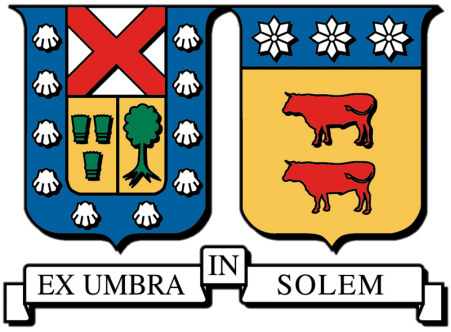

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INDUSTRIAS

ANÁLISIS DE DESEMPEÑO DE UN SISTEMA DE PICKING COLABORATIVO HUMANO-ROBOT CON CONFIGURACIÓN BASADA EN ENJAMBRE

TESIS PARA OPTAR AL TÍTULO DE INGENIERO CIVIL INDUSTRIAL

AUTOR

DIEGO NICOLÁS SALINAS MAURER

PROFESOR GUÍA

RAUL STEGMAIER BRAVO

PROFESOR CO-REFERENTE

PABLO VIVEROS

VALPARAÍSO DE CHILE, 26/11/2025

CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): X Memoria o trabajo de título ☐ Tesis de Postgrado

Título del trabajo: Análisis de Desempeño de un Sistema de Picking Colaborativo Humano-Robot con Configuración Basada en Enjambre

Nombre del candidato(a): Diego Nicolás Salinas Maurer

Carrera / Grado: Ingeniería Civil Industrial

Campus: Casa Central **Departamento:** Industrias

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Raúl Stegmaier Bravo, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente

DEJO CONSTANCIA que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución.

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL (marcar una opción)

X El trabajo **NO contiene** información que amerite confidencialidad y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (**embargo**) por **(marcar una opción)**:

☐ 6 meses ☐ 12 meses ☐ 2 años ☐ 3 años ☐ 5 años ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 26/11/25 **Firma:**

Estudiante o Candidato(a):

Fecha: 26/11/2025 **Firma:**

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

Tabla de Contenidos

1. Resumen

En el contexto de una industria del e-commerce en rápido crecimiento, especialmente acelerado por la pandemia del COVID-19, y los cambios en el comportamiento de los clientes producidos desde entonces, es decir, clientes que demandan pedidos de bajo volumen con gran variedad y esperan una alta calidad de servicio, además de plazos de entrega cortos, las empresas deben adaptar su logística para seguir siendo competitivas en este nuevo escenario empresarial. Además, en los últimos años la Industria 4.0 ha tenido lugar, la cual se basa, en términos generales, en la conectividad, la analítica de datos y la inteligencia de datos, la potencia computacional, la ingeniería avanzada y la interacción hombre-máquina. Este estudio explorará un sistema de picking colaborativo humano-robot con configuración de enjambre a través de una simulación desarrollada en Python. Consiste en un sistema de picking en el que los robots consolidan pedidos, viajan a las ubicaciones de picking de forma autónoma y esperan a que un recolector o picker disponible realice la actividad de picking, y, una vez finalizada la misión de picking, el robot se dirige al almacén y continúa operando hasta el final del turno. El objetivo de esta tesis es analizar cómo cambia el rendimiento de este sistema de picking (el tiempo medio de producción de pedidos) y la utilización de los agentes (utilización media de los robots y utilización media de los recogedores) cuando: el número de recogedores y el número de robots cambian manteniendo constante la tasa de llegada de pedidos (RQ1 y RQ2), y el número de recogedores, el número de robots y la tasa de llegada de pedidos cambian manteniendo constante el ratio entre el número de recogedores y el número de robots (RQ3 y RQ4). Tras ejecutar la simulación, se explican en profundidad los resultados mediante varios gráficos y tablas. Se observa que al aumentar el número de recogedores y robots mejora el rendimiento del sistema, pero la mejora producida al añadir más recogedores es mayor,

hasta cierto punto en el que el número de recogedores es suficiente para gestionar la tasa de llegada de pedidos. Además, el aumento de la tasa de llegada de pedidos produce un peor rendimiento si se mantiene un número fijo de agentes. En cuanto a la utilización de los agentes, el aumento de la tasa de llegada de pedidos incrementa la utilización tanto de los recolectores como de los robots. Al aumentar el número de recolectores, disminuye la utilización de los recolectores, pero no afecta a la utilización de los robots. Por último, al aumentar el número de robots aumenta ligeramente la utilización de los recogedores y disminuye la de los robots.

Palabras clave: interacción hombre-máquina, picking, enjambre, simulación, robots colaborativos.

2. Abstract

In the context of a rapidly growing e-commerce industry, particularly accelerated by the COVID-19 pandemic, and the changes in the customer behaviour produced since then, i.e. customers demanding low volume orders with high variety and expecting high service quality plus short delivery times, companies mut adapt their logistics to remain competitive in this new business arena. Moreover, in the recent years Industry 4.0 has been taking place. This new Industrial Revolution is based on, in general terms, connectivity, data analytics and data intelligence, computational power, advanced engineering and human-machine interaction. This study will explore a human-robot collaborative picking system with a swarm configuration through a simulation developed in Python. It consists in a picking system where robots consolidate orders, travels to the picking locations autonomously and wait for an available picker performs the picking activity, and, once finished the picking mission, the robot goes to the depot and then it continues operating until the end of the shift. The aim of this thesis is to analyse how the performance of this picking system (the average order throughput time) and the utilization of the agents (average robot utilization and average picker utilization) change when: the number of pickers and number of robots change while keeping a constant order arrival rate (RQ1 and RQ2), and the number of pickers, number of robots and the order arrival rate change while keeping the ratio between the number of pickers and the number of robots constant (RQ3 and RQ4). After running the simulation, the output is deeply explained using several plots and tables. It was observed that by increasing the number of pickers and robots the performance of the system improves, but the improvement produced by adding more pickers is more important, until certain point in which the number of pickers is enough to manage the order arrival rate. Also, increasing the order arrival rate produces a worse perfor-

mance if keeping a fixed number of agents. Regarding the utilization of the agents, the increase in the order arrival rate increases both pickers and robots' utilization. By augmenting the number of pickers, the pickers' utilization decreases but it does not affect the utilization of the robots. Finally, increasing the number of robots slightly increases the utilization of the pickers and decreases the utilization of the robots.

Key-words: human-machine interaction, picking, swarm, simulation, collaborative robots.

3. Introducción

En los últimos años, el e-commerece ha experimentado un rápido crecimiento, que se vio especialmente acelerado por la pandemia del COVID-19. Como muestra la Figura 1, los ingresos del mercado del comercio electrónico en Europa aumentaron considerablemente en 2021. También muestra que el comercio electrónico va a seguir creciendo, alcanzando casi un 60 % de crecimiento en 2027 con respecto a 2020. Esto establece claramente que el e-commerce está aquí para quedarse, y las empresas deben adaptar su logística para seguir siendo competitivas en este nuevo escenario empresarial.

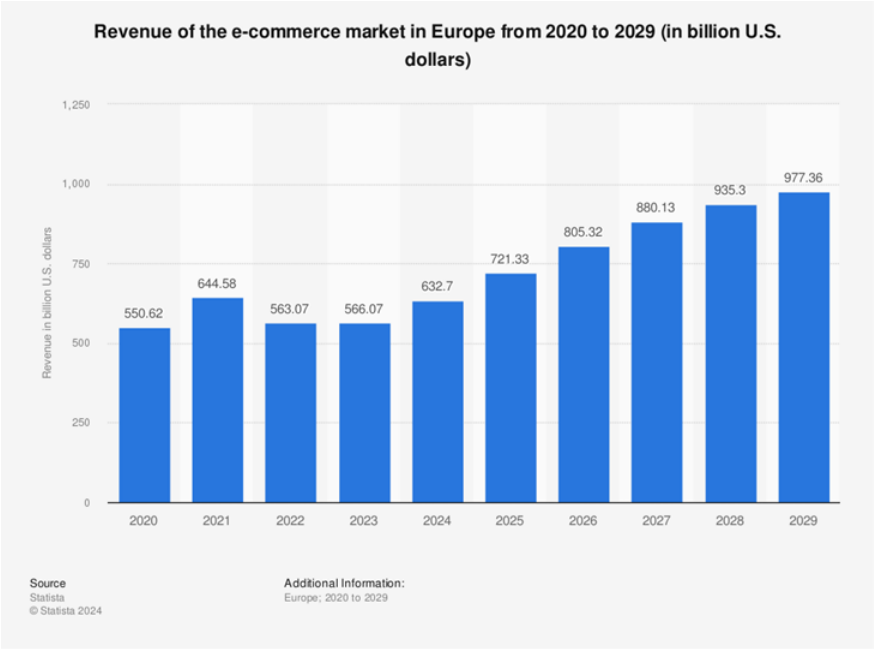

Figura 1: Ingresos del mercado del e-commerce en Europa del 2000 al 2029.

Además de esta tendencia en el canal de distribución, los clientes exigen bajos volúmenes y gran variedad al mismo tiempo, lo que hace aún más complejo para las empresas satisfacerlos. Además, los clientes esperan una alta calidad de servicio con plazos de entrega cortos. Para satisfacer los requerimientos de los clientes y seguir siendo competitivas, las empresas aspiran ahora a

entregas incluso en el mismo día. Según Straits Research, se prevé que este mercado crezca a un ritmo del 21,3 % entre 2024 y 2032.

Todos estos elementos representan importantes retos para las empresas. Para seguir las tendencias actuales y el comportamiento de los clientes, las empresas deben revisar sus actividades logísticas. Entre todas las actividades logísticas, este estudio se centra en la actividad de picking, que representa hasta un 50 % del costo operativo de un centro de distribución.

La actividad de picking es el proceso en el que se recogen los artículos para satisfacer los pedidos. Para llevar a cabo esta tarea existen varios sistemas de preparación de pedidos; este estudio se centra en el sistema de preparación de pedidos Picker-to-Parts, en el que el preparador de pedidos se desplaza dentro de una instalación a varias ubicaciones con el fin de completar todo el pedido para poder enviarlo al cliente. Dada la naturaleza de esta actividad, que consiste en tomar artículos de diferentes tamaños y formas, requiere mucha mano de obra, ya que se realiza principalmente de forma manual. De hecho, en Europa solo un 5 % de los almacenes están totalmente automatizados, según Mordor Intelligence.

Por otro lado, durante los últimos años se ha estado produciendo la Cuarta Revolución Industrial o Industria 4.0. Este concepto hace hincapié en:

- **Conectividad, datos y potencia computacional:** tecnología en la nube, Internet, block-chain, sensores.
- **Analítica e inteligencia:** analítica avanzada, aprendizaje automático, inteligencia artificial.
- **Interacción humano-máquina:** realidad virtual (RV) y realidad aumentada (RA), robótica y automatización, vehículos autónomos.
- **Ingeniería avanzada:** fabricación aditiva (por ejemplo, impresión 3D).

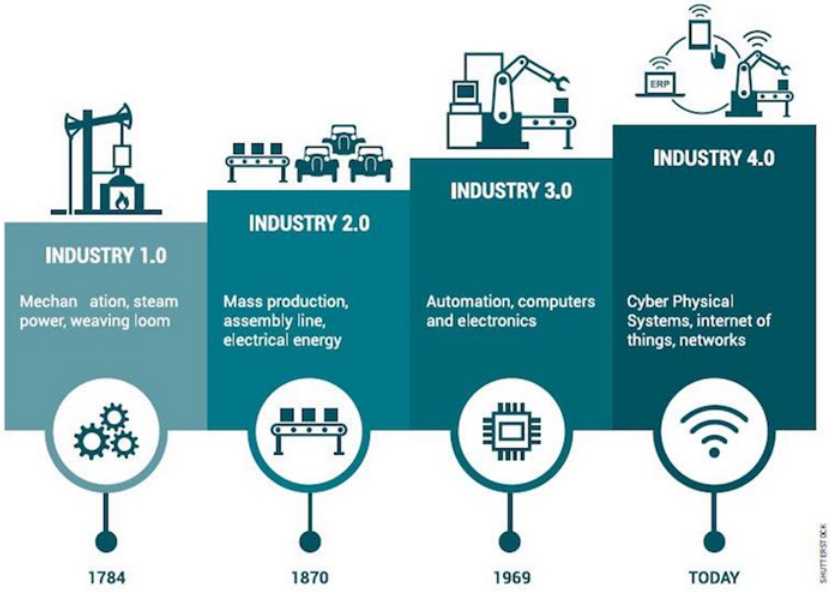

Figura 2: Revoluciones industriales y sus principales características.

