad es el principal criterio que se ha utilizado para analizar qué estructuras de coalición se formarán. Para ello se exponen los diferentes conceptos de estabilidad para los juegos hedónicos.

**Definición 2.15.** Una partición $\pi$ , es **individualmente racional** (IR) si cada agente prefiere su coalición a estar solo, i.e., para todo $i \in N$, $\pi(i) \gtrsim_i \{i\}$. Esto es realmente un requisito mínimo para que una solución sea considerada estable, y muchos de los conceptos de solución que consideramos satisfacen la racionalidad individual.

**Definición 2.16.** Una **partición perfecta** es aquella en la que cada agente está en una de sus coaliciones preferidas.

**Definición 2.17.** Si se cumple la **optimizacición de Pareto** (PO) no existe ninguna partición que cada agente prefiera débilmente y al menos un agente la prefiere estrictamente.

**Definición 2.18.** Una partición es **Nash estable** (NS) si ningún jugador puede beneficiarse moviéndose de su coalición a otra (posiblemente vacía) coalición *T*.

**Definición 2.19.** Una partición es **individualmente estable** (IS) si ningún jugador puede beneficiarse moviéndose de su coalición a otra coalición T existente (posiblemente vacía) sin empeorar la situación de los miembros de *T*.

**Definición 2.20.** Una partición es **contractual Nash estable** (CNS) si ningún agente tiene un incentivo para mudarse a otra coalición sin empeorar la situación de ningún miembro de la antigua coalición.

**Definición 2.21.** Una partición es **contractual estable individualmente** (CIS) si ningún agente puede beneficiarse al pasar de su coalición a otra existente (posibemente vacia) mientras no empeora a ningún miembro de ninguna de las dos coaliciones.

Conceptos de soluciones basados en la estabilidad del grupo:

**Definición 2.22.** Una coalición $S \subseteq N$ **bloquea** una partición $\pi$, si cada jugador $i \in S$ prefiere estrictamente S a su actual coalición $\pi(i)$ en la partición $\pi$. Una partición que no admite ninguna coalición de bloqueo se dice que está en el **núcleo** (C).

**Definición 2.21.** Una coalición $S \subseteq N$ **bloquea débilmente** una partición $\pi$, si cada jugador $i \in S$ prefiere débilmente $S$ a $\pi(i)$ y existe al menos un jugador $j \in S$ que prefiere estrictamente $S$ a su coalición actual $\pi(j)$. Una partición que no admite ninguna coalición que bloquee débilmente está en el **núcleo estricto** (SC).

También hay otros conceptos de estabilidad basados en desviaciones más complejas por parte de coaliciones de agentes. Si una partición no es fuertemente Nash bloqueada por cualquier conjunto $S \subseteq N$, se llama Nash fuerte estable (SNS). El concepto de estabilidad Nash fuerte estable puede ser convenientemente debilitado o fortalecido para obtener otros conceptos de estabilidad como la estabilidad Nash fuerte estricta (SSNS) y la estabilidad individual fuerte (SIS).

Todos los conceptos anteriores (excepto la estabilidad contractual de Nash) han sido establecidos por Bogomolnaia y Jackson (2002). En la Figura 2.2, se muestran las relaciones entre estos y otros conceptos de solución. Por ejemplo, la figura muestra que si una partición es perfecta, entonces satisface todos los conceptos de estabilidad definidos anteriormente.

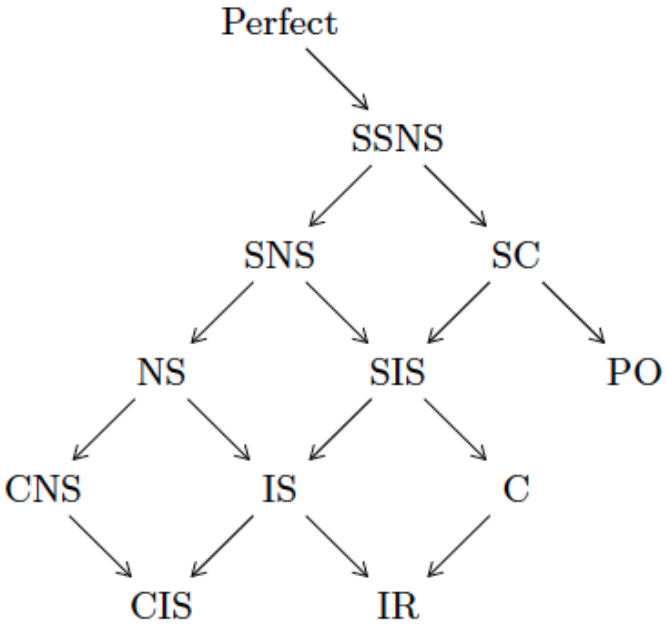


Figura 2.2. Relaciones de inclusión entre los conceptos de estabilidad para los juegos hedónicos. Por ejemplo, cada paridad de NS es también IS. NS, SC, PO.

Si aplicamos estos conceptos al ejemplo del *invitado no deseado* se puede comprobar que si consideramos la partición $\pi = \{\{1, 2\}, \{3\}\}$. Se observa que en $\pi$, el jugador 3 preferiría unirse a la coalición $\{1, 2\}$, ya que $\{1, 2, 3\} \succ_3 \{3\}$ y por lo tanto $\pi$ no es Nash estable. Sin embargo, si el jugador 3 se uniera a $\{1, 2\}$, el jugador 1 (y también el 2) se vería empeorado por esta desviación, y así la desviación del jugador 3 cumpliría la estabilidad individual. De hecho, se puede comprobar que $\pi$ es individualmente estable. Podemos además ver que no hay ningún grupo $S \subseteq N$ de jugadores de tal manera que cada miembro de *S* prefiere *S* a su coalición en $\pi$ y por lo tanto la partición también está en el núcleo.

Por otro lado, en el ejemplo de *dos es compañía, tres son multitud* el núcleo está vacio mientras que $\{1, 2, 3\}$ es la única partición estable de Nash, además de ser la única partición individualmente estable.

# 3 FORMACIÓN DE COALICIONES HEDÓNICAS ENTRE ROBOTS INALÁMBRICOS

## 3.1 Introducción

El uso de agentes autónomos se ha centrado principalmente en la robótica, sistemas informáticos o ingeniería de software; por el contrario, son pocos los modelos que abordan el uso de estos agentes en redes inalámbricas y de comunicación. Sin embargo, la necesidad reciente de estos agentes autónomos en las redes inalámbricas ha adquirido una importancia notable. Una de las aplicaciones donde se pone de manifiesto la necesidad de estos robots inalámbricos es en el desarrollo de tareas de seguridad y vigilancia. Así, este trabajo se centrará en el patrullaje multi-robot, consistente en un conjunto finito de robots que deben recorrer repetidamente una infraestructura dada, comprobando que no haya intrusos en ciertos puntos de interés (las llamadas tareas).

Por ello, el modelo que se propone considera un número de agentes inalámbricos que deben recolectar los datos de tareas localizadas. Cada tarea representa una fuente de datos, es decir, en el problema del patrullaje representa un lugar por el cual el robot tiene que pasar, una vez allí recoger la información y luego transmitirla a través de un enlace inalámbrico al receptor central. Así, modelamos el problema de asignación de tareas como un juego hedónico de formación de coaliciones entre los agentes y las tareas, e introducimos un algoritmo para formar coaliciones. A medida que los robots, ya sea individualmente o en grupos, se enfrentan a nuevas tareas para las que los recursos individuales o de grupo no son suficientes, es necesario formar coaliciones de robots que sean capaces de cumplir estos requisitos de forma colectiva.

Para formar coaliciones, los agentes y las tareas pueden tomar autónomamente la decisión de unirse o abandonar una coalición basada en relaciones de preferencia individuales. Estas preferencias se basan en una función de valor de coalición que tiene en cuenta el beneficio recibido por el servicio de una tarea, en términos de rendimiento efectivo (datos recopilados), así como el costo en términos de retraso del sistema que se produce a partir del tiempo necesario para el servicio de todas las tareas de una coalición.

Como resultado de estas decisiones, los agentes y las tareas se estructuran en coaliciones independientes y disociadas que constituyen una partición de red estable de Nash. También se tiene en cuenta la capacidad de los agentes a autoadaptarse a los cambios ambientales como el despliegue de nuevas tareas y la eliminación de tareas existentes. Por lo tanto, a medida que se enfrentan a nuevas tareas, las coaliciones se fusionan y se dividen para que las coaliciones resultantes sean capaces de realizar las tareas recién encontradas. En la mayoría de los entornos, se supone que todas las tareas se conocen inicialmente, a diferencia de las tareas dinámicas en las que las condiciones ambientales fluctúan mucho, de modo que sólo es posible la asignación instantánea.

## 3.2 Modelo del Sistema

Se considera una red con *M* agentes inalámbricos, que pertenecen a un solo operador de red, los cuales están controlados por un comando central (por ejemplo, un nodo central de control o un sistema satelital). Estos agentes estarán obligados a ocuparse de una serie de tareas ubicadas en un área geográfica que tiene un receptor inalámbrico central asociado conectado al centro de control. Se indica el conjunto de agentes y tareas por $M = \{1, \ldots, m\}$ y $T = \{1, \ldots, n\}$, respectivamente. Consideramos sólo el caso en el que el número de tareas es mayor que el número de agentes, por lo tanto $T > M$. Las tareas que consideramos son fuentes de datos que no pueden enviar su información al receptor central (y, posteriormente, al centro de control) sin la ayuda de un agente. Estas tareas pueden representar un grupo de dispositivos móviles, tales como sensores, dispositivos de videovigilancia o cualquier otro nodo estático o dinámico inalámbrico que tenga una potencia limitada y no pueda proporcionar una transmisión de larga distancia. Estos dispositivos (tareas) necesitan almacenar sus datos localmente y esperar a ser atendidos por un agente que pueda recopilarlos.

Cada tarea $i \in T$ representa un sistema de colas M/D/1, en el que los paquetes de tamaño constante generados tienen una tasa media de llegada de $\lambda_i$. El sistema de colas M/D/1 se basa en que la tasa de llegada de los paquetes sigue una distribución de Poisson, significando una distribución exponencial para los tiempos entre llegadas y que además, dispone de un único servidor. Cada agente $i \in M$ ofrece una capacidad de transmisión de enlace de $\mu_i$ (en paquetes/segundo) mediante la cual el agente puede dar servicio a los datos de las tareas. Por último, la cantidad $1/\mu i$ representa el tiempo de servicio conocido para un solo paquete que está siendo atendido por el agente *i*. Además, cada agente puede transmitir los datos al receptor con una potencia máxima de transmisión de $P_i = \tilde{P}$ , asumiendo lo mismo para todos los agentes sin pérdida de generalidad.

Cada tarea podrá ser atendida por múltiples agentes, y también, cada agente (o grupo de agentes) pueden atender multiples tareas. Cualquier grupo de agentes que actúen juntos para la recogida de datos de la misma tarea, puede ser visto como un único recolector con capacidad de transmisión de enlace mejorada. Se considera que la capacidad de transmisión del enlace depende únicamente de las capacidades de los agentes y no de la naturaleza de las tareas. Por ello, dado un grupo de agentes $G \subseteq M$ que actúan como recolectores para cualquier tarea, la capacidad total de transmisión de enlaces con los que $G$ puede atender a las tareas viene dado por

$$\mu^G = \sum_{j \in G} \mu_j \tag{3-1}$$

La transmisión de los paquetes por los agentes desde una tarea $i \in T$ al receptor central está sujeta a la pérdida de paquetes debido al desvanecimiento en el canal inalámbrico. Por lo tanto, además de actuar como recolectores, algunos agentes pueden actuar como enlaces para una tarea. Estos agentes de enlace se ubicarían a igual distancia de la tarea (dado que la tarea ya está siendo atendida por al menos un recolector), y, por lo tanto, los recolectores transmiten los datos al receptor a través de múltiples agentes, lo que aumenta la probabilidad de que la transmisión tenga éxito. La probabilidad de conseguir una transmisión exitosa de un paquete de tamaño B bits de los recolectores presentes en una tarea $i \in T$ a través de una ruta de m agentes, $Q_i = \{i_1, \dots, i_m\}$, donde $i_1 = i$ es la tarea que está siendo atendida y $i_m$ el receptor central viene dada por:

$$Pr_{i,CR} = \prod_{h=1}^{m-1} Pr^{B}_{i_h, i_{h+1}} \tag{3-2}$$

Donde $Pr^{B}_{i_h, i_{h+1}}$ es la probabilidad de la transmisión exitosa de un solo bit desde el agente $i_h$ al agente (o el receptor central) $i_{h+1}$.

Los recolectores se moverían de una tarea a otra, sólo si todos los paquetes de la cola en la tarea actual han sido transmitidos al receptor (el proceso a través del cual los agentes se mueven de una tarea a otra para la recolección de datos es cíclico). Según lo dicho anteriormente, la red final estará formada por grupos de tareas atendidas por grupos de agentes, de forma continua. En la Fig 3.1 se muestra una ilustración de este modelo. Según el modelo, el objetivo principal es proporcionar un algoritmo para distribuir las tareas entre los agentes, teniendo en cuenta el funcionamiento de los agentes descritos anteriormente y mostrados en la Fig 3.1.

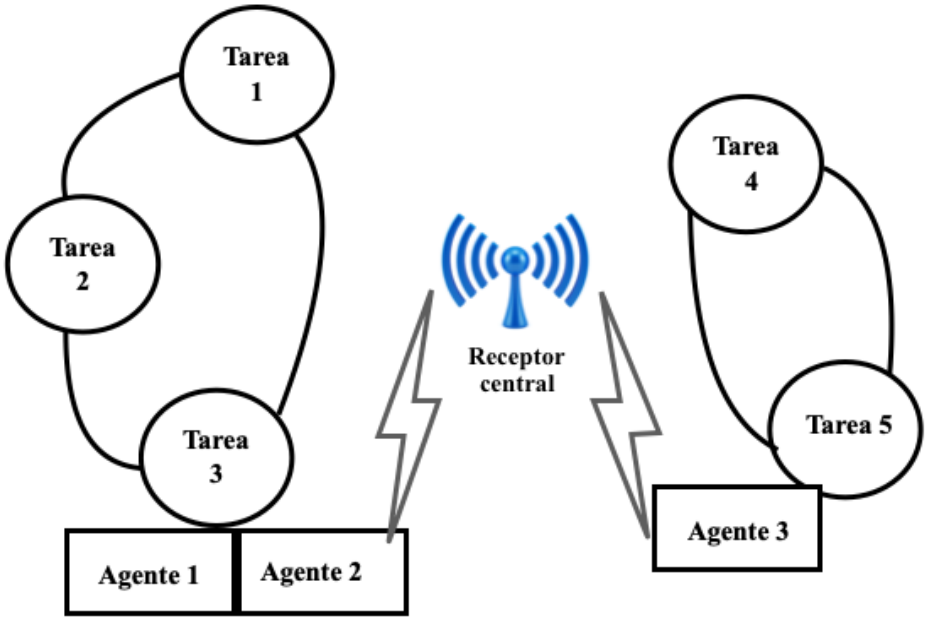


Figura 3.1. Modelo propuesto para la asignación de tareas en redes inalámbricas.