Para hacer frente a los nuevos retos en las operaciones de almacén, los sistemas inteligentes basados en la interacción hombre-máquina que utilizan vehículos autónomos son el centro de varios estudios y han sido implementados por muchas empresas utilizando muchas configuraciones diferentes.

La idea principal de estos sistemas interactivos que utilizan humanos con robots colaborativos es aprovechar los puntos fuertes de ambos agentes: los robots colaborativos son más precisos y eficientes que los humanos a la hora de realizar tareas repetitivas y físicamente exigentes, mientras que los humanos son mejores identificando los distintos objetos y agarrando objetos de distintos tamaños y formas colocados de distintas maneras; los robots son incapaces de reproducir totalmente esta tarea, a pesar de su precisión y eficiencia.

Mientras que en 2019 la tasa mundial de adopción de tecnología en robótica y automatización en almacenes fue del 36 %, se prevé que sea del 85 % en 2030 y que el mercado de automatización de almacenes en todo el mundo crezca de 2023 a 2027 a una CAGR del 15 %, como se

muestra en la Figura 3.

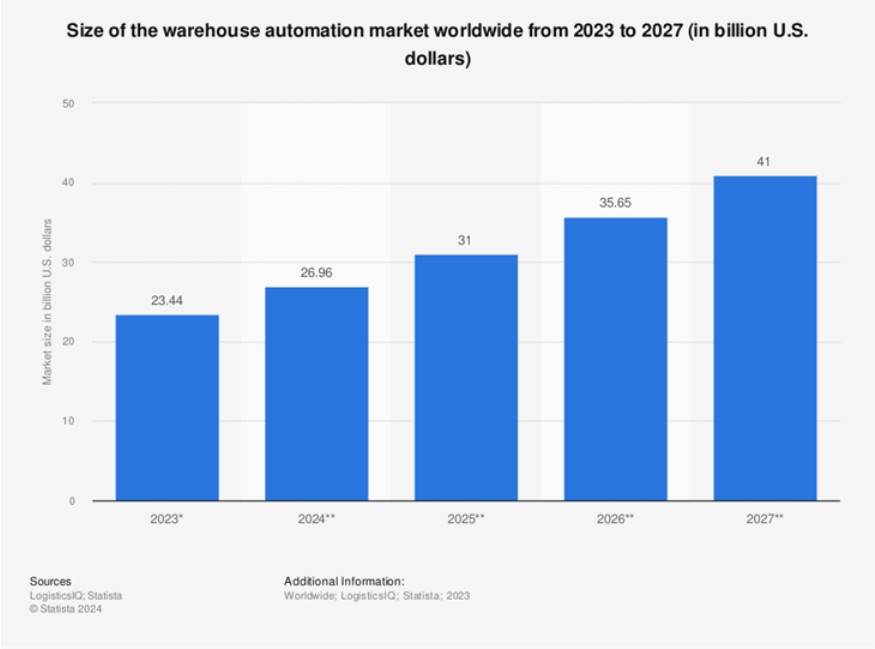

Figura 3: Tamaño del mercado mundial de automatización en bodegas entre 2023 y 2027.

Sin embargo, como ya se ha explicado, automatizar por completo el proceso de picking plantea un gran reto: los humanos ofrecen una flexibilidad inigualable a la hora de manejar tareas diversas y situaciones inesperadas. En cambio, los robots, aunque son muy eficaces en tareas repetitivas, tienen dificultades para adaptarse al mismo nivel. De ahí que un enfoque híbrido de colaboración humano-robot parezca la solución más valiosa para los sistemas de picking.

En un sistema de picking picker-to-parts puede haber muchas configuraciones diferentes a la hora de adoptar robots colaborativos que interactúen con humanos. Por lo general, los robots se desplazan a las diferentes ubicaciones de picking junto con los humanos, indicándoles las ubicaciones que deben visitar, el artículo específico que deben recoger en cada ubicación y la cantidad, y también realizan la función de cesta, recogiendo todos los artículos del pedido. En otras configuraciones, los robots y los humanos trabajan de forma independiente: los humanos recogen un

tipo de artículos, mientras que los cobots recogen otro tipo de artículos, que son fáciles de recoger por los cobots y, normalmente, demasiado pesados para los humanos. Hay otra configuración en la que el cobot no está acoplado al humano mientras se desplaza, sino sólo cuando llega a un lugar de recogida, donde el robot espera a un humano que realizará la recogida, y entonces se vuelven a desacoplar. Existen varias configuraciones más que ya se están implementando o estudiando, pero este estudio se centrará en esta última que se denomina Enjambre (Swarm en inglés).

Los sistemas colaborativos humano-robot en operaciones de almacén son un tema reciente que sigue siendo objeto de investigación, ya que se están desarrollando tecnologías innovadoras y, por tanto, surgen nuevas configuraciones posibles. En esta tesis se desarrolla una simulación de diferentes escenarios del sistema de picking de un almacén genérico con el fin de analizar el rendimiento de un sistema de picking colaborativo hombre-robot utilizando una configuración Swarm.

- **RQ1:** ¿Cuál es el impacto en el rendimiento del sistema cuando se varía el Número de Recogedores y el Número de Robots manteniendo una Tasa de Llegada de Pedidos fija?
- **RQ2:** ¿Cuál es el impacto en la utilización de los recursos del sistema cuando se varía el Número de Recogedores y el Número de Robots manteniendo una Tasa de Llegada de Pedidos fija?
- **RQ3:** ¿Cuál es el impacto en el rendimiento del sistema cuando se varía la Tasa de Llegada de Pedidos mientras se mantiene una proporción fija entre el Número de Recogedores y el Número de Robots?
- **RQ4:** ¿Cuál es el impacto en la utilización de los recursos del cuando se varía la Tasa de Llegada de Pedidos mientras se mantiene una proporción fija entre el Número de Recogedores

y el Número de Robots?

Para responder adecuadamente a estas preguntas de investigación, se desarrollará un modelo de simulación de un sistema genérico de picking-to-parts hombre-robot con una configuración de Enjambre. Los objetivos generales y específicos se expondrán en la sección 4. La sección 5 cubrirá la metodología, la cual se dividirá en dos partes: la descripción del sistema, considerando el layout y el comportamiento general de los agentes, y la Simulación, pasando por las suposiciones y simplificaciones, el tipo de simulación y los algoritmos desarrollados. En la sección 6 se realizará el análisis de los resultados. Posteriormente, en la sección 7 se presentarán las conclusiones, dando respuesta a cada pregunta de investigación y, finalmente, en la sección 8, se expondrán las Limitaciones de este estudio.

4. Objetivos

4.1. Objetivo General

Analizar el desempeño de un sistema de picking colaborativo humano-robot con una configuración Swarm, evaluando su impacto en la eficiencia operativa, la utilización de recursos y el tiempo de procesamiento de pedidos a través de simulaciones en distintos escenarios.

4.2. Objetivos Específicos

1. Evaluar el impacto de la variación en la cantidad de pickers y robots en el rendimiento del sistema, midiendo el tiempo promedio de procesamiento de pedidos y la eficiencia operativa.

2. Determinar cómo afecta la tasa de llegada de pedidos al desempeño del sistema y a la utilización de los pickers y robots, manteniendo una proporción fija entre ambos agentes.

3. Desarrollar un modelo de simulación en Python para representar con precisión el comportamiento del sistema, permitiendo la experimentación con diferentes configuraciones y estrategias de asignación de tareas.

5. Metodología

5.1. Descripción del sistema

En este Capítulo se explicará detalladamente el sistema diseñado, incluyendo el Layout, la definición de los elementos con sus comportamientos y el comportamiento global del sistema. Para ello, las descripciones irán acompañadas de diagramas de flujo. En primer lugar, se explicará la disposición y, a continuación, se describirá el sistema en su conjunto. A continuación, se definirán los elementos objeto de análisis y se describirán uno por uno.

5.1.1. Layout

Como el objetivo de la tesis es analizar el rendimiento de la actividad de picking utilizando un sistema colaborativo, en particular, un sistema de enjambre que involucra a recolectores y robots, se eligió un almacén genérico. Esta disposición consta de:

- Lugares de recogida.
- Depósitos.
- Estaciones de carga para los robots.
- Un pasillo transversal

La Figura 4 muestra el Layout. Al tratarse de un almacén genérico, las dimensiones de los bloques y las del propio almacén son fácilmente modificables con los parámetros M y N. Además, los depósitos y las estaciones de carga pueden colocarse en distintos lugares. La principal ventaja de esta disposición es su gran flexibilidad. La principal ventaja de este diseño es su alta flexibilidad, lo que permite evaluar diferentes configuraciones en la simulación.

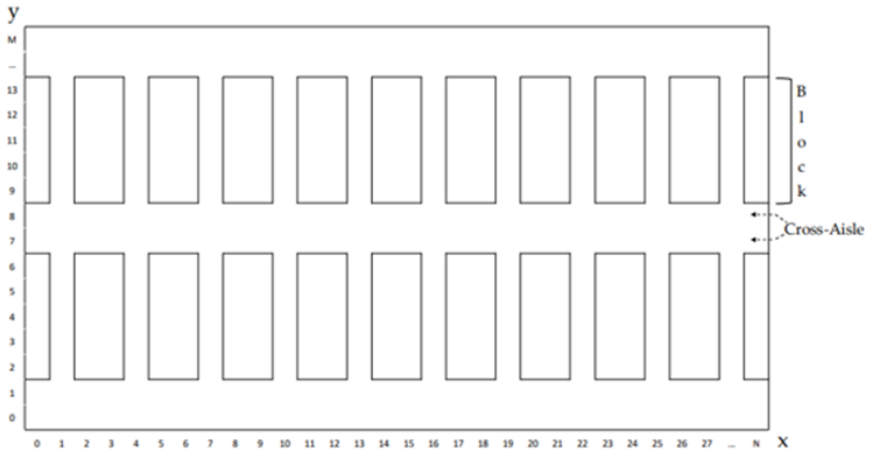

Figura 4: Layout.

5.1.2. Sistema general

El sistema analizado es un sistema de picking colaborativo robot-humano, en concreto, con funcionamiento de Enjambre. La principal distinción entre una operación de enjambre y otras operaciones de picking es que en la de enjambre, un robot desacoplado va a una ubicación de picking, donde espera a un recolector que será asignado al robot y realizará el picking; por lo tanto, se acoplan y, finalmente, se vuelven a desacoplar para que el robot pueda ir a otra ubicación de picking y el recolector pueda ir a otro robot.

Para comprender mejor el sistema global, considere el Layout ya presentado y los siguientes elementos:

- **Pedidos:** los pedidos están compuestos por líneas de pedido de diferentes artículos y, para cada línea de pedido, la cantidad respectiva de ese artículo, como se ilustra en la Figura 5. Los pedidos son una variable externa, ya que proceden de los clientes, y pueden llegar en cualquier momento en el sector del comercio electrónico minorista.

- **Robots:** en este caso, como están en colaboración con humanos, también pueden llamarse

Item	Quantity
Item 1	Qty 1
Item 2	Qty 2
Item 3	Qty 3
...	...
Item N	Qty N

Order lines

Figura 5: Ilustración de un pedido.

cobots (robots colaborativos) y, dado el funcionamiento en enjambre, pueden denominarse robot enjambre. La tecnología específica que se toma en consideración es el Robot Locus. Este tipo de robot no tiene funciones para realizar el picking por si solo, sino que de desplazamiento autónomo y tiene el espacio para los artículos recogidos por otro agente, en este caso, un humano. Estos robots pueden moverse de forma autónoma según reglas programadas e interactuar con su entorno para, por ejemplo, evitar obstáculos y colisiones con otros robots o humanos. También son muy flexibles en cuanto al número de pedidos que pueden gestionar al mismo tiempo, ya que son posibles muchas configuraciones, como muestra la Figura 6. El número de robots, su velocidad máxima, el número máximo de pedidos simultáneos y las reglas que siguen al desplazarse por la instalación son decisiones de diseño que pueden afectar al rendimiento del sistema de picking y a la utilización de los agentes.

Figura 6: Locus Robot.

- **Recolectores:** son los encargados de realizar el picking, en este caso serán humanos. Los preparadores tienen que ir a las ubicaciones de picking indicadas donde espera un robot, realizar el picking y colocar los artículos en la cesta correcta del robot, y confirmar que se ha realizado el picking para que el robot pueda desacoplar al humano e ir a otra ubicación. Esto se ilustra en la Figura 7. El número de recolectores y las reglas que siguen en la asignación de los robots son decisiones de diseño que también pueden influir en el rendimiento del sistema de picking y en la utilización de los agentes.