En el modelo propuesto en la figura 3.1 se puede observar una red compuesta por 5 tareas y tres agentes encargados en dar servicio a las tareas. Una vez creadas las coaliciones necesarias para la atención de estas tareas se puede ver como se han formado dos coaliciones, la primera formada por el agente 1, el agente 2 y las tareas 1, 2 y 3 y la segunda formada por el agente 3 y las tareas 4 y 5. Los agentes de cada coalición visitarán las tareas de su coalición transmitiendo la información de estas al receptor central.

## 3.3 Juego de coalición

### 3.3.1 Formulación del juego

Fijándonos en la Fig. 3.1, se puede ver claramente que el problema de la asignación de tareas entre los agentes se puede traducir en el problema de la formación de coaliciones. Para el modelo propuesto, el juego de coalición se juega entre los agentes y las tareas. Por lo tanto, los jugadores establecidos para el juego de coalición de asignación de tareas propuesto se denotan por $N$, y contiene agentes y tareas, es decir, $N = M \cup T$. En lo sucesivo, utilizaremos el término jugador para indicar una tarea o un agente.

Este esquema de servicio de tareas propuesto puede ser asignado a un concepto bien conocido que es ampliamente estudiado en los sistemas informáticos, el concepto de sistema de votación [7] (o sistema de polling). En un sistema de votación, un único servidor se mueve entre múltiples colas para extraer los paquetes de cada cola, de forma secuencial y cíclica, repitiendo así su ruta continuamente. El término polling hace referencia al sondeo que realiza el servidor central para acceder a la información y para comprobar en que estado se encuentra cada estación (en nuestro caso las tareas), las estaciones acumulan mensajes en sus colas, pero no los transmiten hasta que son encuestadas. Las transmisiones entre estaciones se realizan a través del ordenador central, que recibe todos los paquetes entrantes. Siempre que los agentes se detienen en cualquier tarea $i \in S$, recogen y transmiten los datos presentes en esta tarea hasta que la cola se vacía. Además, el tiempo que tarda el servidor en pasar de una cola a otra se conoce como tiempo de conmutación. En consecuencia, se destacan las siguientes propiedades:

**Propiedad 3.1.** En el modelo de asignación de tareas propuesto, cada coalición $S \subseteq N$ es un sistema de votación con una estrategia de votación exhaustiva y tiempos determinantes de conmutación distintos de cero. En cada uno de estos sistemas de votación $S$, los agentes recolectores son vistos como el servidor del sistema de votación, y las tareas son las colas que los agentes recolectores deben mantener.

**Propiedad 3.2**. Dentro de cualquier coalición $S$, el tiempo de conmutación entre dos tareas corresponde al tiempo constante que tarda uno de los recolectores en pasar de una de las tareas a la siguiente.

Para cualquier coalición *S*, una vez que se vacía la cola de una tarea $i \in S$, los recolectores de una coalición pasan de la tarea *i* a la siguiente tarea $j \in S$ con una velocidad constante η, con lo que se produce un tiempo de conmutación $\theta_{i,j}$. Asumiendo que los agentes inician su movilidad al mismo tiempo, este tiempo de conmutación se refiere al tiempo necesario para que el agente más lejano se mueva de una tarea a la siguiente. Igualmente se producen retrasos los cuales debemos analizar, sin embargo, encontrar las expresiones exactas del retardo en cada cola es una tarea complicada. Un criterio clave para el análisis del retardo de un sistema de votación es la ley de pseudo-conservación que proporciona expresiones para la suma ponderada de los medios de los tiempos de espera en las colas [7], [8]. Para proporcionar la ley de pseudo-conservación para una coalición $S \subseteq N$ compuesta por un número de agentes y un número de tareas, se introducen las siguientes definiciones:

**Deinición 3.1.** Dentro de la coalición *S*, un grupo de agentes $G_S \subseteq S \cap M$ son designados como **recolectores**, encargados de dar servicio a las tareas.

**Deinición 3.2**. Por cada tarea $i \in S \cap T$ con una tasa media de llegada de $\lambda_i$, y atendida por un número de recolectores $|G_S|$ con una capacidad de transmisión de enlace de $\mu^{Gs}$ (como se indica en (3-1)), se define el **factor de utilización de la tarea** *i* como

$$\rho_i = \frac{\lambda_i}{\mu^{Gs}} \tag{3-3}$$

Y el **factor de utilización de todas las tareas**

$$\rho_S = \sum_{i \in S \cap T} \rho_i \tag{3-4}$$

Por tanto, para una coalición *S*, la suma ponderada de los tiempos de espera de los agentes en todas las tareas de la coalición viene dada por la ley de pseudo-conservación de la siguiente manera.

$$\sum_{i \in S \cap T} \rho_i \overline{W}_i = \rho_S \frac{\sum_{i \in S \cap T} \frac{\rho_i}{\mu^{Gs}}}{2(1-\rho_S)} + \rho_S \frac{\theta_S^2}{2} + \frac{\theta_S}{2(1-\rho_S)} \left[ \rho_S^2 - \sum_{i \in S \cap T} \rho_i^2 \right] \tag{3-5}$$

Donde $\overline{W}_i$ es el tiempo medio de espera en la tarea *i* y $\theta_S = \sum_{h=i}^{|S \cap T|} \theta_{i_h, i_{h+1}}$ es la suma de los tiempos de conmutación dados a una ruta de tareas $\{i_1, \ldots, i_{|S \cap T|}\}$ seguida por los agentes. El primer término en la parte derecha de la ecuación (3-5) corresponde al retardo medio de espera para colas M/D/1, ponderado por $\rho_S$ . El segundo y tercer termino en el lado derecho de la ecuación (3-5) representan el aumento medio de retardo que se produce debido al tiempo de desplazamiento necesario para que los colectores pasen se una tarea a otra. Además, para cualquier coalición *S* que se forme en el sistema, debe cumplir la siguiente condición:

$$\rho_S < 1 \tag{3-6}$$

Esta condición es requisito para la estabilidad de cualquier sistema de votación [7] y, por lo tanto, debe ser cumplida por cualquier coalición que se forme en el modelo propuesto. En caso de que se incumpla esta condición, el sistema se considera inestable y el retraso se considera infinito.

### 3.3.2 Función de utilidad

Para cada coalición $S \subseteq N$, los agentes deben determinar el orden en que las tareas en $S$ son visitadas, es decir, el camino $\{i_1, \ldots, i_{|S \cap T|}\}$. Como es lógico, los agentes deben seleccionar la ruta que minimice el tiempo total de conmutación para una sola ronda de recolección de datos. Por lo que se plantea la siguiente propiedad:

**Propiedad 3.3.** El problema de encontrar el camino que minimiza el tiempo total de conmutación para una ronda de recolección de datos dentro de una coalición $S \subseteq N$ se asocia al problema de los vendedores ambulantes [9], donde se requiere de un vendedor, es decir, los agentes $S \cap M$, que minimizan el tiempo de visita a una serie de ciudades, es decir, las tareas $S \cap T$.

El problema del vendedor ambulante responde a la siguiente pregunta: dada una lista de ciudades y las distancias entre cada par de ellas, ¿cuál es la ruta más corta posible que visita cada ciudad exactamente una vez y al finalizar, regresa a la ciudad origen? En la teoría de la complejidad computacional, la toma de decisión del problema del vendedor ambulante pertenece a la clase de problemas NP-completos. Por tanto, es probable que en el peor caso el tiempo de ejecución para cualquier algoritmo que resuelva el problema aumente de forma exponencial con respecto al número de ciudades. Aunque el problema es computacionalmente complejo, una gran cantidad de algoritmos heurísticos y métodos exactos son conocidos, de manera que, se puede llegar a una solución aceptable casi óptima.

Uno de los algoritmos más simples es el algoritmo del vecino más cercano (el cual es por construcción un algoritmo voraz) [9]. En este algoritmo, a partir de una ciudad determinada, el vendedor elige la ciudad más próxima como su siguiente visita. Utilizando el algoritmo del vecino más cercano, se selecciona el orden de las ciudades que minimiza la ruta total. Este es subóptimo, pero puede rápidamente encontrar una solución casi óptima con una pequeña complejidad computacional (lineal en el número de ciudades) que la hace adecuada para problemas como el problema de asignación de tareas propuesto. Por lo tanto, en el modelo expuesto, para cada coalición *S*, los agentes pueden calcular fácilmente la ruta vecina más cercana para las tareas y operar de acuerdo con ella.

Habiendo modelado cada coalición como un sistema de votación, la ley de pseudo-conservación en (3-5) permite evaluar el costo, en términos de tiempo promedio de espera (o retraso), de formar una coalición particular. Sin embargo, por cada coalición, hay un beneficio, en términos del rendimiento efectivo promedio que la coalición es capaz de lograr. El rendimiento medio efectivo de una coalición *S* viene dado por

$$L_s = \sum_{i \in S \cap T} \lambda_i \ Pr_{i,CR} \tag{3-7}$$

Con $Pr_{i,CR}$ dado por (3-2). Si nos fijamos en la ecuación (3-1) se puede ver que la adición de más colectores mejora la capacidad del enlace de transmisión y, por lo tanto, reduce el tiempo de servicio que recibe una determinada tarea. Basándose en esta propiedad y utilizando (3-5) se puede ver fácilmente que, añadiendo más colectores, es decir, mejorando el tiempo de servicio, se reduce el retardo total en (3-5).

En consecuencia, utilizando (3-7), se puede ver que esta mejora en la probabilidad de una transmisión exitosa se traduce en una mejora en el rendimiento efectivo. Por lo tanto, cada agente posee su propio benefio para la coalición.

Al existir una relación de compensación entre el rendimiento y el retardo se hace uso del concepto de potencia del sistema para caracterizar la utilidad de la red.

**Deinición 3.3.** Llamamos **potencia del sistema** a la relación entre cierta potencia del rendimiento y el retardo (o la potencia del retardo).

Por lo tanto, el concepto de potencia es una noción que permite capturar el equilibrio fundamental entre el rendimiento y el retardo en el modelo de asignación de tareas propuesto**.** Para el juego planteado, la utilidad de cada coalición *S* se evalúa utilizando una función de valor de coalición basada en el concepto de poder [10] de la siguiente manera:

$$v(S) = \begin{cases} \delta \dfrac{L_S^{\beta}}{\left(\sum_{i \in S \cap T} \rho_i \overline{W}_i\right)^{(1-\beta)}}, & si\ \ \rho_S < 1\ and\ |S| > 1 \\ 0, & otros \end{cases} \tag{3-8}$$

Donde $\beta \in (0,1)$ es un parámetro de compensación entre el rendimiento y el retardo. El término δ representa el coste por unidad de potencia que la red ofrece a la coalición *S.* Por lo tanto, δ representa un parámetro de control genérico que permite al operador de red monitorear de alguna manera el comportamiento de los jugadores.

Por tanto, la función de utilidad en (3-8) representaría los ‘ingresos’ totales logrados por una coalición *S*, dado el poder de red que obtiene la coalición *S*. Para las coaliciones que consisten en un solo agente o una sola tarea, es decir, coaliciones de tamaño 1, la utilidad asignada se debe al hecho de que tales coaliciones no generan beneficios para sus miembros (un solo agente no puede recopilar datos a menos que se desplace a la tarea, mientras que una sola tarea no puede transmitir ninguno de los datos generados sin que un agente recopile estos datos).

Dado el conjunto de jugadores *N*, y la función de valor en (3-8), se define un juego de coalición *(N, v)* con la utilidad transferible. La utilidad en (3-8) representa la cantidad de dinero o ingresos recibidos por una coalición y, por lo tanto, esta cantidad puede ser repartida arbitrariamente entre los miembros de la coalición, lo que indica la naturaleza UT del juego. Para dividir esta utilidad entre los jugadores, podemos adoptar diferentes métodos de distribucíon como ya vimos en el capítulo anterior, estos son por ejemplo el valor de Shapley y la regla de distribución igualitaria, donde el pago que recibiría cada jugador $i \in S$ sería:

Con el valor de Shapley:

$$x_i^S(v) = \sum_{S \in P(N)} q^N(S)\,[v(S) - v(S - \{i\})] \tag{3-9}$$

en donde

$$q^N(S) = \frac{(s-1)\,!\,(n-s)!}{n!}$$

Con la distribución igualitaria:

$$x_i^S(v) = \frac{v(S)}{|S|} \tag{3-10}$$

Como se ve en (3-5) y (3-8), cuando el número de tareas en una coalición aumenta, el retraso total aumenta, por lo tanto, se reduce la utilidad de formar una coalición. Además, en una coalición en la que el número de tareas es grande, la condición de estabilidad del sistema de votación, tal como se indica en (3-6), puede ser violada debido a la gran cantidad de tareas que producen una utilidad nula según (3-8). Por lo tanto, formar coaliciones entre las tareas y los agentes implica un coste que puede limitar el tamaño de una coalición. A este respecto, los conceptos de solución tradicionales para los juegos de UT, como el core [11], pueden no ser aplicables. De hecho, para que exista el core, un juego de coalición de UT debe asegurar que se forme la gran coalición, es decir, la coalición de todos los jugadores. Sin embargo, como se ve en la Fig. 3.1 y corroborado por la utilidad en (3-8), en general, debido al costo para la formación de la coalición, la gran coalición no se formará. En cambio, las coaliciones independientes y disociadas aparecen en la red como resultado del proceso de asignación de tareas. El objetivo del juego es encontrar un algoritmo que permite formar la estructura de la coalición, en lugar de encontrar sólo un concepto de solución, como el core, que tiene como objetivo principal estabilizar una gran coalición de todos los jugadores.

## 3.4 Juego de formación de coaliciones hedónicas

### 3.4.1 Conceptos y modelo

Como se comentó anteriormente, modelamos el sistema propuesto como un juego de formación de coaliciones hedónicas. Esta clase de juegos tiene varias propiedades interesantes que pueden ser aplicadas en redes inalámbricas. En consecuencia, dos requisitos clave para clasificar un juego de coalición como juego hedónico son:

**Condición 1:** El beneficio de cualquier jugador depende únicamente de los miembros de la coalición a la que pertenece el jugador.

**Condición 2**: Las coaliciones se forman como resultado de las preferencias de los jugadores sobre el conjunto de sus posibles coaliciones.

Estas dos condiciones caracterizan el marco de los juegos hedónicos. Principalmente, el término hedónico se refiere a la primera condición, por lo que la recompensa de cualquier jugador *i*, en un juego hedónico, debe depender solo de la identidad de los jugadores en la coalición a la que pertenece el jugador, sin depender de los otros jugadores. Para la segunda condición, definiremos formalmente cómo se pueden usar las preferencias de los jugadores sobre las coaliciones para el proceso de formación. Como ya se introdujo en el capítulo anterior, en un juego hedónico, cada jugador debe construir preferencias sobre su propio conjunto de posibles coaliciones. Por lo que, cada jugador debe ser capaz de comparar las coaliciones y ordenarlas en función de la coalición a la que prefiere pertenecer.

Usamos, por tanto, un juego de formación de coaliciones hedónicas para proporcionar una solución adecuada al problema de asignación de tareas propuesto. Este problema es modelado como un juego hedónico $(N, \succ)$ donde *N* es el conjunto de agentes y tareas y $\succ$ es un perfil de preferencias. Para el modelo de juego propuesto, dada una partición de red $\Pi$ de *N*, el beneficio de cada jugador *i* depende únicamente de los jugadores de la coalición a la que pertenece $S_\Pi(i)$. Por lo tanto, nuestro juego verifica la primera condición hedónica.

Además, para modelar el problema planteado como un juego hedónico de formación de coaliciones, las relaciones de preferencia de los jugadores deben ser claramente definidas. En este sentido, determinamos dos tipos de relaciones de preferencia, un primer tipo adecuado para indicar las preferencias de los agentes, y un segundo tipo adecuado para las tareas. Posteriormente, para evaluar las preferencias de cualquier agente $i \in M$, determinamos la siguiente operación (esta relación de preferencia es común para todos los agentes, por lo que la denotaremos como $\succsim_i = \succsim_M,\ \forall i \in M$)

$$S_2 \succsim_M S_1 \iff u_i(S_2) \geq u_i(S_1) \tag{3-11}$$

Donde $S_1 \subseteq N$ y $S_2 \subseteq N$ son dos coaliciones cualesquiera que contienen al agente *i*, es decir, $i \in S_1, i \in S_2$ y $u_i: 2^N \to \mathbb{R}$ es la función de preferencia definida por cualquier agente *i* definida como:

$$u_i(S) = \begin{cases} \infty, & si\ S = S_\Pi(i)\ y\ S \setminus \{i\} \subseteq T, \\ 0, & si\ S \in h(i), \\ x_i^S, & otros, \end{cases} \tag{3-12}$$

Donde Π es la partición de coalición actual que se establece en el juego, $x_i^S$ es el pago recibido por el jugador *i* de cualquier división de la función de valor entre los jugadores de la coalición S tal como se expone en la repartición mediante el valor de Shapley en (3-9) o el reparto igualitario en (3-10) y *h(i)* es el historial del jugador *i*. En cualquier momento, el historial *h(i)* es un conjunto que contiene las coaliciones de las que el jugador *i* ha formado parte en el pasado, antes de la formación de la partición actual Π.

En este modelo se usa la función de preferencia expuesta en (3-12) ya que, los agentes, al ser entidades propiedad del operador, buscan alcanzar dos objetivos:

1) Atender todas las tareas de la red para el beneficio del operador
2) Maximizar la calidad del servicio, en términos de potencia según (3-8), para extraer los datos de las tareas

La función de preferencia debe ser capaz de permitir a los agentes tomar decisiones de formación de coaliciones que puedan recoger el equilibrio entre la atención de todas las tareas (en beneficio del operador) y la obtención de una buena calidad de servicio (en beneficio tanto de los agentes como del operador). Para este propósito, según (3-12), cualquier agente *i* que sea el único agente que realice tareas en su coalición actual $S = S_\Pi(i)$ de tal forma que $S_\Pi(i) \cap M = \{i\}$, asigna un valor infinito de preferencia a *S*. Por lo tanto, para poder realizar todas las tareas, el agente siempre asignará una preferencia máxima a su coalición actual, si esta coalición actual está compuesta sólo por tareas y no contiene otros agentes. Este caso de la función de preferencia $u_i$ prohíbe al agente abanonar un grupo de tareas, ya asignadas a él, desatendidas por otros agentes. En este contexto, esta situación se refiere al primer objetivo del agente antes comentado, que conlleva siempre que exista riesgo de dejar tareas sin servicio, que el agente no actúa egoístamente, en cambio, actúará en beneficio del operador y permanecerá con estas tareas, independientemente de la retribución generada por ellas. Esta decisión permite a un agente evitar tomar una decisión que puede implicar el riesgo de tener en última instancia tareas sin servicio en la red, en cuyo caso, el operador de red perdería ingresos por estas tareas desatendidas. De lo contrario, la relación de preferencia de los agentes pondría de manifiesto el segundo objetivo del agente, es decir, maximizar su propio beneficio, es decir, la potencia según (3-8) con la cual se están realizando las tareas.

En este caso, la relación de preferencia es fácilmente generada por los agentes al comparar el valor de los pagos que reciben de las dos coaliciones $S_1$ y $S_2$. Además, se observa que ningún agente tiene ningún incentivo para volver a visitar una coalición que había dejado previamente, y, por lo tanto, los agentes asignan un valor de preferencia de 0 a cualquier coalición en su historia (esto puede ser visto como un proceso de aprendizaje básico). En resumen, teniendo en cuenta los objetivos de los agentes, entre dos coaliciones $S_1$ y $S_2$, un agente prefiere la coalición que da el mejor resultado, dado que el agente no está solo en su coalición actual, y la coalición con un mejor resultado no está en la historia del agente *i*.

Para evaluar las preferencias de cualquier tarea $j \in T$, se define la siguiente operación (esta relación de preferencia es común para todas las tareas, por lo tanto, la denotaremos por $\succsim_j = \succsim_T,\ \forall j \in T$)

$$S_2 \succsim_T S_1 \iff w_j(S_2) \geq w_j(S_1) \tag{3-13}$$

Con la función de preferencia $w_j$ dada por

$$w_j(S) = \begin{cases} 0, & si\ S \in h(j), \\ x_j^S, & otros, \end{cases} \tag{3-14}$$

Por lo que, las preferencias de las tareas se reflejan en la función $w_j$, esta función es diferente a la de los agentes, ya que las tareas son entidades independientes que actúan únicamente en su propio interés. Por ello, en base a (3-14), cada tarea prefiere a la coalición que le proporcione mayor beneficio $x_j^S$, a menos que la coalición ya haya sido visitada previamente y abandonada. En ese caso, la función de preferencia de las tareas asigna un valor de preferencia de 0 a cualquier coalición que la tarea ya haya visitado en el pasado (y que haya abandonado para unirse a otra coalición). Usando esta relación de preferencias, cada tarea puede evaluar sus preferencias sobre las posibles coaliciones que la tarea puede formar.

Es preciso destacar que según las relaciones de preferencia definidas en (3-12) y (3-14), cualquier coalición que se forme en el modelo propuesto debe ser estable ya que ningún agente o tarea tiene un incentivo para unirse a una coalición inestable. Por lo tanto, para cada coalición $S \subseteq N$, tenemos $|G_S|$ recolectores con $G_S \subseteq S \cap M$ , sabiendo de (3-6) la condición de estabilidad, y asumiendo que la capacidad de trasmisión de enlaces es la misma ($\mu_i = \mu$) y que todas las tareas son de la misma clase ($\lambda_i = \lambda$) llegamos a que cualquier coalición $S \subseteq N$ con $|S \cap M|$ agentes, debe tener al menos $|G_S|_{\,min}$ agentes recolectores como se indica

$$|G_S|_{\,min} = \frac{|S \cap T|\lambda}{\mu} \tag{3-15}$$

En consecuencia, para cualquier algoritmo propuesto para la formación de coaliciones, los límites del número de agentes recolectores en cualquier coalición deben satisfacer la ecuación anterior.

### 3.4.2 Algoritmo

Una vez modelado el problema de asignación de tareas como un juego hedónico de formación de coaliciones, el siguiente objetivo es idear un algoritmo para formar las coaliciones. El algoritmo que se usa en este trabajo [12] para la formación de coaliciones, permite a los jugadores tomar decisiones sobre las coaliciones a las que deciden unirse en cualquier momento. En este sentido, para formar coaliciones entre las tareas y los agentes, se usa la siguiente regla [12] para la formación de coaliciones:

**Deinición 3.4**. La **regla de conmutación** (o switch rule) establece que, dada una partición $\Pi = \{S_1, \dots, S_l\}$ del conjunto de jugadores (agentes y tareas) N, un jugador i decide dejar su coalición actual $S_\Pi(i) = S_\mathrm{m}$ por algunos $m \in \{1, \dots, l\}$ y unirse a otra coalición $S_\mathrm{k} \in \Pi \cup \{\emptyset\}$, $S_\mathrm{k} \neq S_\Pi(i)$, si y solo si $S_\mathrm{k} \cup \{i\} \succ_i S_\Pi(i)$. Por lo tanto, $\{S_\mathrm{m}, S_\mathrm{k}\} \rightarrow \{S_\mathrm{m} \setminus \{i\}, S_\mathrm{k} \cup \{i\}\}$.

Para cada partición Π, la regla de conmutación proporciona un mecanismo a través del cual cualquier tarea o agente, puede abandonar su coalición actual $S_\Pi(i)$, y unirse a otra coalición $S_\mathrm{k} \in \Pi$, dado que la nueva coalición $S_k \cup \{i\}$ es estrictamente preferida sobre $S_\Pi(i)$ a través de cualquier relación de preferencia que *i* esté usando (en particular usando las relaciones de preferencia definidas en (3-11) y (3-13)). Independientemente de las relaciones de preferencia seleccionadas, la regla de conmutación puede ser vista como una decisión egoísta tomada por el jugador, para moverse de su coalición actual a una nueva sin importar el efecto de este movimiento en los otros jugadores. Además, consideramos que, cada vez que un jugador decide cambiar de una coalición a otra, el jugador actualiza su conjunto histórico *h(i)*, almacenando la coalición antigua.

En consecuencia, se propone un algoritmo de formación de coaliciones compuesto de tres fases principales:

1) Descubrimiento de tareas
2) Formación de coaliciones hedónicas
3) Recopilación de datos

En la primera fase, el comando central recibe información sobre la existencia de tareas que requieren servicio e informa a los agentes de las ubicaciones y características de las tareas (por ejemplo, la tasa de llegada). Por lo tanto, los agentes empiezan por tener pleno conocimiento de la partición inicial $\Pi_{inicial}$. Una vez que los agentes son conscientes de las tareas, transmiten su propia presencia a las tareas. En consecuencia, los jugadores pueden interactuar entre sí, para llevar a cabo la formación de coaliciones.

La segunda fase del algoritmo es la fase de formación de la coalición hedónica. En esta fase, todos los jugadores (tareas y agentes) investigan la posibilidad de realizar una operación de conmutación. Para identificar posibles operaciones de conmutación, dado el conocimiento completo de la red, cada agente investiga su preferencia principal y decide realizar una operación de conmutación, si es posible a través de (3-11). Como se puede ver fácilmente en (3-8), para el modelo propuesto, no se formaría ninguna coalición compuesta sólo de tareas, ya que tal coalición siempre generaría una utilidad 0. Por lo tanto, las tareas sólo están interesadas en cambiar a coaliciones que contengan al menos un agente único.

Desde la perspectiva de las tareas, para determinar su operación de conmutación preferida, cada tarea solo necesita negociar con los agentes existentes con el fin de averiguar la cantidad de utilidad que puede obtener al unirse con este agente. Para cualquier agente, una operación de conmutación se realiza fácilmente ya que el agente puede dejar su coalición actual y unirse a la nueva coalición si cumple (3-11).

La convergencia del algoritmo propuesto durante la fase de formación de la coalición hedónica se garantiza de la siguiente manera.

Si denotamos $\Pi_{n_k}^k$ como la partición formada durante el tiempo $k$ cuando el jugador $i \in N$ decide actuar después de que las operaciones de conmutación $n_k$ hayan ocurrido previamente (el índice $n_k$ denota el número de operaciones de cambio realizadas por uno o más jugadores en el tiempo $k$). Dada cualquier partición inicial $\Pi_{inicial} = \Pi_0^1$, la fase de formación de la coalición hedónica del algoritmo propuesto consiste en una secuencia de operaciones de conmutación. Según la Definición 3.4, cada operación de conmutación transforma la partición actual $\Pi$ en otra partición $\Pi'$, por lo tanto, la formación de una coalición hedónica consiste en una secuencia de reglas de conmutación, produciendo, por ejemplo, las siguientes transformaciones

$$\Pi_0^1 = \Pi_0^2 \to \Pi_1^3 \to \cdots \to \Pi_{n_L}^L \ldots \to \cdots \to \Pi_{n_T}^T \tag{3-16}$$

Donde el operador $\to$ indica la ocurrencia de un cambio de operación. Por lo que, podemos decir que partiendo de cualquier partición de red inicial $\Pi_{inicial}$, la fase de formación de la coalición hedónica propuesta del algoritmo siempre converge a una partición de red final $\Pi_f$ compuesta por un número de coaliciones disociadas.

En términos de estabilidad, cualquier partición $\Pi_f$ que resulte de la fase de formación de la coalición hedónica del algoritmo propuesto es Nash estable y, por lo tanto, individualmente estable. Ya que, para cualquier partición $\Pi$, ningún jugador (agente o tarea) $i \in N$ tiene un incentivo para dejar su coalición actual, y actuar solo según la función de utilidad en (3-8).

El algoritmo expuesto es el siguiente:

Algoritmo: Formación de coaliciones hedónicas para la distribución de tareas entre robots inalámbricos

**Estado inicial:** La red está dividida en $\Pi_{inicial} = \{S_1, \dots, S_{k\}}$ **,** al inicio ($\Pi_{inicial} = N = M \cup T$) ninguna tarea está siendo atendida

*Fase 1- Descubrimiento de tareas*

a) El centro de mando es informado por uno o varios propietarios sobre la existencia y características de nuevas tareas.

b) El centro de mando central transmite la información sobre la partición de red inicial $\Pi_{inicial}$

*Fase 2- Formación de coaliciones hedónicas*

**repetir**

Para cada jugador $i \in N$, dada una partición actual $\Pi_{actual}$

a) El jugador *i* investiga posibles cambios usando las preferencias dadas, respectivamente, por (3-11) y (3-13) para los agentes y las tareas.

b) El jugador *i* realiza la operación de conmutación que maximiza su pago:

b.1) El jugador *i* actualiza su historia *h(i)* añadiendo $S_{\mathbf{\Pi}_{actual}}(i)$.

b.2) El jugador *i* abandona su coalición actual $S_{\mathbf{\Pi}_{actual}}(i)$.

b.3) El jugador *i* se une a la nueva coalición que maximiza su resultado.