Figura 7: Ilustración de recolectores con Locus Robots.

El sistema global se modela en el diagrama de flujo de la Figura 8. Los pedidos se asignan a los robots disponibles. A continuación, los robots comienzan la misión de picking dirigiéndose a cada ubicación de picking. En cada ubicación de picking, el robot espera a un recolector. Cuando se asigna un recolector a un robot, se acoplan y el recolector se desplaza a la ubicación de picking, realiza el picking y coloca los artículos en la cesta del robot. Una vez finalizado el picking, se desacoplan y continúan trabajando como se ha descrito. Cuando se han recogido todos los artículos, él se dirige al almacén con el pedido completo. La operación se repetirá hasta que finalice el turno.

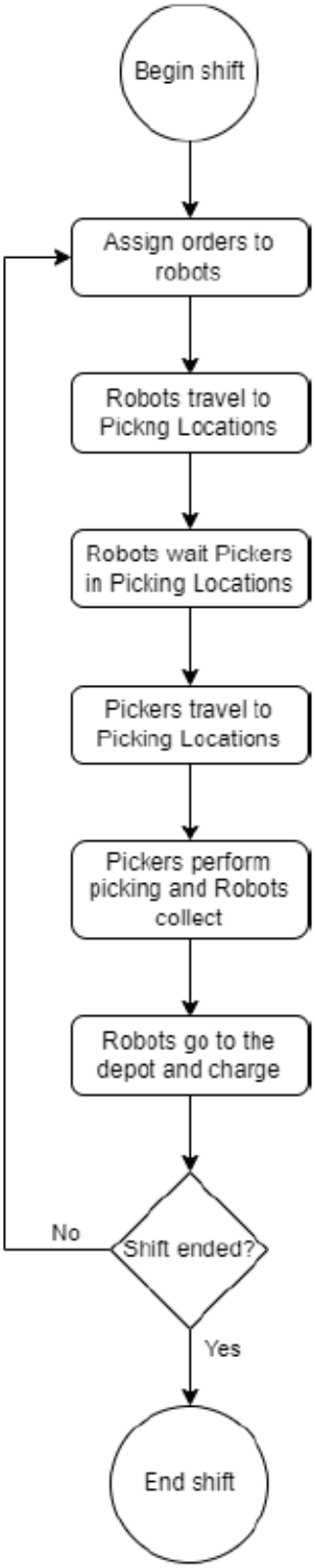

Figura 8: Diagrama de flujo del sistema.

El comportamiento detallado de los pedidos, robots y pickers en cada una de estas etapas es más complejo y no se ha descrito en esta sección, ya que aquí se proporciona una visión general del sistema. Una descripción más detallada de las operaciones de cada agente y del procesamiento de pedidos se presentará en las siguientes secciones de este capítulo.

5.1.3. Pedidos

Como se explicó en la sección anterior, los pedidos pueden estar compuestos por varias líneas de pedido y cada línea de pedido puede tener cantidades diferentes. Además, no hay decisiones de diseño sobre el comportamiento de los pedidos, ya que son un factor externo que proviene directamente de los clientes. Por lo tanto, los pedidos se consideran como dados y las decisiones de diseño deben tomarse para cumplir con los pedidos de la mejor manera posible.

Hay varias formas de modelar los pedidos sin ser demasiado arbitrarios, una de ellas es teniendo en cuenta el perfil de los pedidos, por lo tanto, el número de líneas de pedido y las cantidades se basan en probabilidades de ocurrencia y una distribución de probabilidad para la llegada de los pedidos. Estas determinaciones se explicarán en el capítulo siguiente.

La Figura 9 representa el comportamiento detallado paso a paso de un único pedido en este sistema concreto. Debe entenderse como la forma en que se procesan todos los pedidos. También es importante registrar las distintas etapas por las que pasa el pedido en términos de tiempo. En cuanto llega el pedido, se identifican los siguientes estados:

- Asignable, pero a la espera de ser asignado a un robot.
- Asignado a un robot, pero a la espera de que este comience la misión de picking.
- En proceso, pero aún no está terminado.

- Pedido procesado, terminado.

El procesamiento de los pedidos es el siguiente:

- El pedido llega al sistema, por lo que es asignable.
- Cuando el pedido se asigna a un robot, registra el tiempo transcurrido desde que llegó el pedido hasta ese momento. Esto se define como el Tiempo de asignación del pedido en espera (WOAT).
- Cuando el robot al que se asignó la orden comienza a desplazarse, registra el tiempo transcurrido desde que se asignó la orden a ese robot hasta ese momento. Esto se define como el Tiempo de Movimiento en Espera (WMT).
- Cuando el robot al que se ha asignado el pedido finaliza la misión de picking, registra el tiempo transcurrido desde que el robot inició la misión de picking hasta ese momento. Esto se define como el Tiempo de Proceso (TP).
- En este punto, este único pedido ya no es objeto de análisis, puesto que está fuera del sistema; ya fue procesado.

Figura 9: Diagrama de flujo de un pedido.

5.1.4. Robots

Como se ha anticipado, la tecnología específica considerada son los Robots Locus, un modelo de robot muy flexible que permite varias decisiones de diseño que pueden repercutir en el rendimiento del sistema de picking, así como en la utilización de los agentes: robots y recolectores. Entre las decisiones de diseño se encuentran:

- Número de robots.
- Velocidad máxima de los robots.
- Número máximo de pedidos procesados simultáneamente por un robot.
- Política de movimientos dentro de la instalación.
- Reglas de cola en la asignación de los pedidos.
- Reglas de cola en la asignación de los operarios.

En cuanto a las órdenes, es fundamental registrar varios indicadores para analizar el comportamiento de los robots y del sistema en general. Para ello, se va a registrar la cantidad de tiempo empleado en cada paso de la operación de los robots, considerando los siguientes estados:

- **Carga:** para cada robot se registra el **Tiempo de Carga (TC)**.
- **Órdenes en espera:** es el tiempo transcurrido desde que el robot está disponible para recibir órdenes hasta que comienza a desplazarse para cumplir la misión de picking. Se define como **Tiempo de Espera de Pedidos (WOT)**.

- **Desplazamiento:** la cantidad de tiempo que el robot se desplaza, ya sea a una ubicación de recogida o a un almacén. Se define como **Tiempo de Desplazamiento del Robot (RTT)**.

- **Espera de recogedor:** la cantidad de tiempo que transcurre desde que un robot llega a una ubicación de recogida hasta el momento en que un recogedor llega a su posición. Se define como **Tiempo de Espera del Recolector (WPT)**.

- **Recogida:** El **Tiempo de Picking del Robot (RPT)** es el tiempo empleado por el robot mientras el recolector realiza la actividad de picking.

El funcionamiento de los robots en la operación de enjambre del sistema, como también se ilustra en la Figura 10, es el siguiente:

- Un robot se carga hasta un nivel de batería previamente determinado con lo cual se vuelve disponible.

- Cuando el robot está disponible, puede recibir pedidos.

- Tras recibir un número previamente determinado de pedidos o tras esperar un tiempo previamente determinado una vez que ya tiene al menos un pedido, el robot comienza a desplazarse para completar la misión de picking. Para ello, el robot tiene que consolidar con reglas establecidas la trayectoria que va a seguir.

- El robot sigue su trayectoria hasta una ubicación de recogida, donde espera a un recogedor.

- Una vez que un recolector, siguiendo las reglas establecidas, se asigna a un robot, se dice que están acoplados.

- Cuando el recolector llega a la posición de su robot asignado, se lleva a cabo el picking. En cuanto el recogedor finaliza esta actividad, se desacoplan.

- El robot se dirige a la siguiente ubicación de picking, si aún quedan líneas de pedido pendientes, o se dirige al almacén en caso de que ya haya cumplido todos los pedidos asignados en la misión de picking actual.

- Cuando el robot termina la misión de recogida, debe cargarse hasta el nivel en que pueda volver a estar disponible.

- A continuación, el robot realiza un bucle en este flujo de trabajo hasta que finaliza el turno.

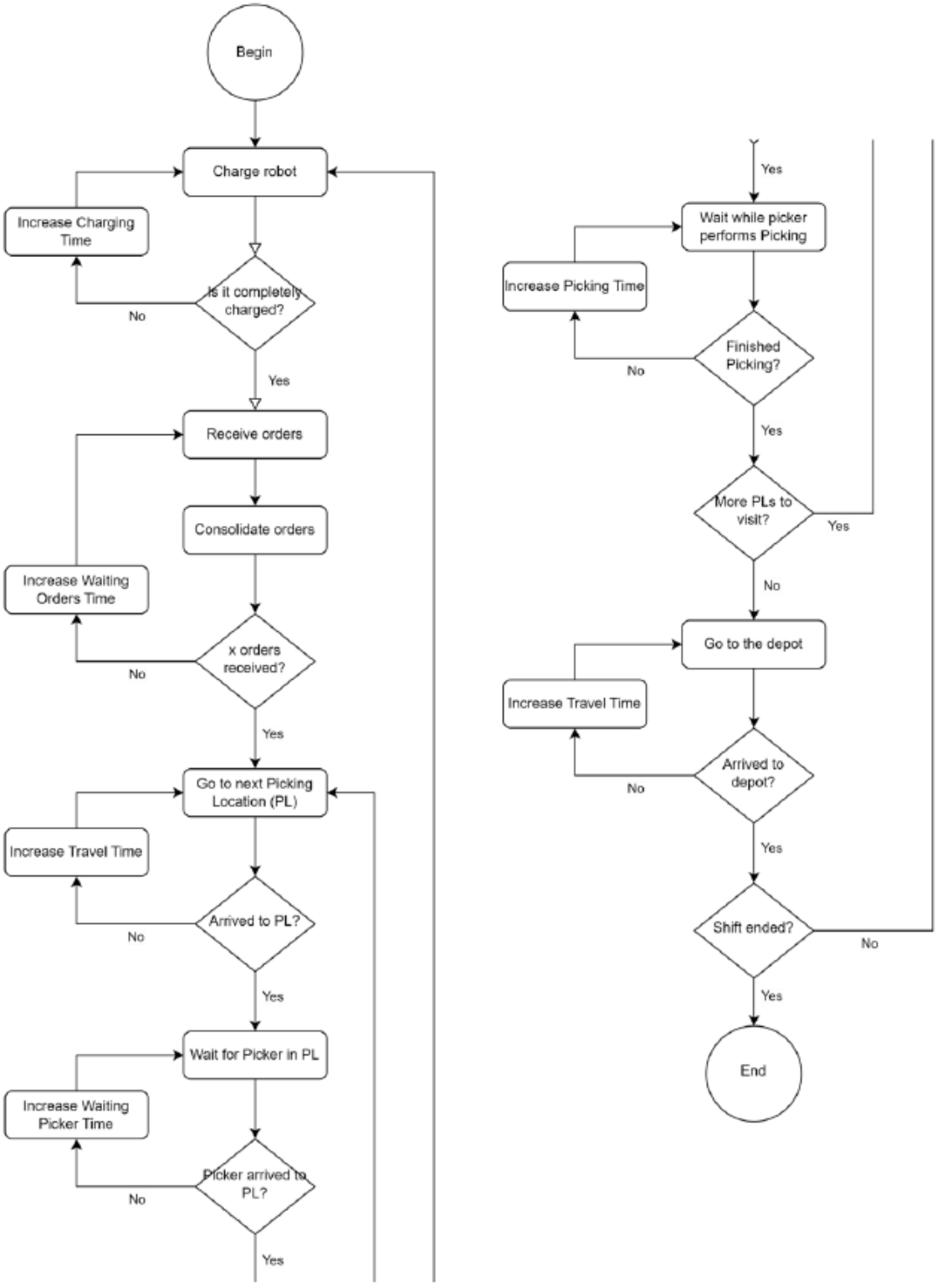

Figura 10: Diagrama de flujo de un robot.

5.1.5. Recogedores

Por último, los recolectores, humanos que realizan la actividad de picking, en este caso, en colaboración con los robots bajo la operación de enjambre, también están sujetos a decisiones de diseño, como las siguientes:

- Número de recolectores.
- Política de movimientos dentro de la instalación.
- Política de asignación de operarios a robots.