**hasta** la convergencia a una partición final Nash estable $\Pi_f$

*Fase 3- Recopilación de datos*

a) La red se divide utilizando $\Pi_{final}$

b) Los agentes de cada coalición $S_k \in \Pi_{final}$ realizan continuamente las siguientes operaciones, es decir, actúan como un sistema de votación con una estrategia exhaustiva y tiempos de conmutación:

b.1) Visitar una primera tarea en sus respectivas coaliciones.

b.2) Los agentes recopilan los datos de la tarea que se está visitando.

b.3) Los agentes colectores transmiten los datos mediante enlaces inalámbricos al receptor central,

b.4) Una vez que la cola de la actual esté vacía, visite la siguiente tarea.

El orden en el que se visitan las tareas es determinado por la solución más cercana del vecino al problema del vendedor ambulante como en la Propiedad 3.3. Esta tercera fase se repite continuamente.

Algoritmo 3–1 Algoritmo general de formación de coaliciones hedónicas para la asignación de tareas en redes inalámbricas

# 4 IMPLEMENTACIÓN DEL MODELO PROPUESTO EN LA ETSI

## 4.1 Entorno de trabajo

El modelo de vigilancia propuesto es aplicable a una gran variedad de infraestructuras que tienen la necesidad de ser controladas y supervisadas. En este trabajo se va a aplicar a las distintas instalaciones de la Escuela Técnica Superior de Ingeniería de Sevilla (ETSI). La Escuela está compuesta por un edificio principal o Edificio plaza América que cuenta con 5 plantas (planta baja, entreplanta 1, planta primera, entreplanta 2 y planta ático) y un sótano, además de talleres y laboratorios. Cada una de las plantas del edificio principal será supervisada por un conjunto de robots y dispondrá de un receptor central que supervisará la correcta trasmisión de la información.

La razón de elegir este ámbito de aplicación es la gran cantidad de material, datos, equipamiento… de gran valor que hay en este tipo de entornos académicos. Por ejemplo, el centro de cálculo que se encuentra en la entreplanta 2 dispone de 11 salas con un total de 385 ordenadores, igualmente, en la biblioteca situada en la primera planta se encuentran una gran cantidad de portátiles, así como fotocopiadoras y escáneres. Aparte de todo el material que hay en cada departamento de la Escuela.

Por otro lado, en los laboratorios hay todo tipo de maquinaria industrial destinada a la práctica de los alumnos y al desarrollo de distintos proyectos. Todo este material junto con los datos de las investigaciones y estudios realizados, constituyen un entorno de gran valor el cual necesita ser protegido.

El patrullaje multirobot que se va a desarrollar se basa en un conjunto finito de robots que recorrerán repetidamente la planta que se quiera supervisar de la Etsi, comprobando que no haya intrusos en los puntos de interés establecidos, las cuales se han denominado tareas, transmitiendo la información de estas al receptor central. Por ello, se busca que el tiempo que una tarea está sin vigilancia sea el mínimo posible. Considerando que en la teoría de juegos los elementos que toman las decisiones en el juego son los jugadores; en este caso dichos jugadores serán los robots que decidirán que tareas van a visitar.

### 4.1.1 Planos de las plantas a vigilar

Como se ha comentado, es esencial que el tiempo que tardan los robots en pasar de una tarea a otra sea el menor posible para que el modelo sea eficiente. Por ello, hay que tener en cuenta la disposición de las distintas tareas en la planta y la distancia entre estas ya que los robots decidirán que coalición formar en base al tiempo que tardarían en completar la ruta de vigilancia. Los planos de las plantas del edificio principal se pueden observar en las siguientes imágenes.

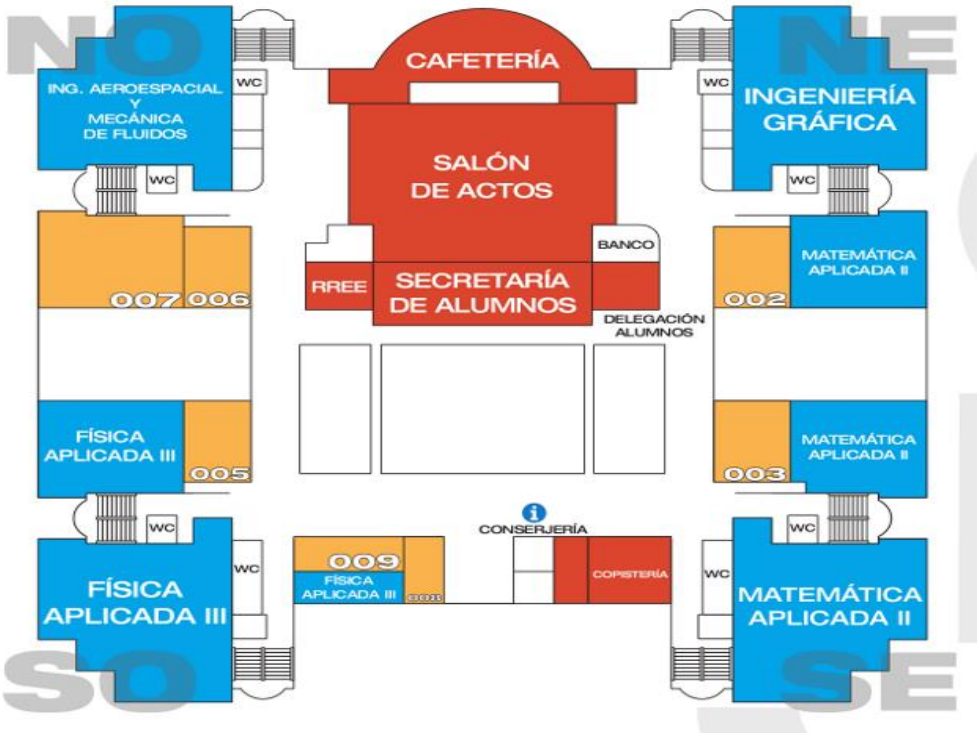


Figura 4.1. Plano planta baja.

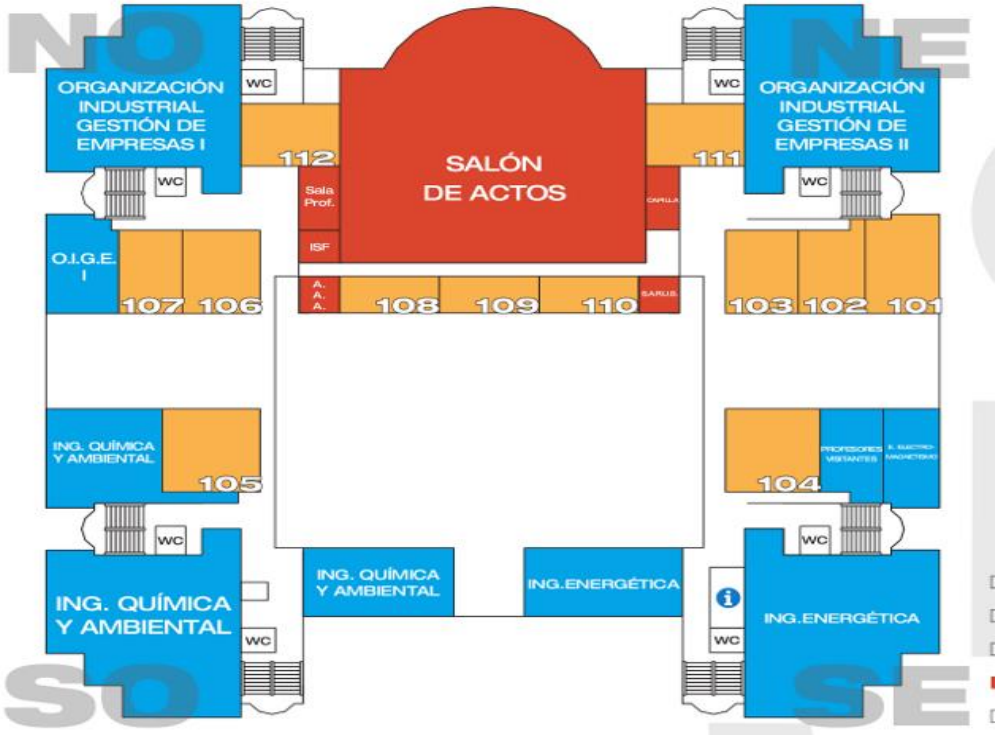


Figura 4.2. Plano entreplanta 1.

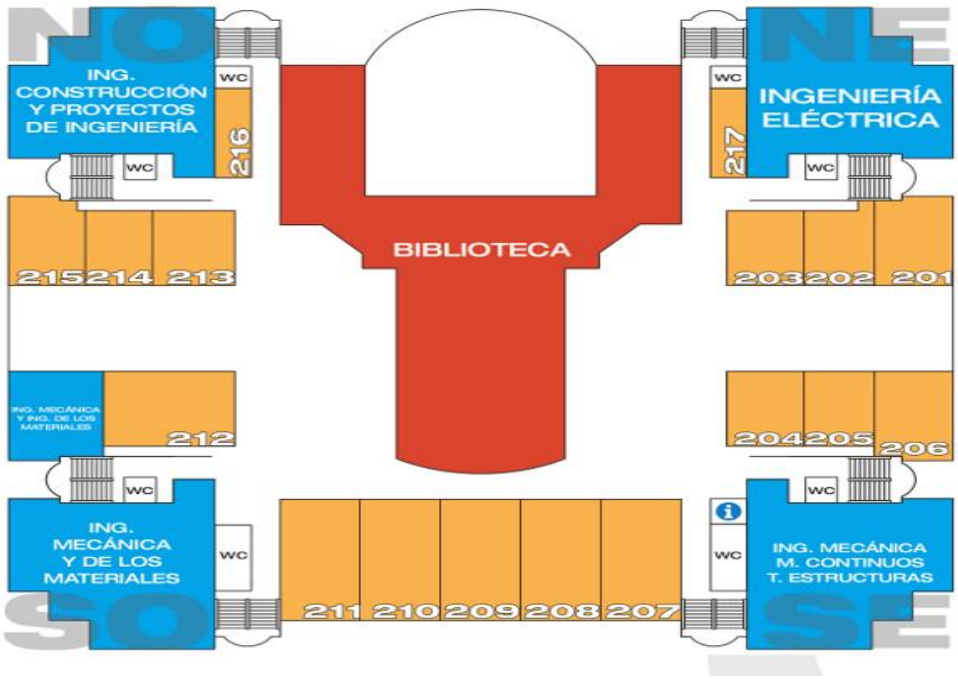


Figura 4.3. Plano planta primera.

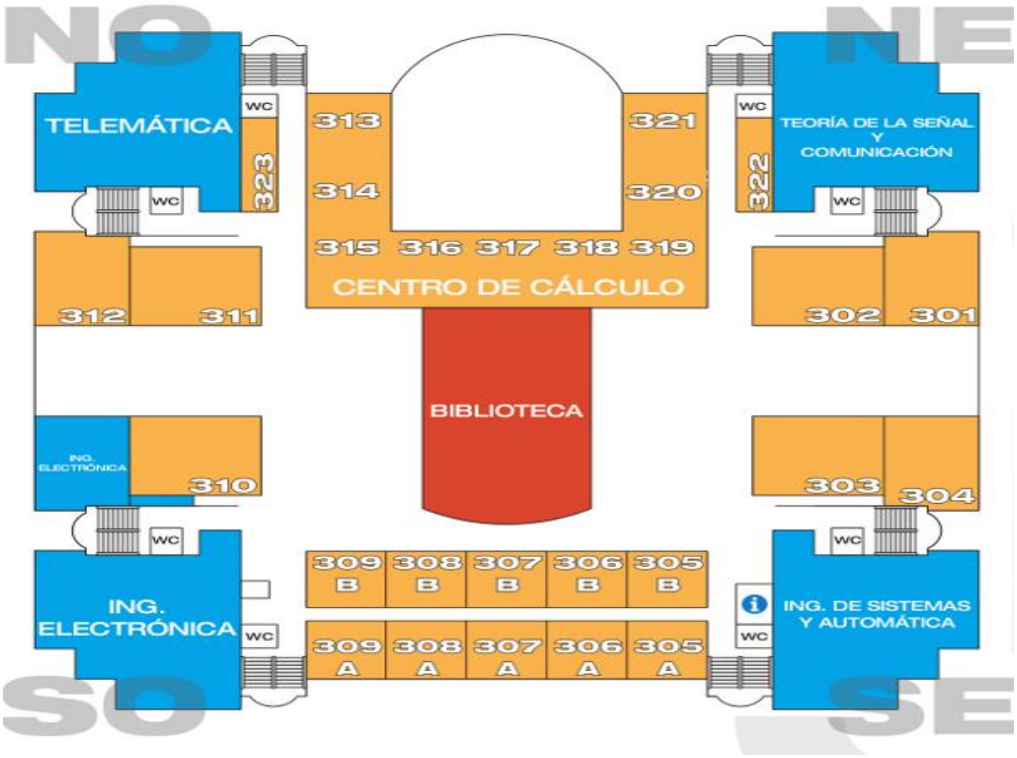


Figura 4.4. Plano entreplanta 2.

### 4.1.2 Posicionamiento de robots y tareas

Para la realización de las simulaciones se ha comenzado estudiando la distribución de las tareas y de los agentes en la planta baja de la ETSI. Se va a utilizar el modelo propuesto anteriormente para la formación de coaliciones entre robots y tareas por lo que el estudio de cada planta será independiente y contará con con un receptor central para cada una de las plantas. La aplicación del modelo en cada planta es similar, lo único que varía es la disposición de las tareas y de los agentes que en cada planta se establecerán en los lugares que susciten más interés.

La distribución de las tareas y los robots se puede ver en la siguiente imagen:

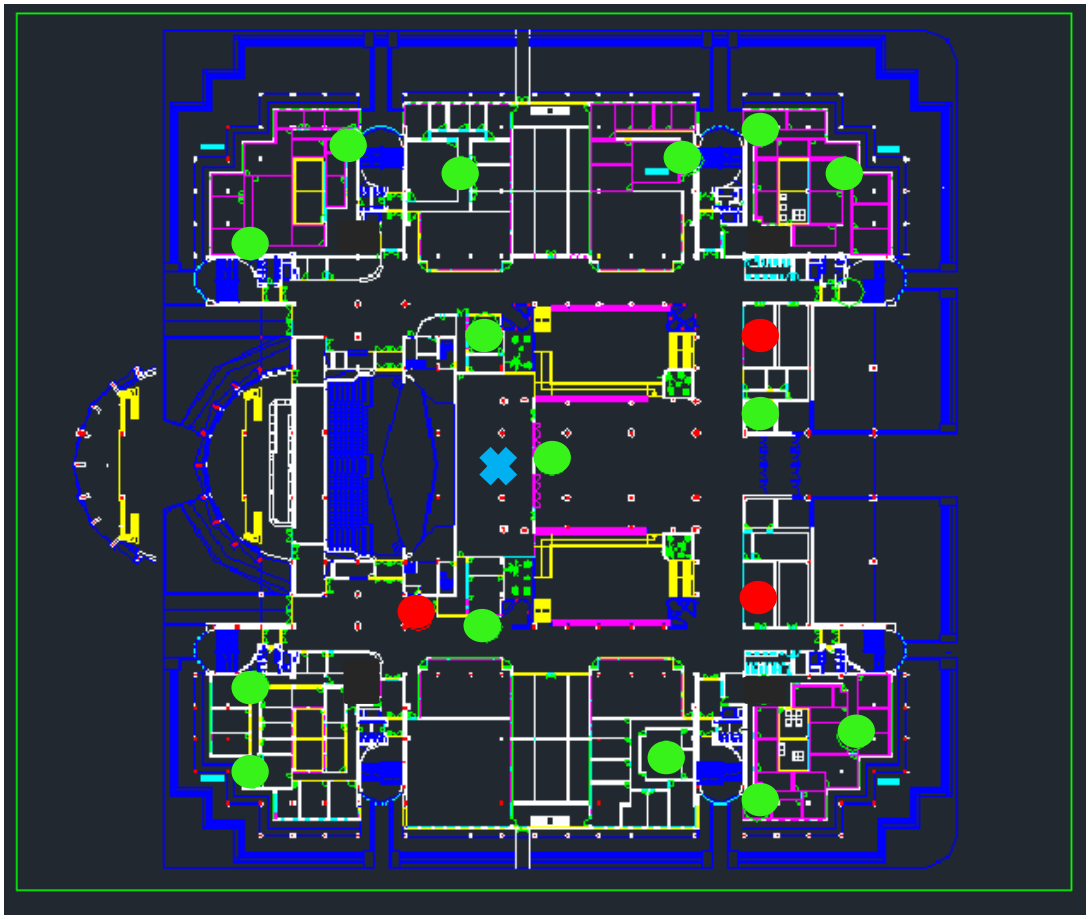

Figura 4.5. Distribución de tareas (puntos verdes) y robots (puntos rojos) en la planta baja de la Etsi.

Como se puede apreciar en la figura 4.5, se han establecido un total de 15 tareas repartidas por los puntos de interés de la planta baja, sobre todo en los distintos departamentos, secretaría, banco y relaciones internacionales. Y 3 robots cuya localización se ha decidido así ya que al estar más espaciados favorece la repartición de tareas, también son zonas donde una vez desarrollada su función, los robots pueden ser almacenados para su siguiente uso. La cruz azul representa donde estará situado el receptor central, el cual recibirá la información que le transmiten los robots al visitar una tarea.

En este trabajo se ha realizado la simulación en la planta baja como ejemplo ya que el número de robots y tareas en cada planta se ha considerado igual, por lo que la formación de las distintas coaliciones se realizará de igual modo en todas las platas, teniendo en cuenta que lo que varía es el posicionamiento de tareas y tobots en cada planta. Por ejemplo, la distribución de tareas y robots en la entreplanta 2, planta donde se encuentra el centro de cálculo, se puede ver en la siguiente imagen:

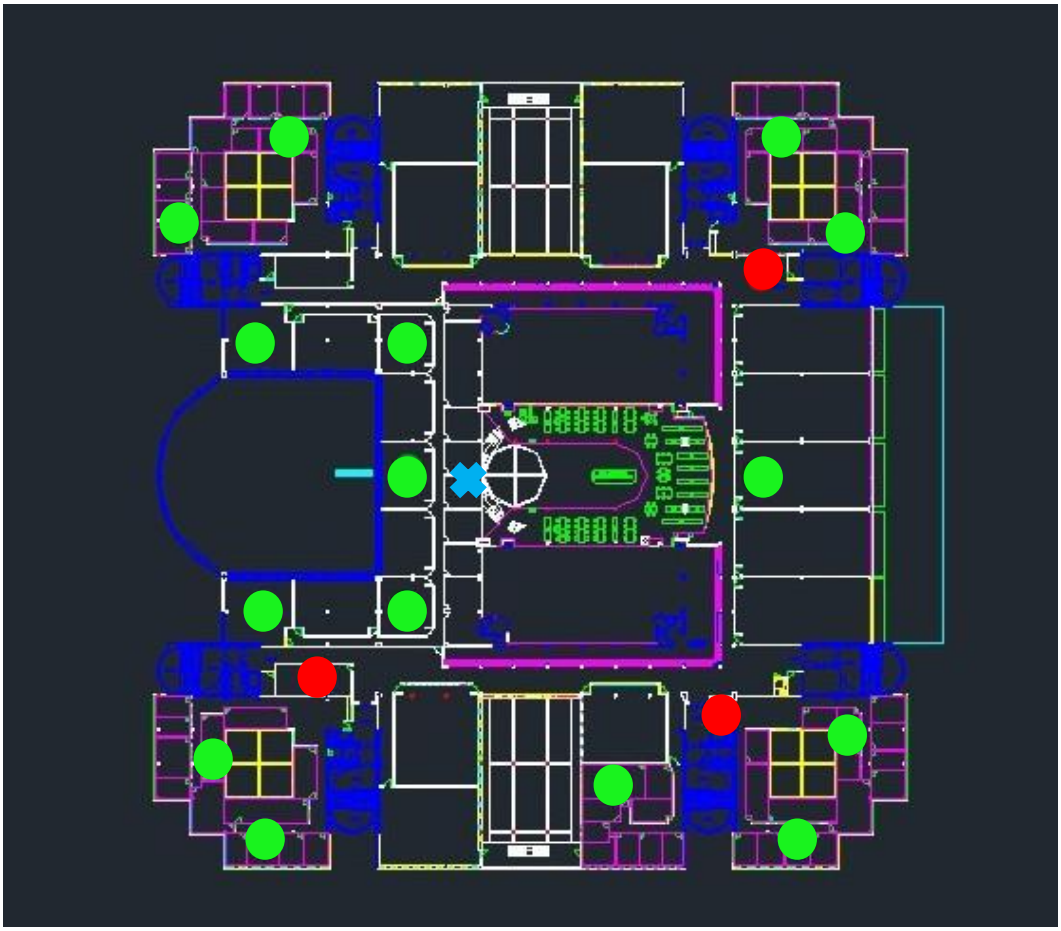

Figura 4.6. Distribución de tareas (puntos verdes) y robots (puntos rojos) en la entreplanta 2 de la Etsi.

Igualmente se dispone de 3 robots y 15 tareas, así como un servidor central, representado con una cruz azul. Los puntos de interés se encuentran en los departamentos y en el centro de cálculo debido al gran número de ordenadores contenidos en el mismo.

### 4.1.3 Parámetros

Para las simulaciones se va a considerar que todos los robots tienen las mismas características, velocidad constante de un 1 m/s, potencia de transmisión de 100 mW y capacidad total de transmisión de enlaces $\mu = 768\ kbps$ (se supone la misma para todos los agentes).

Por otro lado, se consideran dos tipos de tareas. Una primera clase que se puede asignar a servicios de voz con una tasa de llegada $\lambda = 32\ kbps$ y una segunda clase que se puede asignar a servicios de vídeo con una tasa de llegada $\lambda = 128\ kbps$. Las tareas que pertenecen a cada clase se generan con igual probabilidad en las simulaciones. Además, el parámetro de compensación de retardo de rendimiento $\beta$ está fijado a 0.7, para indicar servicios que son razonablemente tolerantes al retardo.

Los paquetes de datos alojados en la cola de cada tarea tienen un tamaño de 256 bits que es el tamaño standard de un paquete IP, la IP del paquete es la dirección que identifica al paquete e indica a donde se debe enviar. Se considera que la probabilidad de transmisión exitosa de un paquete de bits es del 60%.

Por último, se define el parámetro $\delta = 1$ que representa el coste por unidad de potencia que ofrece la red y permite al operador de red monitorear de alguna manera el comportamiento de los robots.

## 4.2 Aplicación del modelo matemático

### 4.2.1 Posibles combinaciones

Para poder aplicar el modelo matemático planteado en el capítulo anterior se necesita conocer todas las posibles coaliciones que se pueden formar con 3 robots y 15 tareas. Si llamamos R1, R2 y R3 a cada uno de los robots, R12 al conjunto formado por el robot 1 y 2, R13 al conjunto formado por el robot 1 y 3, R23 al conjunto formado por el robot 2 y 3 y finalmente R123 al conjunto formado por los tres robots, habrá que calcular todas las combinaciones posibles de cada una de las posibles uniones de los robots (o ellos solos) con las 15 tareas.

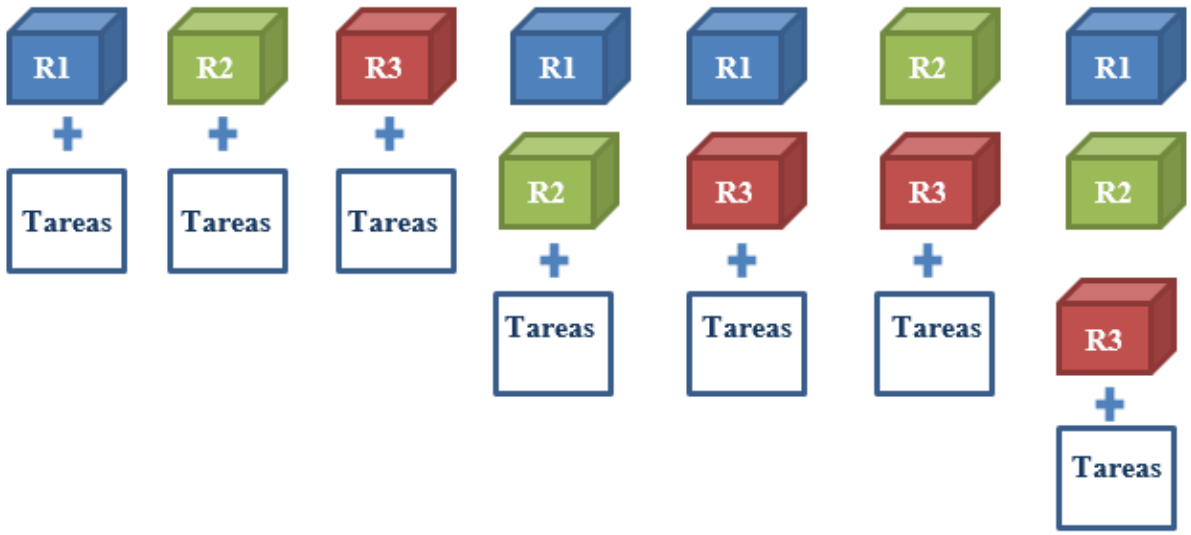


Figura 4.7. Combinaciones posibles entre los 3 robots y las tareas.

Cada robot y cada tarea se representa como un punto en el plano mostrado en la figura 4.5 por lo que cada punto tendrá coordenadas x e y. Las coordenadas reales donde se encuentran los robots en la planta baja de la ETSI son las siguientes:

| **Robots** | **Coordenadas** |
|---|---|
| Robot 1 | (205.13, 154.21) |
| Robot 2 | (205.13, 83.02) |
| Robot 3 | (11.83, 77.87) |

Tabla 4-1. Coordenadas de los robots situados en la planta baja.

Las coordenadas de las tareas son las siguientes:

| **Tareas** | **Coordenadas** |
|---|---|
| Tarea 1 | (206.7, 209.95) |
| Tarea 2 | (229.04, 199.95) |
| Tarea 3 | (185.91, 202.46) |
| Tarea 4 | (124.02, 199.18) |
| Tarea 5 | (93.54, 208.08) |
| Tarea 6 | (66.35, 178.57) |
| Tarea 7 | (130.58, 156.09) |
| Tarea 8 | (148.4, 120.49) |
| Tarea 9 | (205.6, 132.2) |
| Tarea 10 | (231.39, 44.62) |
| Tarea 11 | (205.6, 25.41) |
| Tarea 12 | (180.28, 39.496) |
| Tarea 13 | (131.05, 74.12) |
| Tarea 14 | (67.76, 58.67) |
| Tarea 15 | (66.82, 33.84) |

Tabla 4-2. Coordenadas de las tareas situadas en la planta baja.

Una vez se han adquirido los parámetros y coordenadas necesarias, se calculan las posibles coaliciones para cada robot con todas las tareas y los demás robots. Para ello se ha utilizado una función en Matlab (todos los programas realizados para el desarrollo del modelo han sido realizados en Matlab y el contenido de cada una de las funciones vendrá especificado en el Anexo A). La función utilizada es la siguiente:

`Nchoosek(M,K)`, donde M es una matriz de dos columnas (coordenada x e y) cuyas filas representan cada uno de los puntos donde se encuentran las diferentes tareas y los robots con los que estas formarán las coaliciones. Además, K representa el número en el que se van a agrupar los componentes de M, por ejemplo, si M está formado por el robot 1 y las 15 tareas y K=2, se formarán todas las coaliciones

posibles de dos puntos (el robot y una tarea) y si K=3 se formarán todas las combinaciones posibles de 3 puntos (el robot y dos tareas). Cabe destacar que solo nos quedaremos con las combinaciones que contentan algún robot, ya que sin el robot la tarea no puede trasmitir la información al receptor central.