En cuanto a los pedidos y los robots, se definen algunos indicadores de tiempo para registrar cada estado de los recogedores:

- **Espera de robot:** cuando un recolector está disponible, es decir, no asignado (o acoplado) a ningún robot, en realidad está esperando a ser asignado a uno de ellos. Esto se define como **Tiempo de Espera de Robot (WRT)**.
- **Desplazamiento:** tan pronto como un recolector es asignado a un robot (acoplado) comienza a desplazarse hasta la ubicación de picking donde se el robot. Esto se define como el **Tiempo de Desplazamiento del Recolector (PTT)**.
- **Picking:** finalmente, cuando el recolector llega a la ubicación de picking, realiza la actividad de picking, colocando los artículos en la cesta del robot. Esto se define como el **Tiempo de Picking del Recolector (PPT)**.

Del mismo modo, como ya se ha descrito a través de los indicadores, el funcionamiento de un solo recogedor, ilustrado en la Figura 11, es el que se detalla a continuación:

- El recogedor está disponible y espera a ser asignado a un robot según la política establecida. Cuando es asignado a un robot se dice que está acoplado a ese robot y se le da la ubicación de picking.

- El recolector comienza a desplazarse hasta el lugar de recogida.

- Cuando llega, realiza la actividad de recogida.

- Una vez finalizada la actividad de recogida, está desacoplada y, por tanto, disponible de nuevo.

- El recolector realiza un bucle en este flujo de trabajo hasta que finaliza el turno.

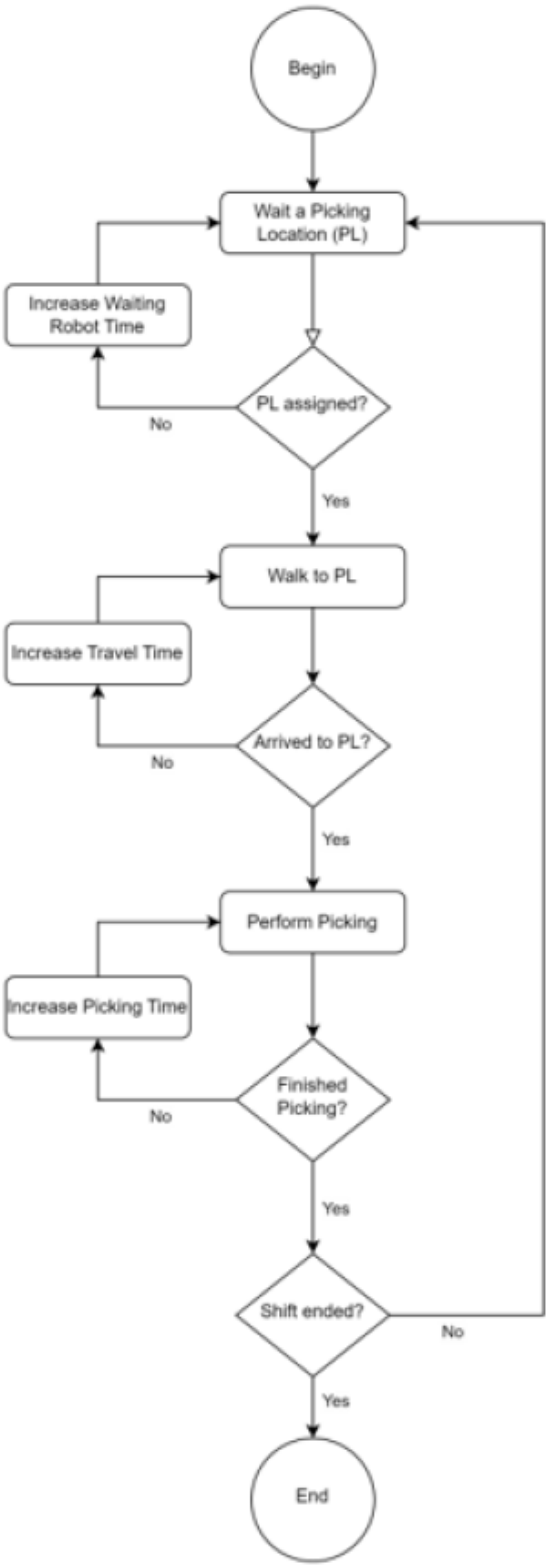

Figura 11: Diagrama de flujo de un recogedor.

5.2. Simulación

En este capítulo se proporcionarán todos los detalles de la simulación del sistema, incluyendo la definición del problema y los objetivos, la recopilación de datos, el desarrollo del modelo conceptual, la selección del método de simulación y de la herramienta de simulación y el desarrollo del modelo de simulación. Este capítulo se basa en el marco de la Figura 12, que sugiere adecuadamente paso a paso cómo desarrollar un estudio de simulación. En particular, en este capítulo se describirán las fases 1 a 6, mientras que las fases 7 a 9 se explicarán en el capítulo siguiente y la fase 10 se tratará en la Discusión. Para la fase 3, remítase a la sección 5.1 Descripción del Sistema, donde se describe el sistema y se proporcionan diagramas de flujo del comportamiento general de los agentes; en este capítulo solo se proporcionarán especificaciones referidas a dichas explicaciones, basadas en los supuestos considerados para esta simulación, que se presentarán en los siguientes apartados y utilizando también los parámetros expuestos en la fase 2.

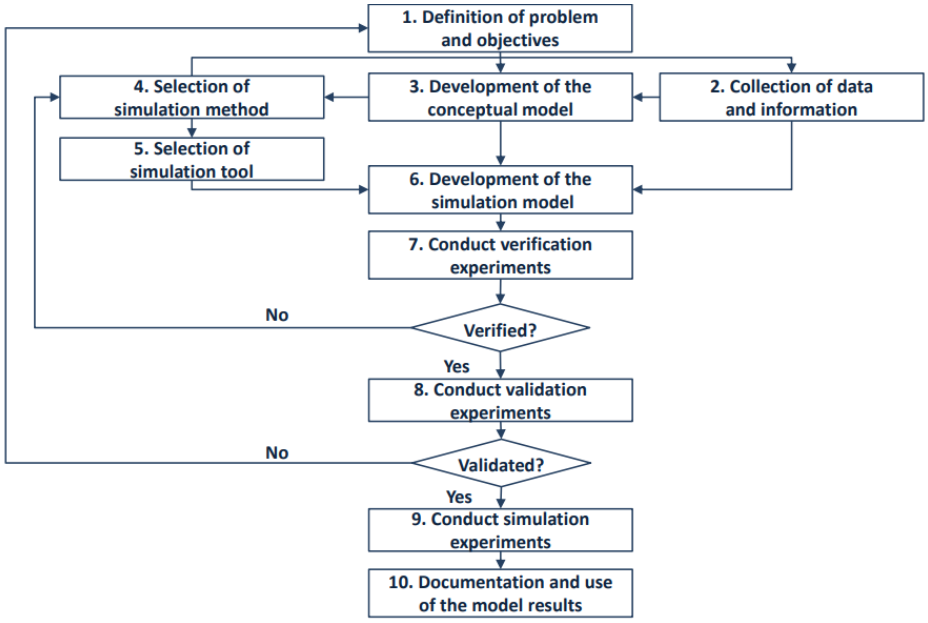

Figura 12: Fases para desarrollar un proyecto de simulación.

5.2.1. Definición del problema y objetivos

El objetivo de este estudio es analizar el rendimiento de un sistema de picking interactivo humano-robot en el contexto de la creciente industria del comercio electrónico en el sector minorista y en plena cuarta Revolución Industrial. En concreto, el sistema de picking de interés tiene un funcionamiento en enjambre mediante Robots Locus.

El modelo considera las siguientes simplificaciones e hipótesis:

- Sólo se modela la actividad de picking, no la reposición.
- No se modelan las colisiones de ningún tipo.
- No se han modelado las colas en los puntos de picking.
- Los artículos estarán siempre disponibles.
- La cantidad de una línea de pedido no es relevante, la única información importante es la ubicación de picking de esa línea de pedido.
- Los elementos no están definidos y no hay diferencias entre ellos desde el punto de vista del modelo.
- La ubicación de picking de una línea de pedido se asigna aleatoriamente.
- La llegada de pedidos sigue una distribución exponencial con una tasa de llegada de pedidos constante. Esta elección se justifica porque los tiempos entre llegadas sucesivas que siguen una distribución exponencial implican un proceso de Poisson para el número de llegadas en un intervalo de tiempo dado. El proceso de Poisson es un modelo estándar para eventos que ocurren de forma aleatoria e independiente en el tiempo, lo cual es una suposición razonable

para un sistema de picking genérico donde los pedidos no están correlacionados con eventos pasados y la demanda se considera espontánea y sin memoria (18).

- El perfil de los pedidos viene dado por una distribución de probabilidad fija (3).
- A modo de simplificación, el tiempo de viaje no tiene variabilidad, y sólo depende de la distancia recorrida y de la velocidad del agente.
- A modo de simplificación y para dar un poco de variabilidad, el tiempo de recogida sigue una distribución uniforme.
- Los robots y los recolectores no cometen errores.
- Los robots están disponibles cuando están completamente cargados, las especificaciones se detallaran en la siguiente sección.
- Al principio de la simulación, todos los robots están completamente cargados y situados en los depósitos.
- Los robots solo gastan energía mientras se desplazan.
- Al principio de la simulación todos los recolectores estarán en lugares aleatorios del almacén.
- Los depósitos y las estaciones de recarga se sitúan en los mismos lugares.

El objetivo de la simulación es:

- Estimar cómo cambia el rendimiento del sistema de picking y la utilización de los agentes al variar el número de preparadores y el número de robots manteniendo constante la tasa de llegada de pedidos y al variar la tasa de llegada de pedidos manteniendo fija la relación entre

el número de preparadores y el número de robots. En otras, el objetivo es responder a RQ1, RQ2, RQ3 y RQ4.

Las medidas de rendimiento de interés son:

- El indicador de rendimiento del sistema de picking es el **Tiempo Medio de procesamiento ATT**): la media del tiempo transcurrido desde la llegada de cada pedido. Considere también el Tiempo de Espera de Llegada de Ordenes (WOAT) y el Tiempo de Proceso (PT) que es el tiempo desde que se comienza a procesar el pedido hasta que cada pedido se coloca en el depósito, todos en segundos. La fórmula conceptual es:

$$ATT = \frac{\sum_{o \in Orders} PT_o + WOAT_o}{|Orders|} \tag{1}$$

- **Utilización Media de Robot (ARU):** porcentaje del tiempo medio en el que cada robot está realizando tareas activas, como desplazarse y hacer picking. El tiempo de realización de tareas activas se denomina Tiempo Activo del Robot (RAT), en segundos. El resto del tiempo es, en consecuencia, el Tiempo Pasivo del Robot (RPT), en segundos. Además, todo el tiempo puede identificarse como Horizonte temporal (TH), en segundos. Entonces, las ecuaciones a continuación representan el ARU, de dos maneras equivalentes:

$$ARU = \frac{\sum_{r \in Robots} RAT_r}{TH} \tag{2}$$

$$ARU = \sum_{r \in Robots} \frac{RAT_r}{RAT_r + RPT_r} \tag{3}$$

- **Utilización Media del Recogedor (APU):** análogamente, es el porcentaje del tiempo medio

en el que cada recolector está realizando tareas activas. El Tiempo Activo del Recolector (APT) y el Tiempo Pasivo del Recolector (PPT), ambos en segundos, son iguales que el RAT y el RPT, pero referidos al recolector. Las ecuaciones siguientes representan el APU de dos formas equivalentes:

$$APU = \frac{\sum_{p \in Pickers} PAT_p}{TH} \tag{4}$$

$$APU = \sum_{p \in Pickers} \frac{PAT_p}{PAT_p + PPT_p} \tag{5}$$

El horizonte de simulación comprende un turno de 8 horas; por tanto, TH equivale a 28.800 segundos.

5.2.2. Recolección de datos e información

Como se preveía en parte, algunos de los datos se generarán de acuerdo con distribuciones de probabilidad, basadas en estudios anteriores, mientras que otros datos son fijos, también recogidos de estudios anteriores, o asignados arbitrariamente. La simulación considera los siguientes datos:

- Layout: ilustrada en figura 2.2
  - Anchura= 29 metros.
  - Altura= 15 metros.
  - Estaciones de carga y depósitos en las posiciones (0, 0) y (29, 0).
  - 5 posiciones de picking por lado y por bloque.

- Pasillo transversal en medio del almacén.
- Pasillos de 1 metro de ancho.
- Pasillos de 5 metros de altura.
- Los robots se inicializan por igual en uno de los dos depósitos. En caso de número impar de robots, habrá un robot más en el depósito situado en la posición (0, 0).
- Los recolectores se inicializan en cualquier parte del almacén de forma aleatoria.

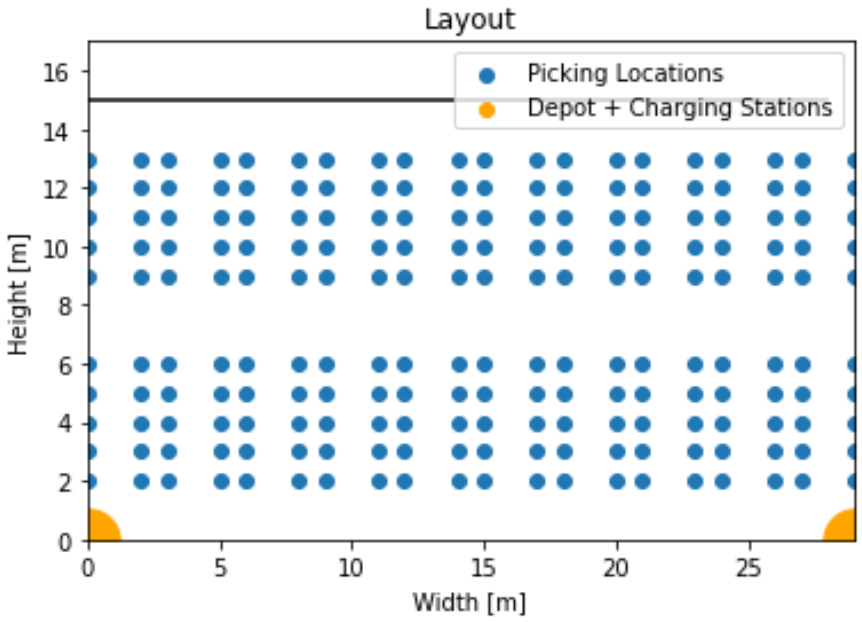

Figura 13: Layout.

- Robots
  - El número robots dependerá del escenario.
  - Todos los parámetros se resumen en Tabla 1.
- Recogedores
  - El número de recolectores dependerá del escenario.

- Velocidad= 0,75 [m/s].
- Velocidad de recogida: devuelta a partir de una distribución uniforme discreta de 2 a 10 segundos.

- Pedidos:
  - La tasa de llegada de pedidos dependerá del escenario.
  - La generación de pedidos sigue el Algoritmo 1. El algoritmo durante el tiempo de simulación genera pedidos en un tiempo dado por un número aleatorio de una distribución exponencial con factor 1 / Tasa de llegada de pedidos.
  - El perfil de los pedidos o la frecuencia de las líneas de pedido sigue la distribución de la Tabla 2 (3). Esto mantiene un perfil de pedidos realista del sector minorista del comercio electrónico que se caracteriza por varios pedidos pequeños con pocas líneas de pedido.
  - Para la generación de pedidos (línea 4 del Algoritmo 1), el número de líneas de pedido sigue el Algoritmo 2. En este algoritmo se supone que existe una estructura de datos que asocia el tamaño del pedido con una distribución de probabilidad acumulada, construida con la Tabla 2. A continuación, para cada pedido se genera un número real aleatorio entre 0 y 1. Finalmente, dada la probabilidad, se calcula el número de líneas de pedido. A continuación, para cada pedido se genera un número real aleatorio entre 0 y 1. Finalmente, dada la probabilidad, el número de líneas de pedido corresponde a la distribución de probabilidad acumulada; si la probabilidad es menor que la distribución de probabilidad acumulada asociada, entonces ese es el número de líneas de pedido de ese pedido, en caso contrario se continúa buscando a través de la estructura de datos.

Tabla 1: Parámetros de los robots

Parameter	Number	Unit of measure
Speed	1	m/s
Battery size	2.5	kWh
Battery consumption	0.9 x 2.5 / 4.5 / 60 / 60	kWh/s
Power of charging station	0.9 x 2.5 / 1.75	kW
Max waiting order time	30	s
Max number of orders	3	#orders

Tabla 2: Distribucion de pedidos

Number of order lines per order	Frequency	Percentage
1	367264	95,42 %
2	10548	2,74 %
3	2087	0,54 %
4	999	0,26 %
5	480	0,12 %
6	617	0,16 %
7	185	0,05 %
8	167	0,04 %
≥ 9	2526	0,66 %

Algorithm 1 Generación de órdenes (enfoque en la llegada de pedidos)

0: $time \leftarrow 0$
0: **while** $time < TH$ **do**
0: $time \leftarrow time +$ integer(random.exponential$(1/$order arrival rate$)$)
0: Generar orden en el instante de tiempo
0: **end while**=0

Algorithm 2 Generación de órdenes (enfoque en la frecuencia de líneas de orden)

0: Generar lista con frecuencia acumulada ordenada de 1 línea de orden a 9
0: $prob \leftarrow$ random$(0, 1)$
0: **for** line en list **do**
0: **if** $prob \leq$ frecuencia acumulada **then**
0: $order\_lines \leftarrow$ número de líneas de orden que corresponde a esta línea de la lista
0: **break**
0: **end if**
0: **end for**=0

5.2.3. Desarrollo del modelo conceptual

Esta sección hace referencia a las explicaciones y diagramas de flujo presentados en la sección 5.1 y utiliza las hipótesis, parámetros y algoritmos proporcionados previamente en esta sección. El objetivo de esta sección es explicar cómo se va a desarrollar la simulación, especialmente en aquellas partes en las que se podrían haber adoptado diferentes políticas, comportamientos o reglas.

Pedidos

Si nos remitimos a la Figura 9, hay que tener en cuenta los siguientes puntos para la simulación:

- Antes de ejecutar la simulación, se crean todos los pedidos para todo el horizonte temporal, tal como se representa en los algoritmos 1 y 2. El número de pedidos no es fijo, sino que depende de la tasa de llegada de pedidos.
- Las órdenes se guardan en una estructura de datos que se irá actualizando a medida que avance el tiempo de simulación. Esta estructura de datos será una lista de Python y, en algunos casos, un array, para facilitar los cálculos y la exportación de la salida. Esta estructura de datos tiene los siguientes campos:
  - **ID de pedido:** número único que identifica un pedido.
  - **Hora de llegada del pedido (OAT):** hora en la que el pedido llega al sistema y puede empezar a procesarse.
  - **Estado:** número que identifica el estado del pedido. Este parámetro es clave en el mo-

delo de simulación, como se explicará en profundidad más adelante en este estudio. Los estados que puede adoptar un pedido son:

  - **Estado 0:** Pedido no asignable. Cuando el tiempo de llegada es mayor al tiempo de simulación actual.
  - Estado 1: Pedido asignable. Cuando el tiempo de llegada es igual al tiempo de simulación actual, por lo tanto, el pedido acaba de llegar al sistema.
  - **Estado 2:** Orden asignada a un robot.
  - **Estado 3:** Pedido asignado a un robot en espera de salida. Cuando el robot sigue esperando el tiempo máximo de espera o tener el número máximo de pedidos asignados.
  - **Estado 4:** Pedido procesado. El pedido ya ha salido del sistema.

- **Ubicaciones de picking:** lista de coordenadas que indica la posición del almacén donde se encuentra la ubicación de picking. Dispone de una ubicación de picking por línea de pedido.
- **Número de líneas de la orden.**
- **ID del robot asignado:** mientras la orden no esté en Estado 1 o si está en ese Estado, pero aún no se ha asignado ningún robot, este parámetro toma el valor -1.
- **Tiempo de espera de asignación de la orden (WOAT)**: tiempo que ha esperado la orden antes de ser asignada a un robot.
- **Tiempo de proceso (TP):** tiempo total transcurrido desde que se asigna el pedido a un robot hasta que el pedido sale del sistema, es decir, llega al depósito.

- **Tiempo de procesamiento (OTT)**: tiempo total transcurrido desde que el pedido entra en el sistema hasta que sale de él.
- **Tiempo de movimiento en espera (WMT)**: tiempo que transcurre desde que se asigna el pedido a un robot hasta que este sale.

- En la Figura 14 se ofrece un resumen exhaustivo de todos los estados y su transición.
- En la Figura 15 puede verse una representación visual de los indicadores temporales.
- El pseudocódigo del algoritmo 3 representa la forma en que se crean los pedidos en la estructura de datos explicada anteriormente.

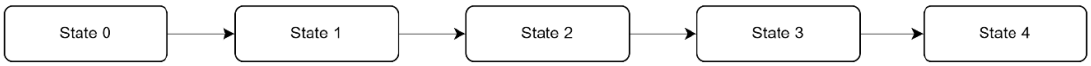

Figura 14: Transición de los estados de los pedidos

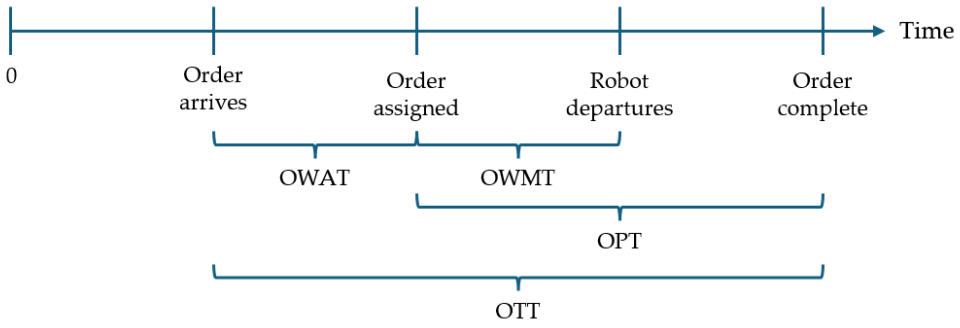

Figura 15: Representacion visual de los indicadores de timepo de los pedidos

Algorithm 3 Generación de órdenes (completa)

0: $time \leftarrow 0$
0: $orders \leftarrow []$
0: $id \leftarrow 0$
0: **while** $time < TH$ **do**
0: $time \leftarrow time +$ integer(random.exponential$(1/$order arrival rate$)$)
0: Ejecutar el algoritmo 2
0: $picking\_locations \leftarrow []$
0: $x \leftarrow$ random.choice$(x_picking_locations)$
0: $y \leftarrow$ random.choice$(y_picking_locations)$
0: $picking\_locations$.append$([x, y])$
0: **end while**
0: $structure \leftarrow [id, 0, time, picking\_locations, order\_lines, -1, 0, 0, 0, 0]$ {Todos los valores iniciales de la estructura explicados}
0: $id \leftarrow id + 1$ =0

Robots

El diagrama de flujo de la Figura 10 se desarrolló conceptualmente, teniendo en cuenta los siguientes puntos:

- Antes de ejecutar la simulación, se crean todos los robots y se ubican en partes iguales en ambos depósitos/estaciones de carga.
- El número de robots depende del escenario.
- Los robots se guardan en una estructura de datos que se actualiza a medida que avanza la simulación. Se almacenará en una lista de Python y en algunos casos en una matriz para facilitar los cálculos y la exportación de la salida. Esta estructura de datos tiene los siguientes campos:
    - **ID de robot:** un número único que identifica a un robot.

- **Estado**: número que identifica el estado del robot. La importancia de este parámetro se explicará más adelante. Los estados que puede adoptar un robot son:
  - Estado 0: el robot se está cargando.
  - Estado 1: el robot está disponible, por tanto, totalmente cargado, con menos de 3 pedidos y ha pasado menos de 30 segundos sin recibir ningún pedido.
  - Estado 2: el robot está esperando la siguiente ubicación de recogida.
  - Estado 3: el robot se desplaza a la siguiente ubicación de recogida.
  - Estado 4: el robot se encuentra en una ubicación de picking a la espera de un recolector que realice la actividad de picking.
  - Estado 5: el robot está recibiendo los artículos del recogedor, porque está realizando la actividad de recogida.
  - Estado 6: el robot espera la posición del depósito.
  - Estado 7: El robot se dirige al depósito.
- Número de pedidos asignados: el número total de pedidos que se han asignado al robot.
- ID de pedido: una lista de todos los ID de pedido gestionados por el robot durante el tiempo del robot.
- Número de recogidas: número total de ubicaciones de recogida visitadas por el robot en el tiempo de simulación.
- Posición real: la coordenada que indica la posición real del robot, en forma de lista de dos números.
- Ubicaciones de picking: lista con todas las ubicaciones de picking de los pedidos que

está procesando el robot y que aún no han sido visitadas, por lo que, una vez visitada una ubicación de picking, se elimina de esta lista.