Por ello, se calculan las posibles coaliciones para el robot 1, robot 2, robot 3, para las uniones de robot 1 y robot 2, robot 1 y robot 3, robot 2 y robot 3 y para la unión de los tres robots (R123) con todas las tareas como se muestra en la figura 4.7. Lo descrito anteriormente se realiza en las funciones realizadas en Matlab:

```
R1=Robot1(robot1x,robot1y);
R2=Robot2(robot2x,robot2y);
R3=Robot3(robot3x,robot3y);
R12=Robot12(robot1x,robot1y,robot2x,robot2y);
R13=Robot13(robot1x,robot1y,robot3x,robot3y);
R23=Robot23(robot2x,robot2y,robot3x,robot3y);
R123=Robot123(robot1x,robot1y,robot2x,robot2y,robot3x,robot3y);
```

A estas funciónes se les pasa como variables de entrada las coordenadas de los robots que participarán en las distintas combinaciones de robot/s con las tareas y se obtiene una matriz de celdas, cada celda de esta matriz contiene los distintos puntos que forman cada coalición. Se utilizan celdas ya que al trabajar con puntos (dos coordenadas) facilita el trabajo de guardar los distintos puntos de las tareas y robots que participan en cada una de las coaliciones.

Calculadas las 7 matrices de celdas, se crea una función que, dándole como parámetros de entrada el número en el que se quieren agrupar los componentes de la coalición (n), es decir, de cuanto en cuanto se agruparán las tareas y robots y una de las matrices de celdas calculadas anteriormente, devuelve todas las posibles coaliciones del robot/s con las distintas tareas. Esta función es la siguiente:

```
coal=Rob(n,R)
```

Por ejemplo, si se introduce n=4 y R=R2, devolverá una matriz con las posibles coaliciones de todas las tareas con el robot 2 agrupadas en grupos de 4 (3 tareas y un robot), si en vez de R2 introducieramos R12 la función devolverá una matriz con las posibles coaliciones de todas las tareas con los robots 1 y 2 unidos en grupos de 4 (2 tareas y 2 robots). Si se introduce R=R123, n debe ser mayor o igual que 6 ya que la formación será de tres robots con 3 tareas o más ya que no puede haber mas robots que tareas ya que sería ineficiente.

### 4.2.2 Beneficio de las coaliciones

Una vez calculadas todas las posibles agrupaciones de robots y tareas, se va a calcular el beneficio que obtendría cada robot por permanecer en cada posible coalición. Para ello, se hace uso del modelo matemático propuesto, empezando por calcular la capacidad total de transmisión de enlaces con los que los robots que intervengan en la coalición pueden atender a las tareas como se define en (3-1), además del factor de utilización de todas las tareas teniendo en cuenta que hay dos tipos de tareas (audio y video) que se generan con la misma probabilidad en las simulaciones.

Para cualquier coalición, una vez que el robot ha terminado de dar servicio a una tarea pasa a la siguiente con una velocidad constante, con lo que se produce un tiempo de conmutación. Como se ha asumido que los robots comienzan su trabajo al mismo tiempo, este tiempo de conmutación se refiere al tiempo necesario para que el agente se mueva de una tarea a la siguiente, aunque se producirán retardos que se deben analizar. Para ello, se hace uso de la ley de pseudo-conservación definida en (3-5).

Los agentes deben determinar el orden en que las tareas en cada coalición son visitadas minimizando el tiempo total de conmutación para una sola ronda de recolección de datos. Para esto, se utiliza el problema de los vendedores ambulantes, que consiste en que cada robot elige como próxima tarea a visitar la que está más cerca, minimizando así la ruta total. La función que basándose en el método de los vendedores ambulantes calcula la ruta óptima en cada coalición es la siguiente:

```
rte=RTE(coal)
```

Donde el parámetro de entrada que se le da a la función es la coalición que se quiere analizar (calculada con la función anterior) y se obtiene un vector (rte) con la ruta óptima entre los puntos, al ser el primer punto de la coalición un robot/s, este será siempre el punto incial desde el cual se calcula la ruta óptima.Una vez conocido el mejor orden en el que el robot tiene que visitar las tareas que se encuentran en su coalición se debe calcular las distancias entre cada uno de los puntos para así poder calcular el tiempo que tarda el agente es pasar de una tarea a la siguiente. Para calcular el vector de distacias entre los puntos de la coalición (coal) según el orden de visita obtenido en rte, se usa la siguiente función:

```
Vdis = VectorDistancias(coal,rte)
```

A dicha función se le da como parámetros de entrada la coalición a analizar y su ruta óptima, y se obtiene como resultado un vector con las distancias que hay entre cada punto de la coalición siguiendo la ruta óptima. Una vez conocemos las distancias entre cada punto y la velocidad a la que se mueven los robots, se puede calcular el tiempo que tarda el agente en realizar esa distancia. Por tanto, se calcula la suma ponderada de los tiempos de espera de los agentes en todas las tareas de la coalición, esto viene definido en de la ley de pseudo-conservación (3-5). Posteriormente, se calcula la función de valor de la coalición teniendo en cuenta, el rendimiento efectivo promedio, la probabilidad de transmisión exitosa de un paquete de bits y la suma ponderada de los tiempos de espera de los agentes. Cabe destacar que, si la coalición no es estable, como se definió en (3-6), su función de valor será nula y por tanto su beneficio también ya que esta condición es requisito indispensable para la estabilidad de cualquier sistema de votación. Finalmente, teniedo la función de valor de la coalición se realizará el reparto de beneficio entre los jugadores que hayan participado en la coalción mediante la regla de repartición justa enunciada en el capítulo anterior. Esto se hará en la siguiente función:

```
[v,B]=Beneficio(Vdis,m,T)
```

Donde los parámetros de entrada son el vector de distancias entre los puntos que forman la coalición que se está estudiando, el número de agentes m y el número de tareas T. Y se obtienen la función de valor v y el beneficion B, de cada jugador que componga la coalición. Para poder comparar cual es la mejor coalición a la que se puede unir cada robot habrá que calcular el beneficio de cada una de las posibles coaliciones entre las distintas uniones de los robots y las tareas y comprobar cual da el mayor beneficio.

Para calcular el beneficio de cada coalición se utiliza el siguiente bucle de Matlab, que engloba las funciones expuestas anteriormente:

```
for i=1:T+m:length(coal)
    rte = RTE(coal(i:i+T+m-1,:));
    Vdis = VectorDistancias(coal(i:i+T+m-1,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB(i)=B;
end
```

Donde según la function coal=Rob(n, R) que introdujamos calculará para cada una de las coaliciones que hay en la matriz coal la ruta óptima entre sus puntos, la distancia entre cada uno de los puntos y por último el beneficio que se obtiene de cada coalición. El beneficio de cada coalición se almacena en un vector (VectB).

### 4.2.3 Preferencias de los robots

Conocidos los beneficios que aporta cada coalición en todas las posibles agrupaciones entre robots y tareas, se analizan los beneficios que obtienen cada uno de los robots según las uniones con los demás robots que puedan formar. Para ello se utilizan las siguientes funciones:

```
M1=BeneficioR1(B1,B12,B13,B123)
M2=BeneficioR2(B2,B12,B23,B123)
M3=BeneficioR3(B3,B13,B23,B123)
```

Como el robot 1 participa en R1, R12, R13 y R123, se calcula su matriz de beneficios totales M1, pasandole como variables de entrada las matrices que contienen los beneficios de todas las combinaciones realizables donde participa el robot 1. Igualmente se calculan las matrices de benefios del robot 2 (M2) y del robot 3 (M3).

Una vez tenemos las matrices de beneficios de los tres robots se procede a calcular la coalición que maximiza el beneficio del robot. Para esto se usan las siguientes funciones:

```
[CoalicionMaxM1,fil,col]=PreferenciaR1(M1)
[CoalicionMaxM2,fil,col]=PreferenciaR2(M2)
[CoalicionMaxM3,fil,col]=PreferenciaR3(M3)
```

Donde se le pasa como variable de entrada la matriz de beneficios del robot a analizar y se obtiene la coalición con el mayor beneficio para el robot en cuestión. Además, se obtienen la fila y la columna dentro de la matriz M de beneficios que servirá para identificar los puntos de la coliación calculada y así saber que tareas y robots intervienen en esta coalición.

Por último, para la realización del Algoritmo 3–1 se tiene que tener en cuenta la historia del robot, ya que guarda en una variable *h* las coaliciones a las que ya a pertenecido e irá cambiando a la coalición que maximice su beneficio sin repetir en una coalición en la que ya habia participado, por lo que se dará un valor nulo a la coalición que se encuentre en la historia del robot como de se definió en (3-12). Esto se puede ver en la siguiente parte de código:

```
for i=1:10
[CoalicionMaxM1,fil,col]=PreferenciaR1(M1);
h(i)={CoalicionMaxM1};
M1(fil,col)=0;
[CoalicionMaxM2,fil,col]=PreferenciaR1(M1);
end
```

Este bucle representa como, una vez encontrada una primera coalición de máximo beneficio, esta se guarda en la variable *h*, que la almacena en una celda y pone su valor en la matriz de beneficios a 0 para así calcular la siguiente coalición que máximiza el beneficio del robot. Esto se aplica a los tres robots hasta que convergen a una partición estable donde todas las tareas sean atendidas de forma eficiente.

## 4.3 Resultados

Usando las funciones expuestas anteriormente llegamos a que, por ejemplo, la coalición de máximo beneficio para el robot 1 sería la formada por el mismo, con el robot 3 y las tareas 1,2,3 y 13 como se ve en la siguiente imagen, obteniendo un beneficio de **10.5575**.

```
CoalicionMaxM1 =

  205.1300  154.2100
  111.8300   77.8700
  206.0700  209.9500
  229.0400  199.1800
  185.9100  202.4600
  131.0500   74.1200
```

Figura 4.8. Coalición de máximo beneficio para el robot 1.

Donde el robot 1 se encargaría de las tareas 1, 2, 3 y el robot 3 de la 13. Aunque para llegar a una partición estable y que todas las tareas sean visitadas por un robot los distintos robots, siguiendo el algoritmo planteado, tienen que comparar las coaliciones que cumplen que todas las tareas sean visitadas en el menor tiempo posible, es decir que obtengan el mayor beneficio posible y que todas las tareas sean atendidas.

Según esto se llega a que una posible partición estable de los robots y tareas sería la compuesta por una primera coalición formada por el robot 1 y las tareas 2, 1, 3, 4, 5 y 6.

```
CoalicionEstable =

  205.1300  154.2100
  229.0400  199.9500
  206.7000  209.9500
  185.9100  202.4600
  124.0200  199.1800
   93.5400  208.0800
   66.3500  178.5700
```

Figura 4.9. Coalición estable para el robot 1.

De la cual el robot 1 obtendría un beneficio de **5.4142**. En la siguiente imagen se puede ver la ruta que realizaría el robot 1 visitando todas las tareas de su coalición.

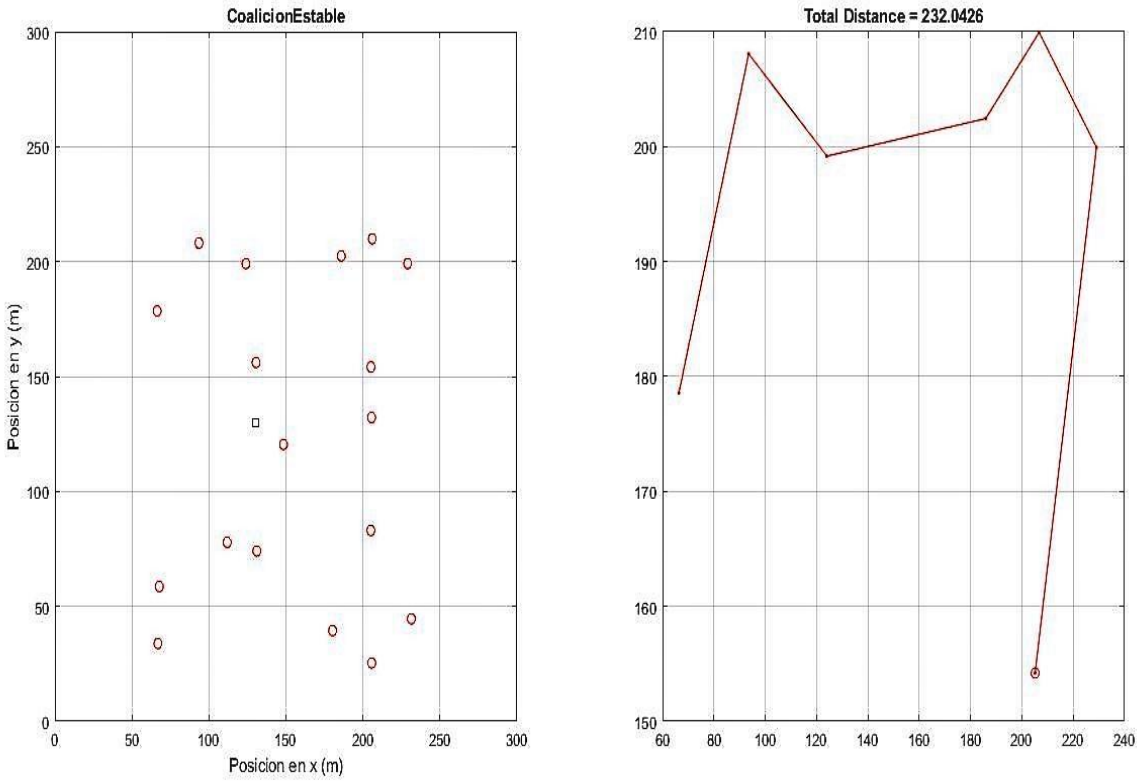


Figura 4.10. Gráfica de la ruta que realiza el robot 1.

En la figura 4.10. se puede observar a la izquierda la posición de todas las tareas, representadas como círculos, y un cuadrado negro que representa la posición del servidor central. Por otro lado, en la parte derecha de la figura se puede ver la ruta que seguirá el robot 1, considerando esta ruta óptima en términos de espacio recorrido en menor tiempo.

Y una segunda coalición formada por el robot 2 y 3 y las tareas 9, 8, 7, 13, 14, 15, 12, 11, 10, 7. Donde el robot 2 se encargará de las tareas 9, 8, 7 y 13 y el robot 3 se encargará de visitar las tareas 14, 15, 12, 11 y 10. De la cual cada robot obtendrá un beneficio de **6.4401**.

```
CoalicionEstable2 =

  205.1300   83.0200
  111.8300   77.8700
  205.6000  132.2000
  148.4000  120.4900
  130.5800  156.0900
  131.0500   74.1200
   67.7600   58.6700
   66.8200   33.8400
  180.2800   39.4960
  205.6000   25.4100
  231.3900   44.6200
  130.5800  156.0900
```

Figura 4.11. Coalición estable para el robot 2 y robot 3.

En la siguiente imagen se puede ver la ruta que realizaría el robot 2 visitando las tareas de su coalición a las que tiene que dar servicio.