- Batería restante.
- ID de recogedor: el ID del recogedor asignado actualmente al robot para realizar la actividad de recogida. Si el robot no se encuentra en el Estado 4 o si se encuentra en ese Estado pero aún no se le ha asignado ningún recolector, este parámetro toma el valor -1.
- Tiempo de disponibilidad: indica el tiempo en el futuro o en el presente en el que el robot terminará su actividad actual. En los estados en los que el robot está esperando, el tiempo de disponibilidad es siempre el tiempo presente, mientras que en los estados en los que el robot está realizando una actividad, como desplazarse o cargar, el tiempo de disponibilidad es un tiempo en el futuro, en concreto, el tiempo en el que finaliza dicha actividad.
- Tiempo de espera de pedidos (WOT): el tiempo que el robot pasó esperando pedidos.
- Tiempo de carga (TC): el tiempo que el robot ha estado cargando.
- Tiempo de desplazamiento del robot (RTT): tiempo que tarda el robot en desplazarse a las ubicaciones y depósitos de picking.
- Tiempo de espera del recolector (WPT): el tiempo que el robot pasó esperando a los recolectores en las ubicaciones de recolección.
- Tiempo de recogida del robot (RPT): el tiempo que el robot pasa recibiendo artículos del recogedor que está realizando la actividad de recogida.

  - Número de pedidos completados: número total de pedidos completados, es decir, consignados al depósito en el tiempo de simulación.

- En la Figura 16 se muestra un resumen exhaustivo de todos los estados y sus transiciones.

- El pseudocódigo del Algoritmo 4 representa la forma en que se crean los pedidos en la estructura de datos explicada anteriormente.

Algorithm 4 Generación de robots

$robots \leftarrow []$
for $i \leftarrow 0$ **to** $num\_robots - 1$ **do**
  if $i \leq$ round$(num_robots/2)$ **then**
    $initial\_depot \leftarrow [0, 0]$
  else
    $initial\_depot \leftarrow [width, 0]$
  end if
  $structure \leftarrow [i, 1, 0, [], initial\_depot, [], battery\_size, -1, 0, 0, 0, 0, 0, 0, 0]$ {Todos los valores iniciales de la estructura explicados}
  $robots$.append$(structure)$
end for=0

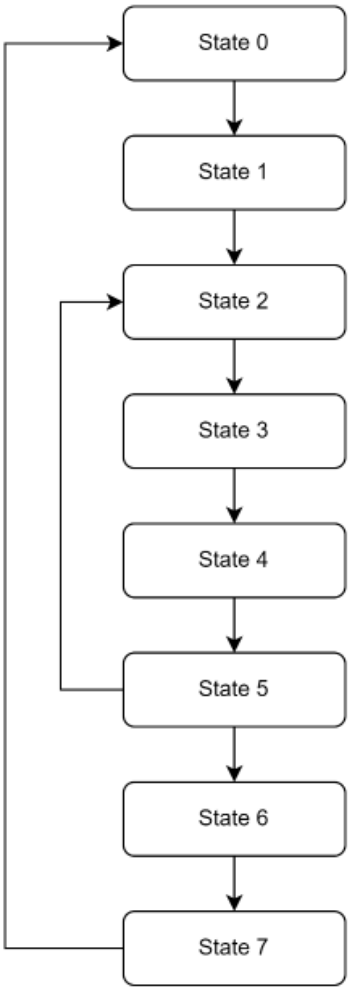

Figura 16: Transición de los estados de los robots

Recogedores

Considere el diagrama de flujo de la Figura 11 y los puntos siguientes para comprender cómo se simulan los recolectores:

- Antes de ejecutar la simulación, se crean todos los recolectores y se colocan en ubicaciones aleatorias del almacén.
- El número de recolectores depende del escenario.

- Los recolectores se guardan en una estructura de datos que se actualiza a medida que avanza la simulación. En cuanto a los robots y los pedidos, el tipo de variable utilizada es una lista de Python y en algunos casos en una matriz para facilitar los cálculos y la exportación de la salida. Esta estructura de datos tiene los siguientes campos:

  - **ID del recolector:** un número único que identifica a cada recolector.
  - **Estado:** un número que identifica el estado actual del recolector. La lógica detrás de esta variable será explicada en las próximas secciones. Los estados que pueden adoptar los recolectores son:
    - **Estado 0:** el recolector está disponible, y está esperando una ubicación de recolección para visitarla, por lo tanto, para acoplarse con el robot en esa ubicación de recolección.
    - **Estado 1:** el recogedor se dirige a la ubicación de recogida de su robot asignado.
    - **Estado 2:** el recogedor está realizando la actividad de recogida.
  - **ID del robot:** el ID del robot que se ha asignado al recolector. Si el recolector está en estado 0, es decir, no se le ha asignado ningún robot, este parámetro toma el número -1.
  - **Posición real:** las coordenadas que indican la posición actual del selector.
  - **Posición siguiente:** las coordenadas de la posición a la que se dirige el recolector, es decir, la posición del robot asignado.
  - **Tiempo de disponibilidad:** indica el tiempo en el futuro o en el presente en el que el recolector terminará su actividad actual. En el Estado 0, en el que el recolector está

esperando, el tiempo de disponibilidad es siempre el tiempo presente, mientras que en los Estados 1 y 2, en los que el recolector está realizando una actividad, el tiempo de disponibilidad es el tiempo en el futuro, en concreto, el tiempo en el que finaliza esa actividad.

- **Tiempo de robot en espera (WRT):** el tiempo que pasa un recolector esperando la asignación de robots.
- **Tiempo de desplazamiento del preparador de pedidos (PTT):** el tiempo que pasa un preparador de pedidos desplazándose por el almacén para llegar a las ubicaciones de picking.
- **Tiempo de preparación de pedidos:** el tiempo que pasa un preparador realizando la actividad de preparación de pedidos.
- **Número de recolecciones.**

- Un resumen exhaustivo de todos los estados del selector y sus transiciones está ilustrado en la Figura 17.
- El pseudocódigo del Algoritmo 5 representa la creación de los recolectores.

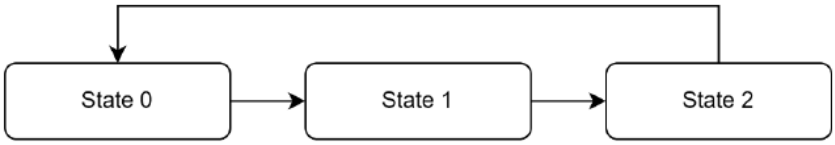

Figura 17: Transición de los estados de los recogedores

Algorithm 5 Generación de pickers

```
pickers ← []
for j ← 0 to num_pickers − 1 do
    initial_position_x ← random.choice(aisles_positions_x)
    initial_position_y ← random.choice(aisles_positions_y)
    initial_position ← [initial_position_x, initial_position_y]
    structure ← [j, 0, −2, initial_position, [], 0, 0, 0, 0, 0]
    robots.append(structure)
    break
end for=0
```

5.2.4. Selección del método de simulación

La simulación de este estudio se clasificará según la taxonomía de modelos de simulación de la Figura 18. El modelo de sistema propuesto se caracteriza por una entrada aleatoria, es decir, los pedidos. Para realizar una simulación más realista del sistema de picking de un almacén, los pedidos deben modelarse teniendo en cuenta la incertidumbre y la aleatoriedad, ya que proceden de los clientes y las empresas no pueden controlarlos; las empresas deben comprender y modelar su comportamiento. De este modo, la variabilidad entre ejecuciones debe tenerse en cuenta en los resultados. Por lo tanto, la simulación es estocástica.

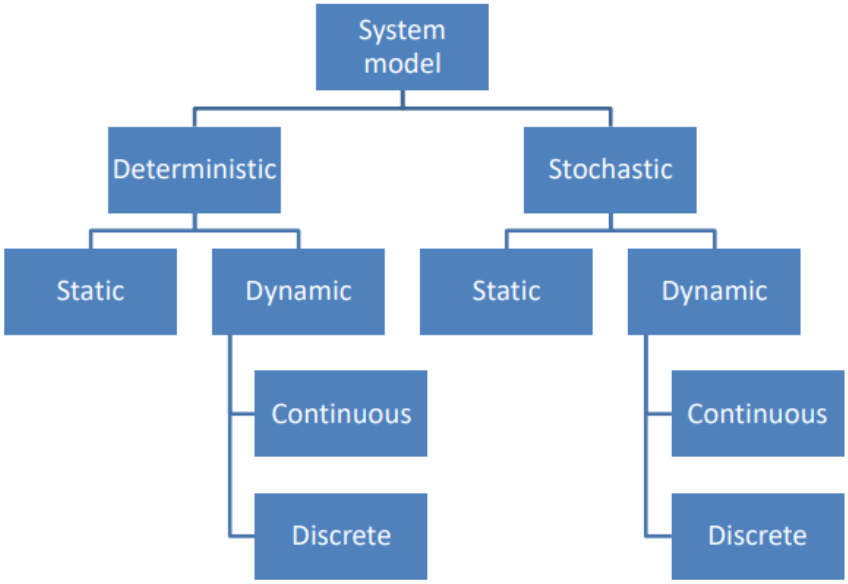

Figura 18: Taxonomía de modelos de simulación

Por otro lado, este sistema está representando un sistema en un horizonte temporal fijo, en particular, un turno, donde existe una evolución temporal. Por lo tanto, se necesita una simulación dinámica. Por último, dado que la simulación es dinámica, es necesario clasificarla entre continua y discreta. En este caso, como se va a explicar más adelante, la simulación se ejecutará en pasos que representan segundos, en consecuencia, como una simulación discreta. Así, esta simulación puede clasificarse como una Simulación Estocástica, Dinámica y Discreta.

Entre las simulaciones discretas se pueden encontrar dos paradigmas principales, la Simulación de Eventos Discretos (DES) y la Simulación Basada en Agentes (ABS). El paradigma DES se centra en sistemas con una serie de eventos discretos, en los que una entidad se desplaza por el sistema y es entendida como un recurso y pasa por varios eventos en los que pueden generarse colas. Se utiliza principalmente para problemas de investigación operativa, como los sistemas de producción y servicios. Por otro lado, el paradigma ABS modela un sistema en el que varios agentes con comportamientos específicos interactúan entre sí y con un entorno.

El paradigma que mejor se ajusta al sistema analizado es el de la Simulación Basada en Agentes. Los componentes de este modelo son:

- Agentes: objetos interactivos y autónomos que se mueven dentro del sistema. Existen varios tipos de agentes que se enumeran a continuación. A continuación, en la Tabla 3 se pueden encontrar todos los agentes del sistema y su clasificación:
    - Discreto: individuo u objeto discreto.
    - Situado: vive en el entorno con el que interactúa con los demás agentes.
    - Dirigidos a objetivos: tienen comportamientos basados en la realización de tareas u objetivos.
    - Autónomos y autodirigidos: pueden funcionar de forma independiente en su entorno y en sus relaciones con otros agentes.
    - Adaptativas: pueden aprender y adaptar sus comportamientos basándose en la experiencia.

Tabla 3: Tipos de agentes en el sistema

Agent	**Type**
Orders	Discrete
Robots	Goal-directed
Pickers	Goal-directed
Depots	Situated
Charging stations	Situated
Picking locations	Situated

- Comportamientos: se refiere a las reglas que siguen los agentes para interactuar entre sí y con su entorno. Estas reglas se explicaron de forma general en la descripción del sistema, pero se explicarán en detalle en las siguientes secciones de este capítulo.

- Entorno: es el espacio donde los agentes interactúan entre sí y con el propio entorno. En este caso, el entorno corresponde al almacén genérico explicado en general en la sección Layout de la sección anterior. En los siguientes apartados se hará una descripción más precisa.

Además, el reloj de la simulación avanza en intervalos de tiempo fijos, que en este representan los segundos de un turno de trabajo. Así mismo, el ABS permite no tener una estructura fija del sistema; los parámetros genéricos del almacén pueden modificarse fácilmente para analizar almacenes de diferentes dimensiones y disposiciones. Otra ventaja de este paradigma es que permite una interacción humano-robot más realista y un modelo más fácil de entender. Las principales desventajas con respecto a un DES son un esfuerzo computacional y de modelación mucho mayor.

5.2.5. Selección de la herramienta de simulación

La simulación se desarrollará en Python, un lenguaje de programación interpretado, orientado a objetos, de alto nivel y semántica dinámica. Es un lenguaje de programación caracterizado por ser de fácil lectura y soporta numerosos módulos y paquetes. En concreto, se va a utilizar la versión Python 3.9.

Existen varios entornos para trabajar con Python, el utilizado para esta tesis es Spyder (Scientific Python Development Environment) versión 4, que es un IDE de código abierto para el lenguaje Python, diseñado para realizar tareas de ciencia de datos de forma más cómoda.

En el código de esta simulación sólo se utilizan cuatro bibliotecas:

- Random: módulo que proporciona funciones relacionadas con la generación aleatoria de datos, la selección aleatoria a partir de una estructura de datos, el ordenamiento aleatorio, entre otras funciones relacionadas.

- NumPy: una librería que permite a los desarrolladores trabajar con arrays, que es una estructura de datos que soporta operaciones de álgebra lineal y matrices. También es útil trabajar con arrays porque muchas otras bibliotecas tienen funciones que reciben como entrada variables en este formato.

- Pandas: una librería de análisis de datos que permite a los desarrolladores trabajar con marcos de datos, fáciles de manipular y de analizar. También permite importar y exportar datos fácilmente en varios tipos de datos, como texto, csv, xlsx, entre otros.

- Matplotlib: una librería para crear visualizaciones estáticas, animadas e interactivas. En este trabajo se utilizó para realizar gráficas de la salida a analizar y para realizar comprobaciones en la fase de desarrollo del script.

5.2.6. Desarrollo del modelo de simulación

Una vez comprendidos todos los elementos relevantes del sistema, considerando todos los supuestos y dinámicas de los diferentes agentes de este sistema de picking humano-robot, en este apartado se va a explicar el desarrollo del modelo de simulación. Se va a realizar una explicación paso a paso del script desarrollado en Python, mediante algoritmos en forma de pseudocódigo; de esta forma no es necesario estar familiarizado con este lenguaje de programación en particular.

El código puede dividirse en seis partes: importación de las librerías, definición de los parámetros, creación de los agentes, creación de funciones útiles que simplifiquen el código, la simulación y la exportación de la salida. Dado que las secciones anteriores ya cubren las librerías, los parámetros y la creación de los agentes, en esta sección sólo se explicarán las funciones y la simulación. La salida se tratará en el próximo capítulo.

Funciones

Creación del layout

Se ha diseñado una función que recibe el ancho y la altura de la bodega para crear automáticamente el diseño. Esto permite que el código sea más flexible, ya que se adapta a las dimensiones específicas que interesen al usuario. En este caso, como ya se ha mencionado, se utilizaron dimensiones fijas: una anchura de 29 metros y una altura de 15 metros. La maqueta se genera en forma de una estructura de datos, en concreto una lista de listas, compuesta de la siguiente manera:

- X_picking_locations: una lista que contiene todas las coordenadas x de los pasillos frente a las ubicaciones de picking, es decir, las posiciones en las que deben estar los robots y los recolectores para realizar la actividad de picking.
- Y_picking_locations: igual que el anterior pero referido a las coordenadas y.
- Blocks_x: una lista que contiene todas las coordenadas x de las ubicaciones físicas de picking, es decir, las posiciones ocupadas por las estanterías. Los recolectores y los robots no pueden desplazarse por estas ubicaciones.
- Bloques_y: igual que el anterior pero referido a las coordenadas y.
- Pasillos_centrales: lista que contiene las coordenadas de los dos pasillos centrales.
- Charging_stations: una lista que contiene las coordenadas de las estaciones de carga.
- Depósitos: lista que contiene las coordenadas de los depósitos.

Cálculo de la distancia

La distancia en este contexto no es la distancia euclidiana, sino la distancia Manhattan, ya que sólo se permiten movimientos perpendiculares. La distancia Manhattan entre dos coordenadas se calcula del siguiente modo:

$$dist((x_1, y_1), (x_2, y_2)) = |(x_1 - x_2)| + |(y_1 - y_2)| \tag{6}$$

Además, hay que considerar la situación en la que un agente se encuentra dentro de un pasillo y debe ir a otro pasillo. En este caso, la simple distancia Manhattan no es suficiente, ya que el agente debe salir del pasillo, ir siguiente, y entrar en el otro pasillo hasta la ubicación de picking.

Para cumplir las diferentes situaciones se desarrolló un algoritmo que las considera todas. Esto se puede ver en el Algoritmo 6. La función recibe como parámetros las coordenadas de la posición inicial y final y da como salida la distancia en metros.

En primer lugar, se calcula la distancia en x, ya que no cambiará entre las distintas situaciones. A continuación, si la coordenada x de ambas posiciones es la misma, se suma la distancia en y. Si ambas posiciones están en el mismo lado del almacén respecto al pasillo central, entonces se suma la distancia entre la posición inicial y el pasillo más cercano, para simular la salida del pasillo, y se suma la distancia entre el pasillo más cercano y la posición final. Por último, si las posiciones se encuentran en distintos lados del almacén respecto al pasillo central, se calcula la distancia Manhattan simple.

Algorithm 6 Cálculo de distancia

$distance \leftarrow \text{abs}(pos1[0] - pos2[0])$
if las coordenadas x de $pos1$ y $pos2$ son iguales **then**
    $distance \leftarrow distance + \text{abs}(pos1[1] - pos2[1])$
else if las coordenadas y de $pos1$ y $pos2$ están en el mismo lado respecto al pasillo central **then**
    $distance \leftarrow distance + \text{abs}(pos1[1] - closest\_aisle\_y) + \text{abs}(pos2[1] - closest\_aisle\_y)$
else if las coordenadas y de $pos1$ y $pos2$ están en lados diferentes respecto al pasillo central **then**
    $distance \leftarrow distance + \text{abs}(pos1[1] - pos2[1])$
end if=0

Determinación de la siguiente posición

Es útil para la simulación determinar la siguiente posición del robot o agente de una forma estándar, de acuerdo con las reglas establecidas en los supuestos. El pseudocódigo que representa la función que realiza esta tarea está en el Algoritmo 7.

Esta función recibe como parámetros la posición real del agente, las posibles ubicaciones de picking, la velocidad del agente y una flag que identifica si el agente es un robot o un picker. Comienza definiendo una lista vacía que contendrá la distancia entre la posición del agente y todas las ubicaciones de recogida; esto se hace mediante un bucle y llamando a la función descrita en el Algoritmo 4. A continuación, se determina la distancia mínima y se toman sus coordenadas. La función devuelve las coordenadas de la siguiente posición, por tanto, la que minimiza la distancia, y el tiempo de viaje, calculado como la distancia mínima dividida por la velocidad del agente.

Algorithm 7 Próxima posición

$distances \leftarrow []$
for $x, y \in picking\_locations$ **do**
    $distance \leftarrow$ calculate_distance$(pos, [x, y])$ {Usando la función explicada en el Algoritmo 6}
    $distances$.append$(distance)$
end for
$distance\_min \leftarrow$ min$(distances)$
$index\_min \leftarrow distances$.index$(distance_min)$
$coord\_min \leftarrow picking\_locations[index\_min]$
$travel\_time \leftarrow$ int$(distance_min/speed)$ =0

Determinación del recolector más cercano

Cuando un robot llega a una ubicación de picking necesita un recogedor que se tiene que asignar. Como se explica en los supuestos, la regla que se sigue para asignar el recogedor al robot es el recogedor disponible más cercano. Se ha desarrollado una función para realizar esta tarea que se representa en el Algoritmo 8.

Esta función recibe como parámetros la posición del robot, la estructura de datos de los pickers, y la velocidad de los pickers. Primero, como parámetros de inicialización, se establece un tiempo de comparación alto, imposible de alcanzar, y se define un id inicial de los pickers. A continuación, el código comienza a recorrer en bucle la estructura de datos de los recolectores. Para cada recolector disponible, se calcula la distancia, llamando a la función representada por el Algoritmo 6, y se compara con el tiempo inicial. Si el tiempo de viaje es menor que el tiempo mínimo actual, entonces el id de este último recolector, con su tiempo de viaje, son registrados como el nuevo candidato. Este bucle continúa hasta que no hay más recolectores disponibles y devuelve el id del recolector más cercano.

Algorithm 8 Picker más cercano

```
0: travel_time ← 100000
0: id_picker ← −1
0: for cada picker en pickers do
0:    if picker está disponible (estado = 0) then
0:       distance ← int(calculate_distance(pos, picker[3])) {Donde picker[3] es la posición del
   picker y esta función está representada en el Algoritmo 6}
0:       candidate ← int(distance/speed)
0:       if candidate < travel_time then
0:          travel_time ← candidate
0:          id_picker ← picker[0] {El id del picker bajo análisis}
0:       end if
0:    end if
0: end for=0
```

Simulación

Básicamente, el código diseñado para esta simulación consiste en un bucle de pasos que simula el tiempo. Así, cada paso representa un segundo hasta completar el tiempo de simulación, es decir, 28800 segundos, lo que equivale a un turno de 8 horas.

Dentro de este bucle, mediante sentencias condicionales, se realizan varias comprobaciones para conocer el estado de los distintos agentes. Si una sentencia condicional es True, es decir, un agente en específico se encuentra en ese estado concreto, entonces se ejecutarán unas instrucciones, que suponen un cambio en sus parámetros almacenados en las estructuras de datos. Entre estos cambios, puede producirse una modificación del estado, por lo que en el siguiente paso ese agente no entrará en esa sentencia condicional, sino en otra.

Dentro del bucle principal hay 3 bucles. Primero se hace una comprobación de los pedidos para activar los que llegan, de forma que sean asignables. Esto se representa en el Algoritmo 9. El segundo, hace un bucle con los robots y comprueba cada posible cambio de estado mediante sentencias condicionales. Dentro de estas estructuras cambian los estados de todos los agentes,

robots, recolectores y pedidos, y se calculan y actualizan los distintos indicadores. En los párrafos siguientes se explicarán todos estos condicionales. Es importante tener en cuenta la Figura 14, la Figura 16, la Figura 17 y las explicaciones de los estados de los distintos agentes para comprender mejor la simulación. Por último, el tercer bucle comprueba los recolectores en estado 0 que no han sido asignados; en este caso, el código simplemente aumenta el tiempo de espera del robot. Esto se ilustra en el Algoritmo 10.

Algorithm 9 Bucle principal para las órdenes

```
0: for cada order en orders do
0:     if order[state] = 0 and order[arrival_time] ≤ time then
0:         order[state] ← 1
0:     end if
0: end for=0
```

Algorithm 10 Bucle principal para los pickers

```
0: for cada picker en pickers do
0:     if picker[state] = 0 and picker[time_of_availability] ≤ time then
0:         picker[waiting_robot_time] += 1
0:     end if
0: end for=0
```

En el bucle de los robots, primero se comprueba si el robot ha terminado de cargarse y, en caso afirmativo, se pone a disposición para recibir pedidos (del estado 0 al 1). Además, se actualiza la hora de disponibilidad a la hora actual. Si no está totalmente cargado, se incrementa el tiempo de carga. Detalles en el Algoritmo 11.

Algorithm 11 Verificar robot en estado 0

```
0: if robot[state] = 0 and robot[time_of_availability] ≤ time then
0:     robot[state] ← 1
0: else
0:     robot[charging_time] += 1
0: end if=0
```

A continuación, se pregunta si el robot está disponible para recibir órdenes y tiene un

número de órdenes asignadas inferior al límite. Si es así, se realiza una verificación de los pedidos que han llegado pero no han sido asignados y, si es posible, se asigna un pedido al robot. En ese caso, se rellenan todos los parámetros relacionados, como las ubicaciones de picking, el número de pedidos asignados al robot y muchos otros. Además, el estado de la orden cambiaría de 1 a 2, y algunos de sus parámetros también cambian. Los indicadores de tiempo de la orden y del robot se actualizan en consecuencia. Si no, ninguna orden asignada, se actualizan los indicadores de tiempo relativos a las órdenes en espera (para el robot) y al movimiento en espera (para las órdenes ya asignadas, si las hay). También se comprueba, tras verificar la posibilidad de asignar una orden al robot, si ha transcurrido el tiempo máximo para recibir órdenes. Si es así, o si el robot alcanza el número máximo de órdenes, lo que ocurre primero, el estado del robot cambia de 1 a 2. Todos los detalles están en el Algoritmo 12.