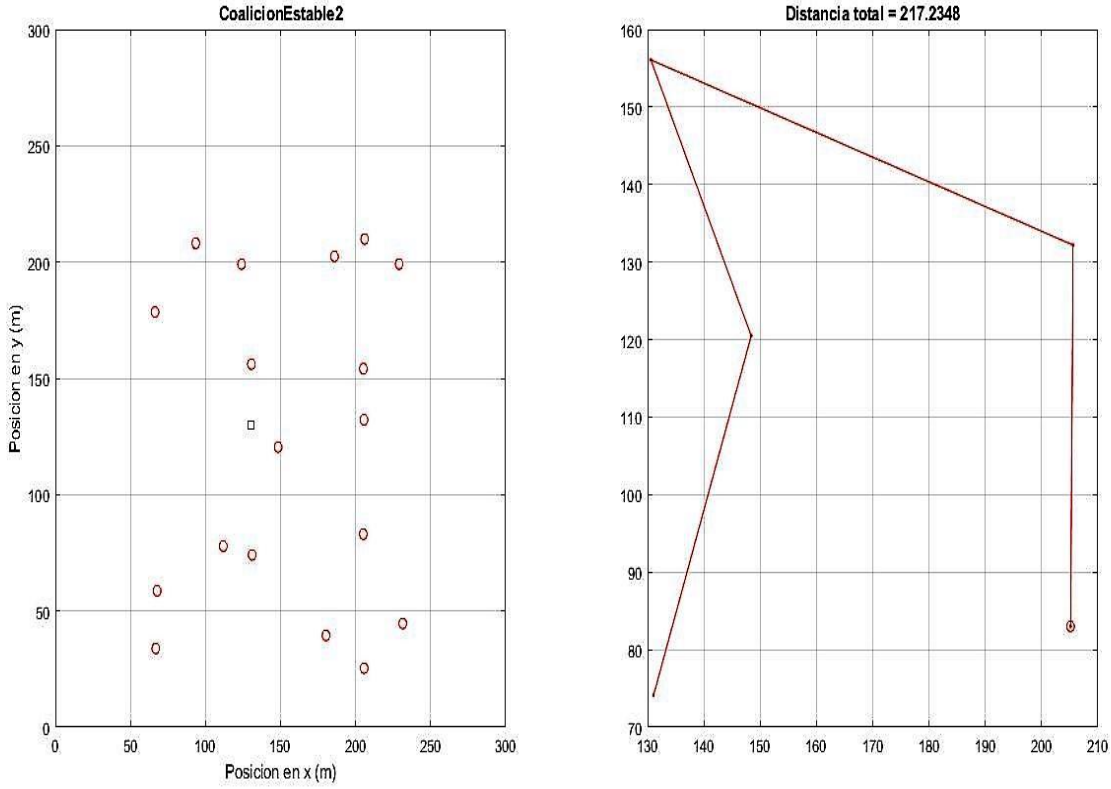


Figura 4.12. Gráfica de la ruta que realiza el robot 2.

Igualmente, se aprecia en la parte derecha, la ruta que realiza el robot 2 partiendo desde su posición inicial (marcada con un punto rojo) y pasando por las tareas asignadas. Y finalmente el robot 3 se encarga de las tareas restantes en la coalición, como se ve en la siguiente imagen, donde también se puede comprobar la distancia total que realiza cada robot tras recorrer todas las tareas que le han sido asignadas. Se comprueba que la distancia total que recorre cada agente es similar, por lo que ningún robot tendrá que esperar mucho tiempo a que los otros robots terminen su trabajo, haciendo que el modelo sea estable y eficiente.

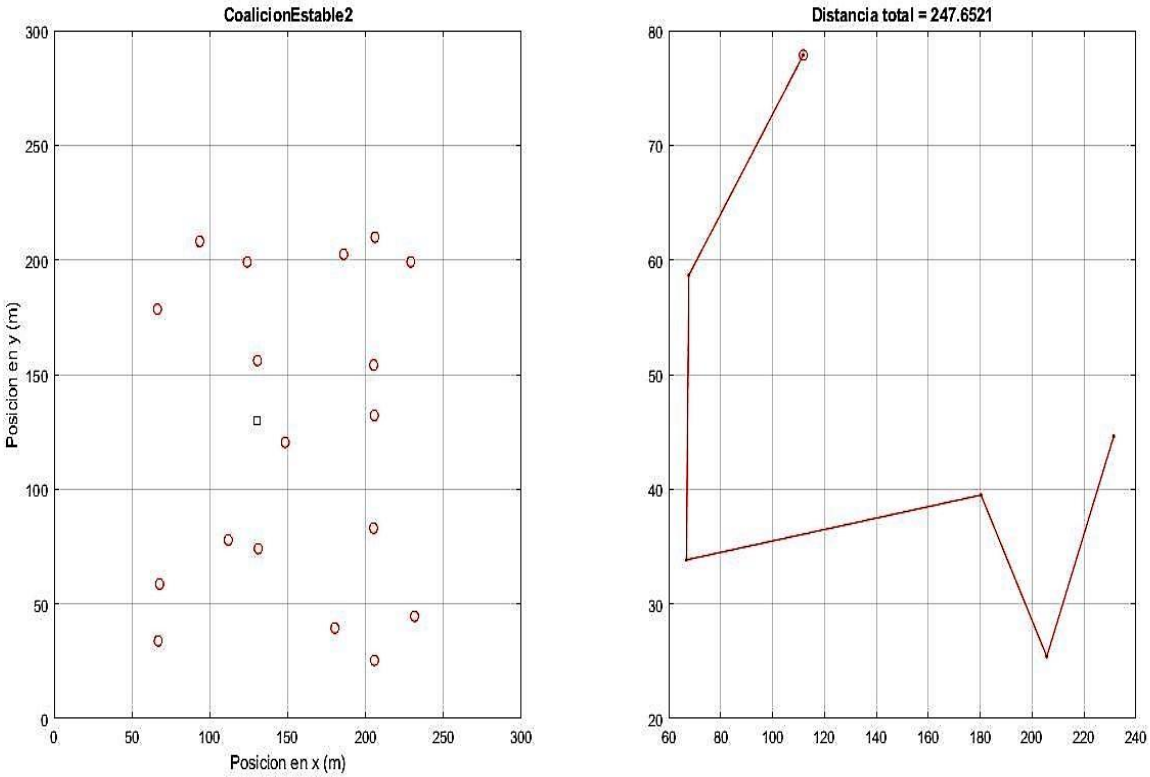


Figura 4.13. Gráfica de la ruta que realiza el robot 3.

Este sería un buen resultado de partición estable ya que los beneficios son bastante aceptables y cumple que todas las tareas serán atendidas, aunque hay más posibles particiones que pueden servir como coaliciones estables, por lo que los robots siempre compararán el beneficio obtenido en la coalición en la que se encuentran para cambiar de coalición y mejorar su beneficio.

Cabe destacar que para algunas coaliciones se ha obtenido beneficio nulo, como por ejemplo en la coalición donde el robot 1 se encarga de 12 tareas y los robots 2 y 3 de 3, esto incumple la condición de estabilidad vista en (3-6), ya que un solo robot se encarga de la mayoria de las tareas y llegaría un momento que los otros robots no tendrían tareas que atender y el robot 1 tardaría mucho tiempo en ocuparse de las tareas de su coalición lo cual no es óptimo en cuanto a tiempo y beneficio.

# 5 CONCLUSIONES Y LINEAS FUTURAS

## 5.1 Conclusiones

El patrullaje multirobot en grandes espacios es un problema que necesita soluciones eficientes y que optimicen el tiempo en el que se realiza la vigilancia. En este trabajo se presenta un modelo para la asignación de tareas entre agentes independientes que se comunican entre ellos y con un receptor central mediante una red inalámbrica. Cada tarea representa una cola de paquetes que requieren ser trasmitido por los robots al receptor central. El problema de la asignación de tareas se modela como un juego hedónico de formación de coaliciones entre los agentes y las tareas que interactúan para formar coaliciones autónomas. Los agentes que compongan cada coalición recolectarán continuamente los paquetes de las tareas y los transmitirán al receptor central.

Para formar las coaliciones, se introduce un algoritmo que permite que los robots se unan o abandonen las coaliciones en base a sus preferencias, las cuales reflejan el equilibrio entre el rendimiento efectivo y el retraso logrado por la coalición. Los resultados de la simulación muestran cómo el algoritmo propuesto permite que los agentes y las tareas se autoorganicen de forma independiente, llegando a alcanzar coaliciones estables entre los robots y las tareas. El algoritmo proporciona características dinámicas en cuanto a que, si el entorno cambia, el comportamiento de los robots puede adaptarse a los distintos cambios. Además, cada planta se estudia independientemente por lo que no todo el edificio principal depende de un servidor central si no que se distribuyen por las distintas plantas.

Por otro lado, se han encontrado limitaciones en el procesado de datos, ya que la cantidad de combinaciones distintas que se pueden formar con los distintos robots y tareas es muy grande y a la hora de realizar las distintas operaciones con cada una de las miles de combinaciones que se pueden formar el tiempo de espera a que el ordenador termine de realizar los cálculos es bastante elevado, pasando de cálculos menos pesados que se realizan en minutos a los más pesados que pueden tardar un dia en terminar.

En definitiva, se ha conseguido un modelo que proporciona escalabilidad, adaptabilidad, robustez y el potencial para acomodar los diferentes intereses de los robots obteniendo resultados aceptables. Sin embargo, se necesitaría más investigación para analizar la calidad de una partición estable Nash obtenida por el marco propuesto.

## 5.2 Lineas futuras

En cuanto al trabajo que queda por hacer, lo primero sería intentar reducir el tiempo de procesado de datos para una mayor comodidad a la hora de realizar las distintas simulaciones. Para la realización del modelo propuesto cada dato que se procesa pasa por muchas funciones, por lo que habría que intentar depurar cada una de ellas para optimizar el programa final que nos da el beneficio que cada robot toma como criterio de preferencia para unirse a una coalición o a otra.

También sería interesante poder introducir robots en el área de vigilancia y que los demás robots los detectarán y los tuvieran en cuenta sin necesidad de parar toda la actividad de vigilancia que ya se está desarrollando. Esto sería interesante en ocasiones que se considere que la vigilancia en esos momentos tiene que ser más precisa y aumentando el número de robots, el tiempo de visita de una tarea a la siguiente disminuye creando un modelo más eficiente.

Por otro lado, en este trabajo se ha utilizado en las simulaciones la regla de igualitaria para repartir los beneficios obtenidos en una coalición entre los distintos jugadores que conforman la coalición. Este método es sencillo, pero los resultados podrían variar mucho si se utilizaran otros métodos más elaborados como el valor de Shapley, por lo que en futuros trabajos se podría hacer una comparación entre estos dos métodos de pago comprobando cual de los dos da resultados más fiables.

# REFERENCIAS


[1] Joaquín Pérez, José Luis Jimeno, Emilio Cerdá (2004), *Teoría de juegos*. Ediciones Pearson.

[2] Morgenster, O. y von Neumann, J (1944). *Theory of games and economic behaviour*. Princeton University Press, Princeton, New Jersey.

[3] Owen, G. (1995). *Game Theory*. Academic Press.

[4] Shapley, Lloyd S. (1953). *"A Value for n-person Games*". In Kuhn, H. W.; Tucker, A. W. Contributions to the Theory of Games. Annals of Mathematical Studies. Princeton University Press.

[5] Haris Aziz and Rahul Savani, "Hedonic Games". Chapter 15 in: *Brandt, Felix; Conitzer, Vincent; Endriss, Ulle; Lang, Jérôme; Procaccia, Ariel D. (2016).* Handbook of Computational Social Choice. *Cambridge University Press.*

[6] Bogomolnaia, Anna; Jackson, Matthew O. (Feb 2002). "The Stability of Hedonic Coalition Structures". *Games and Economic Behavior*.

[7] H. Takagi, Analysis of Polling Systems. Cambridge, MA, USA: The MIT Press, Apr. 1986.

[8] H. Levy and M. Sidi, "Polling systems: applications, modeling, and optimization," IEEE Trans. Commun., vol. 10, pp. 1750–1760, Oct. 1990.

[9] D. Applegate, R. M. Bixby, V. Chvatal, and W. J. Cook, The traveling salesman problem: a computational study. Princeton, NJ, USA: Princeton University Press, 2006.

[10] V. Vukadinovic and G. Karlsson, "Video streaming in 3.5G: On throughput-delay performance of proportional fair scheduling," in Proc. Int. Symp. on Modeling, Analysis and Simulation of Comp. and Telecom. Systems, California, USA, Sep. 2006.

[11] R. B. Myerson, Game Theory, Analysis of Conflict. Cambridge, MA, USA: Harvard University Press, Sep. 1991.

[12] Walid Saad, Zhu Han, Tamer Basar, Mérouane Debbah, and Are Hjørungnes. Hedonic *Coalition Formation for Distributed Task Allocation among Wireless Agents.* IEEE Transactions on Mobile Computing, 2011.

[13] Haluk Bayram and H. Isil Bozma. *Coalition Formation Games for Dynamic Multirobot Tasks*. Bogazici University, Department of Electrical and Electronics Engineering, Intelligent Systems Laboratory.

[14] Khin Haymar Saw Hla, YoungSik Choi, and Jong Sou Park. *The Multi Agent System Solutions for Wireless Sensor Network Applications*. Department of Computer Engineering, Korea Aerospace University, Korea.

[15] Inmo Jang, Hyo-Sang Shin, and Antonios Tsourdos. *Anonymous Hedonic Game for Task Allocation in a Large-Scale Multiple Agent Systems.*

# ANEXO A: CÓDIGO DE LAS FUNCIONES

Función que calcula todas las posibles combinaciones de tareas con el robot 1. En las otras posibles uniones entre robots y tareas se procederá igual.

```
function R1=Robot1(robotx1,roboty1)

%Coordenadas de los puntos donde se encuentran las tareas.
%El primer punto de la matriz D es el Robot1 y todos los demás el conjunto
%de tareas
 x=[robot1x, 206.07, 229.04, 185.91, 124.02, 93.54, 66.35, 130.58, 148.40,
205.6, 231.39, 205.6, 180.28, 131.05, 67.76,66.82];
 y=[robot1y, 209.95, 199.18, 202.46, 199.18, 208.08, 178.57,156.09, 120.49,
132.2, 44.62, 25.41, 39.46, 74.12, 58.67,33.84];
 D=[x;y]';
 %Creo una matriz de celdas con el robot y todas las tareas
for i=1:length(x)
    for j=1:1
    G(i,j)={D(i,:)};
    end
end

%Usando la función nchoosek obtengo todas las posibles combinaciones entre
%robots y tareas, me quedo solo con las combiaciones que contienen uno o
%varios robots
  P2=nchoosek(G,2); A2=P2(1:((size(P2,1)*size(P2,2))/length(x)),:);
  P3=nchoosek(G,3); A3=P3(1:((size(P3,1)*size(P3,2))/length(x)),:);
  P4=nchoosek(G,4); A4=P4(1:((size(P4,1)*size(P4,2))/length(x)),:);
  P5=nchoosek(G,5); A5=P5(1:((size(P5,1)*size(P5,2))/length(x)),:);
  P6=nchoosek(G,6); A6=P6(1:((size(P6,1)*size(P6,2))/length(x)),:);
  P7=nchoosek(G,7); A7=P7(1:((size(P7,1)*size(P7,2))/length(x)),:);
  P8=nchoosek(G,8); A8=P8(1:((size(P8,1)*size(P8,2))/length(x)),:);
  P9=nchoosek(G,9); A9=P9(1:((size(P9,1)*size(P9,2))/length(x)),:);
  P10=nchoosek(G,10); A10=P10(1:((size(P10,1)*size(P10,2))/length(x)),:);
  P11=nchoosek(G,11); A11=P11(1:((size(P11,1)*size(P11,2))/length(x)),:);
  P12=nchoosek(G,12); A12=P12(1:((size(P12,1)*size(P12,2))/length(x)),:);
  P13=nchoosek(G,13); A13=P13(1:((size(P13,1)*size(P13,2))/length(x)),:);
  P14=nchoosek(G,14); A14=P14(1:((size(P14,1)*size(P14,2))/length(x)),:);
  P15=nchoosek(G,15); A15=P15(1:((size(P15,1)*size(P15,2))/length(x)),:);
  P16=nchoosek(G,16); A16=P16(1:((size(P16,1)*size(P16,2))/length(x)),:);

%Introduzco cada posible coalición en una celda
for i=1:size(A2,1)

          C2(i) = {A2(i,:)};
end
for i=1:size(A3,1)

          C3(i) = {A3(i,:)};
end
for i=1:size(A4,1)

        C4(i) = {A4(i,:)};
```

```
end


for i=1:size(A5,1)

          C5(i) = {A5(i,:)};
end
for i=1:size(A6,1)

          C6(i) = {A6(i,:)};
end
for i=1:size(A7,1)

          C7(i) = {A7(i,:)};
end
for i=1:size(A8,1)

          C8(i) = {A8(i,:)};
end
for i=1:size(A9,1)

          C9(i) = {A9(i,:)};
end
for i=1:size(A10,1)

          C10(i) = {A10(i,:)};
end
for i=1:size(A11,1)

          C11(i) = {A11(i,:)};
end
for i=1:size(A12,1)

          C12(i) = {A12(i,:)};
end
for i=1:size(A13,1)

          C13(i) = {A13(i,:)};
end

for i=1:size(A14,1)

          C14(i) = {A14(i,:)};
end
for i=1:size(A15,1)

          C15(i) = {A15(i,:)};
end
for i=1:size(A16,1)

          C16(i) = {A16(i,:)};
end

R1={C2 C3 C4 C5 C6 C7 C8 C9 C10 C11 C12 C13 C14 C15 C16};

 end
```

Función que devuelve una matriz con las posibles coaliciones del robot o unión de robots requerida y agrupados según el número que se introduja como parámetro de entrada.

```
function coal=Rob(n,R)
coal=[];

            for k=1:size(R{1,n},1)
                for z=1:size(R{1,n},2)
                    for p=1:size(R{1,n}{k,z},1)
                        for t=1:size(R{1,n}{k,z},2)

                            coal=[coal;R{1,n}{k,z}{p,t}];

                        end
                    end
                end
            end
end
```

Función que calcula la mejor ruta para visitar un grupo de puntos partiendo del punto inicial (coordenada del robot).

```
function rte=RTE(coal)

M=coal;
n=length(M);
xy = M;
popSize = 60;
numIter = 1e4;

 showProg = 1;
 showResult = 1;
 a = meshgrid(1:n);
 dmat = reshape(sqrt(sum((xy(a,:)-xy(a',:)).^2,2)),n,n);
 [optRoute,minDist] = tspofs_ga(xy,dmat,popSize,numIter,showProg,showResult);

 [N,dims] = size(xy);

 rte = [1 optRoute];
end
```

```
function varargout = tspofs_ga(xy,dmat,popSize,numIter,showProg,showResult)

% Verify Inputs
[N,dims] = size(xy);
[nr,nc] = size(dmat);
if N ~= nr || N ~= nc
    error('Invalid XY or DMAT inputs!')
end
n = N - 1; % Separate Start City

% Sanity Checks
popSize = 4*ceil(popSize/4);
numIter = max(1,round(real(numIter(1))));
showProg = logical(showProg(1));
showResult = logical(showResult(1));

% Initialize the Population
pop = zeros(popSize,n);
pop(1,:) = (1:n) + 1;
for k = 2:popSize
    pop(k,:) = randperm(n) + 1;
end

% Run the GA
globalMin = Inf;
totalDist = zeros(1,popSize);
distHistory = zeros(1,numIter);
tmpPop = zeros(4,n);
newPop = zeros(popSize,n);

for iter = 1:numIter
    % Evaluate Each Population Member (Calculate Total Distance)
    for p = 1:popSize
        d = dmat(1,pop(p,1)); % Add Start Distance
        for k = 2:n
            d = d + dmat(pop(p,k-1),pop(p,k));

        end
        totalDist(p) = d;

    end

    % Find the Best Route in the Population
    [minDist,index] = min(totalDist);
    distHistory(iter) = minDist;
    if minDist < globalMin
        globalMin = minDist;
        optRoute = pop(index,:);
    end
end

if nargout
    varargout{1} = optRoute;
    varargout{2} = minDist;
end
end
```

Función que calcula la distancia entre los puntos de la coalición a estudiar siguiendo la mejor ruta calculada en la función anterior.

```
function Vdis = VectorDistancias(coal,rte)
D=coal;

for i=1:size(D,1)
    for j=1:length(rte)-1
      Dis(i,j)=norm(D(rte(j+1),:)-D(rte(j),:));
    end
end
Vdis=Dis(1,:);
end
```

Función que calcula el beneficio que obtendría cada jugador por pertener a la coalición qu se está estudiando.

```
function [v,B]=Beneficio(Vdis,m,T)
%Consideramos solo el caso de mayor numero de tareas que agentes
if (m>T) || (T==0)
    B=0;
    v=0;
else
S=m+T;
vel=1;
FU=zeros(1,T);
FUtot=0;
landa1=32;
landa2=128;
CT=768*m;

landa=[landa1,landa1,landa2,landa2,landa1,landa1,landa1,landa2,landa1,landa2,
landa2,landa2,landa1,landa1,landa1,landa1,landa1,landa1,landa1,landa2,landa2,
landa2,landa2,landa2,landa1,landa1,landa1,landa1];
landaT=0;
for i=1:T
    FU(i)=landa(i)/CT;
    FUtot=FUtot+FU(i);
    landaT=landaT+landa(i);
end
FUtot;
%%
%2) Suma ponderada de los tiempos promedios de espera de los agentes en
%todas las tareas de la coalicion
Sum1=0;
Sum2=0;
Sum3=0;
for i=1:T
    FU(i)=landa(i)/CT;
    Sum1=Sum1+(FU(i)/CT);
end
for i=1:T
    FU(i)=landa(i)/CT;
    Sum2=Sum2+FU(i)^2;
end
```

```
%Con el algoritmo de los vecinos mas cercanos calculo la ruta y distancia
%entre cada una de las tareas, calculo tiempo en realizar esa distancia
 for i=1:S-1
     omega(i)=Vdis(i);
     omega=Sum3+omega(i);
 end


SumPon=[FUtot*(Sum1/(2*(1-FUtot)))]+[FUtot*((omega^2)/2)]+[(omega/(2*(1-
FUtot)))*((FUtot^2)-Sum2)];

%%
%3) Rendimiento efectivo promedio (Ls)
%Pr Probabilidad de trasmision exitosa de un paquete de bits
Ls=0;
Pr=0.6;
for i=1:T
    Ls=Ls+(landa(i)*Pr);
end
%%
%4) Función de valor de la coalici0n (v)
%El factor de precio se ajusta a = 1
FP=1;
beta=0.7; %Indica servicios que son razonablemente tolerantes al retardo
if (FUtot<1) && (S>1) %FUtot<1 -> condicion de estabilidad
    v=FP*(Ls/((SumPon)^(1-beta)));
else
    v=0;
end
%%
%5) Beneficio (x) de cada jugador por el metodo de la reparticion justa
B=v/S;
end
end
```

Función que cálcula un vector con todos los beneficios para cada posible combinación de robots, según se introduja R1, R2, R12, R13, R23, R123.

```
%Coordenadas Robots
robot1x=205.13;
robot1y=154.21;
robot2x=205.13;
robot2y=83.02;
robot3x=111.83;
robot3y=77.87;
%Robot 1
 R1=Robot1(robot1x,robot1y);
 m=1;
%%
 T=1;
coal=Rob(T,R1);
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB1(i)=B;

end
 %%
T=2;
coal=Rob(2,R1);
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB2(i)=B;
end
 %%
 T=3;
 coal=Rob(T,R1);
for i=1:T+1:length(coal)
     rte = RTE(coal(1:1+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB3(i)=B;
end
 %%
T=4;
coal=Rob(T,R1);
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB4(i)=B;
end
 %%
T=5;
coal=Rob(T,R1);
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB5(i)=B;
end
 %%
T=6;
coal=Rob(T,R1);
```

```
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB6(i)=B;
end
%%
T=7;
coal=Rob(T,R1);
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB7(i)=B;
end
%%
T=8;
coal=Rob(T,R1);
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB8(i)=B;
end
%%
T=9;
coal=Rob(T,R1);
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB9(i)=B;
end
 %%
T=10;
coal=Rob(T,R1);
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB10(i)=B;
end
%%
T=11;
coal=Rob(T,R1);
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB11(i)=B;
end
 %%
T=12;
coal=Rob(T,R1);
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB12(i)=B;
end
 %%
T=13;
coal=Rob(T,R1);
for i=1:T+1:length(coal)
```

```
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB13(i)=B;
end
 %%
T=14;
coal=Rob(T,R1);
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB14(i)=B;
end
 %%
T=15;
coal=Rob(T,R1);
for i=1:T+1:length(coal)
    rte = RTE(coal(i:i+T,:));
    Vdis = VectorDistancias(coal(i:i+T,:),rte);
    [v,B]=Beneficio(Vdis,m,T);
    VectB15(i)=B;
end
```

Función que con los vectores de beneficios calculados crea una única matriz de beneficios. Se procederá igual para las posibles combinaciones obteniendo B1, B2, B3, B12, B23, B13 y B123.

```
VectB1=[VectB1 zeros(1,length(VectB8)-length(VectB1))];
VectB2=[VectB2 zeros(1,length(VectB8)-length(VectB2))];
VectB3=[VectB3 zeros(1,length(VectB8)-length(VectB3))];
VectB4=[VectB4 zeros(1,length(VectB8)-length(VectB4))];
VectB5=[VectB5 zeros(1,length(VectB8)-length(VectB5))];
VectB6=[VectB6 zeros(1,length(VectB8)-length(VectB6))];
VectB7=[VectB7 zeros(1,length(VectB8)-length(VectB7))];
VectB8=[VectB8 zeros(1,length(VectB8)-length(VectB8))];
VectB9=[VectB9 zeros(1,length(VectB8)-length(VectB9))];
VectB10=[VectB10 zeros(1,length(VectB8)-length(VectB10))];
VectB11=[VectB11 zeros(1,length(VectB8)-length(VectB11))];
VectB12=[VectB12 zeros(1,length(VectB8)-length(VectB12))];
VectB13=[VectB13 zeros(1,length(VectB8)-length(VectB13))];
VectB14=[VectB14 zeros(1,length(VectB8)-length(VectB14))];
VectB15=[VectB15 zeros(1,length(VectB8)-length(VectB15))];

B1=[ VectB3; VectB4; VectB5; VectB6; VectB7; VectB8; VectB9; VectB10;
VectB11; VectB12; VectB13; VectB14; VectB15];
```

Función que calcula la matriz de beneficios total para cada robot. Se prodcederá igual para el robot 2 y el 3. Obteniendo M1, M2 y M3.

```
function M1=BeneficioR1(B1,B12,B13,B123)

B1=[B1 zeros(15,length(B123)-length(B1))];
B12=[B12 zeros(14,length(B123)-length(B12))];
B13=[B13 zeros(14,length(B123)-length(B13))];
B123=[B123 zeros(13,length(B123)-length(B123))];

M1=[B1; B12; B13; B123];
end
```

Función que calcula la coalición de máximo beneficio para cada robot.

```
function [CoalicionMaxM1,fil,col,max]=PreferenciaR1(M1)
max=0;
for i=1:size(M1,1)
    for j=1:size(M1,2)
        if M1(i,j)>max
            max=M1(i,j);
            fil=i;
            col=j;
        end
    end
end
robot1x=205.13;
robot1y=154.21;
robot2x=205.13;
robot2y=83.02;
robot3x=111.83;
robot3y=77.87;
if fil<=15
 R1=Robot1(robot1x,robot1y);
 m=1;  T=fil;
 coal=Rob(T,R1);
 CoalicionMaxM1=coal(col:col+fil+m-1,:);
elseif fil>15 && fil<=29
     R12=Robot12(robot1x,robot1y,robot2x,robot2y);
     m=2;  T=fil-15+1;
     coal=Rob(T,R12);
     CoalicionMaxM1=coal(col:col+T+m-1,:);
elseif fil>29 && fil<=43
     R13=Robot13(robot1x,robot1y,robot3x,robot3y);
     m=2;  T=fil-29+1;
     coal=Rob(T,R13);
     CoalicionMaxM1=coal(col:col+T+m-1,:);
elseif fil>43
     R123=Robot123(robot1x,robot1y,robot2x,robot2y,robot3x,robot3y);
     m=3;  T=fil-43+2;
     coal=Rob(T,R123);
     CoalicionMaxM1=coal(col:col+T+m-1,:);
end
end
```

Por último, se comprueba las coaliciones que maxifican el beneficio guardando estas coalición en la historia del robot.

```
iter=4;
for i=1:iter
[CoalicionMaxM2,fil,col]=PreferenciaR2(M2);
h(i)={CoalicionMaxM2};
M1(fil,col)=0;
[CoalicionMaxM2,fil,col]=PreferenciaR2(M2);
end
```