Algorithm 12 Verificar robot en estado 1

```
0: if robot[state] = 1 and robot[_orders_assigned] < 3 then
0:    for cada order en orders do
0:       if order[state] = 2 and order[robot_id] = robot[id] then
0:          order[waiting_movement_time] += 1
0:       else if order[state] = 1 then
0:          order[state] ← 2
0:          order[robot_id] ← robot[id]
0:          order[waiting_assignation_time] ← time − int(order[arrival_time])
0:          robot[_orders_assigned] += 1
0:          robot[orders_ids].append(order[id])
0:          for cada order_line en order[picking_locations] do
0:             if order_line no está en robot[picking_locations] then
0:                robot[picking_locations].append(order_line)
0:             end if
0:          end for
0:       end if
0:       if robot[_orders_assigned] = 3 or robot[current_waiting_orders_time] ≥ 30 then
0:          robot[state] ← 2
0:          robot[time_of_availability] ← time
0:          robot[current_waiting_orders_time] ← 0
0:          break
0:       end if
0:    end for
0:    if robot[state] ≠ 2 then
0:       robot[waiting_order_time] += 1
0:       robot[current_waiting_order_time] += 1
0:    end if
0: end if=0
```

Antes de gestionar los robots en estado 2, es decir, los robots que tienen que determinar su próxima posición, se realiza una comprobación previa para considerar todos los posibles robots en estado 2, es decir, los robots que acaban de terminar de realizar el picking. Si un robot se encuentra en el estado 5, pero el tiempo actual es igual a su tiempo de disponibilidad, significa que ha terminado de recoger, por lo que su estado cambia al estado 2 si todavía hay más ubicaciones de recogida que visitar o al estado 6 si era la última ubicación de recogida. El estado del recolector cambia de 2 a 0, disponible para ser asignado a otro robot. El vínculo entre el recolector y el robot

(y viceversa) se rompe. Si el robot no ha terminado el picking, el tiempo de picking aumenta. Esto se ilustra en el Algoritmo 13.

A continuación, se pregunta si el robot se encuentra en el estado 2, es decir, listo para empezar a viajar. En caso afirmativo, se determina la siguiente posición mediante la función representada en el Algoritmo 5 y el estado cambia al estado 3, viajando. Aquí se determina el tiempo de disponibilidad, el consumo de batería y se elimina la siguiente posición de la lista de posiciones de picking. Esto puede verse en el Algoritmo 14.

Ahora, si el robot está en el estado 3, es decir, viajando, sólo se comprueba si ha llegado la hora de disponibilidad o no. Si es así, el estado cambia a 4, a la espera de un recolector; de lo contrario, el tiempo de viaje se incrementa. El pseudocódigo disponible en el Algoritmo 15.

Posteriormente, se comprueba si el robot se encuentra en el estado 4. En este caso, se asigna el recolector más cercano (utilizando el Algoritmo 6). La salida de esa función puede ser -1 si no hay recolectores disponibles en ese instante. Si es así, se incrementa el tiempo de espera del recolector. En caso contrario, el estado del recogedor seleccionado se cambia de 0 a 1 (de disponible a viajando) y los parámetros relacionados del recogedor y del robot se ajustan en consecuencia. Por ejemplo, el tiempo de disponibilidad de ambos agentes se calcula como la hora actual más el tiempo de desplazamiento del recogedor, se actualiza la posición del recogedor, entre otros cambios. Esta parte se ilustra en el Algoritmo 16.

Algorithm 13 Verificar robot en estado 5

if $robot[state] = 5$ **and** $robot[time\_of\_availability] \leq time$ **then**
  $robot[\_picking\_locations\_visited] \mathrel{+}= 1$
  $pickers[robot[picker\_id]][state] \leftarrow 0$
  $pickers[robot[picker\_id]][robot\_id] \leftarrow -1$
  $pickers[robot[picker\_id]][\_pickings] \mathrel{+}= 1$
  $robot[picker\_id] \leftarrow -1$
  if len($robot[picking\_locations]$) >0 **then**
    $robot[state] \leftarrow 2$
  else
    $robot[state] \leftarrow 6$
  end if
else
  $robot[picking\_time] \mathrel{+}= 1$
  $pickers[robot[picker\_id]] \mathrel{+}= 1$
end if=0

Algorithm 14 Verificar robot en estado 2

if $robot[state] = 2$ **then**
  $robot[state] \leftarrow 3$
  $coord\_min, travel\_time \leftarrow$ next_position$(robot[position], robot[picking_locations], speed,$ True$)$
  $robot[position] \leftarrow coord\_min$
  $robot[picking\_locations]$.remove$(coord_min)$
  $robot[battery] \mathrel{-}= battery\_consumption \times travel\_time$
  $robot[time\_of\_availability] \leftarrow time +$ int$(travel_time)$
end if=0

Algorithm 15 Verificar robot en estado 3

if $robot[state] = 3$ **and** $robot[time\_of\_availability] \leq time$ **then**
  $robot[state] \leftarrow 4$
else
  $robot[travel\_time] \mathrel{+}= 1$
end if=0

Algorithm 16 Verificar robot en estado 4 (asignar pickers)

```
0: if robot[state] = 4 and robot[picker_id] = −1 then
0:     id_picker ← nearest_picker(robot[position], pickers, speed_picker)
0:     if id_picker ≠ −1 then
0:         pickers[id_picker][state] ← 1
0:         pickers[id_picker][robot_id] ← robot[id]
0:         coord_min, travel_time ← next_position
0:         (pickers[id_picker][position], robot[position], speed_picker, False)
0:         pickers[picker_id][position] ← coord_min
0:         pickers[picker_id][time_of_availability] ← time + int(travel_time)
0:         robot[picker_id] ← id_picker
0:         robot[time_of_availability] ← time + int(travel_time)
0:     else
0:         robot[waiting_picker_time] += 1
0:     end if
0: end if=0
```

Después se comprueba de nuevo si el robot está en el estado 4, pero sólo si ya se ha asignado un recolector. De esta forma es más fácil comprobar si ha llegado o no. Si es así, el estado del robot cambia de 4 a 5 y el estado del recogedor cambia de 1 a 2. Se actualiza el tiempo de viaje del recogedor y se calcula la disponibilidad de tiempo de ambos agentes teniendo en cuenta el tiempo de recogida. Para ello se utiliza el Algoritmo 17.

Algorithm 17 Verificar robot en estado 4 (realizar picking)

```
0: if robot[state] = 4 and robot[picker_id] = −1 then
0:     if int(pickers[robot[picker_id]][time_of_availability]) ≤ int(time) then
0:         pickers[robot[picker_id]][state] ← 2
0:         pickers[robot[picker_id]][picking_time] += 1
0:         picking_time ← random.randint(2, 10)
0:         pickers[robot[picker_id]][time_of_availability] ← time + picking_time
0:         robot[state] ← 5
0:         robot[picking_time] += 1
0:         robot[time_of_availability] ← time + picking_time
0:     else
0:         robot[waiting_picker_time] += 1
0:         pickers[robot[picker_id]][travel_time] += 1
0:     end if
0: end if=0
```

En cuanto al caso del estado 2, ahora se comprueba si el robot está en el estado 6, que es el mismo, con la única diferencia de que no hay más ubicaciones de picking, por lo que se debe decidir el depósito. Esto se hace de la misma manera, se elige la siguiente posición más cercana. Una vez determinado, el estado cambia al estado 7, viajando al depósito. El número de pedidos en proceso vuelve a ser 0, se disminuye el depósito en función de lo que vaya a viajar y se actualiza el tiempo de disponibilidad. El pseudocódigo del Algoritmo 18.

Algorithm 18 Verificar robot en estado 6

0: **if** $robot[state] = 6$ **then**
0: $robot[state] \leftarrow 7$
0: $robot[\_orders\_assigned] \leftarrow 0$
0: $coord\_min, travel\_min \leftarrow$ next_position$(robot[position], depots, speed,$ False) {Función explicada en el Algoritmo 7}
0: $robot[position] \leftarrow coord\_min$
0: $robot[battery]$ −= int$(battery_consumption * travel_time)$
0: $robot[time\_of\_availability] \leftarrow time +$ int$(travel_time)$
0: **end if**=0

Por último, se comprueba el estado 7, es decir, si el robot ha llegado al depósito o no. En caso afirmativo, el estado cambia a 0, carga. Los pedidos gestionados por ese robot actualizan sus parámetros: el estado cambia a 3 (procesado) y se calculan los tiempos de proceso y rendimiento. El robot recupera su batería, y su tiempo de disponibilidad se actualiza al tiempo en el que terminará de cargarse. Si el robot aún no ha llegado, simplemente se actualiza su tiempo de recorrido. Esto se detalla en el Algoritmo 19.

El bucle principal que incluye todos los bucles que acaban de explicar, termina aumentando el tiempo en uno, y ordenando los robots por su tiempo de disponibilidad, a partir del que estará disponible primero. El código principal, que incluye todas las demás partes, se representa en el Algoritmo **??**.

Algorithm 19 Verificar robot en estado 7

if $robot[state] = 7$ **and** $robot[time\_of\_availability] \leq time$ **then**
  for each $order$ in $robot[orders\_ids]$ **do**
    $orders[order][state] \leftarrow 3$
    $orders[order][process\_time] \leftarrow time - orders[order][arrival\_time] - orders[order][assignation\_time]$
    $orders[order][throughput\_time] \leftarrow time - orders[order][arrival\_time]$
    $robot[\_orders\_processed] \mathrel{+}= 1$
  end for
  $robot[orders\_id] \leftarrow []$
  $robot[state] \leftarrow 0$
  $charging\_time \leftarrow \text{int}\left(\frac{battery\_size - robot[battery]}{power\_charging\_station}\right) \times 60 \times 60$
  $robot[battery] \leftarrow battery\_size$
  $robot[time\_of\_availability] \leftarrow time + charging\_time$
  $robot[charging\_time] \mathrel{+}= 1$
else
  $robot[travel\_time] \mathrel{+}= 1$
end if=0

Algorithm 20 Simulación completa

$time \leftarrow 0$
while $time < simulation\_time$ **do**
  Algorithm 9
  for each $robot$ in $robots$ **do**
    Algorithm 11
    Algorithm 12
    Algorithm 13
    Algorithm 14
    Algorithm 15
    Algorithm 16
    Algorithm 17
    Algorithm 18
    Algorithm 19
  end for
  Algorithm 10
  $time \mathrel{+}= 1$
  $robots.sort(by : time\_of\_availability)$
end while=0

5.2.7. Verificación y Validación del Modelo

Siguiendo el marco metodológico para proyectos de simulación (Figura 12), es fundamental realizar la verificación y validación del modelo antes de proceder con la experimentación.

Verificación del Modelo

La verificación, el proceso de asegurar que el modelo de simulación implementa correctamente el modelo conceptual, se realizó de forma continua durante el desarrollo del código en Python. Se emplearon técnicas de revisión de código, depuración y pruebas con valores extremos para confirmar que la lógica programada correspondía con los diagramas de flujo y los supuestos establecidos.

Validación del Modelo

La validación busca confirmar que el modelo de simulación es una representación precisa del sistema que se pretende estudiar. Dado que se modela un sistema genérico, la validación no puede realizarse contra datos de un sistema real específico. En su lugar, se utilizó una combinación de técnicas recomendadas por la literatura, similar a la aproximación de otros trabajos en el área.

- Validez Aparente (Face Validity): El modelo conceptual y sus supuestos fueron revisados por expertos académicos (profesora guía) para asegurar su coherencia y lógica. La configuración del sistema de enjambre es consistente con las definiciones operativas descritas en la literatura reciente, donde los robots y los recolectores operan de forma desacoplada y se encuentran en la ubicación del picking.

- Comparación con Modelos Publicados: Esta es la técnica de validación principal para este

trabajo. Se compararon tanto los parámetros de entrada como el comportamiento de los resultados del modelo con los de estudios validados recientemente.

- Validación de Parámetros: Para asegurar el realismo del modelo, se utilizaron parámetros alineados con la literatura. Notablemente, el perfil de los pedidos y la distribución de las líneas de pedido 2 son obtenidos de (3), quienes los obtuvieron de un set de datos real de una gran empresa de e-fulfillment. De igual forma, los parámetros operativos como las velocidades de los agentes, las especificaciones técnicas de los robots y la configuración del almacén son consistentes con los utilizados en estudios de simulación de Guerra (6).
- Validación del Comportamiento (Resultados): Se verificó que las tendencias de los resultados del modelo fueran coherentes con los hallazgos de la literatura. No se busca replicar los valores exactos, sino confirmar que el modelo reacciona de forma lógica y esperada:
    - Impacto del Tamaño del Sistema: En los .Escenarios 2", el Tiempo Medio de Procesamiento de Pedidos (AOTT) aumenta al incrementar la tasa de llegada de pedidos (Factor OAR). Esta tendencia es la misma que la observada en el análisis de sensibilidad de Guerra (6), quien también concluye que un aumento en la tasa de llegada impacta negativamente el rendimiento.
    - Impacto de los Recursos: En los .Escenarios 1", aumentar el número de robots y recolectores mejora el rendimiento del sistema (reduce el AOTT). Este es un principio fundamental que también se confirma en las conclusiones de Ghorashi et al. (5), donde más recursos (pickers, AMRs) aumentan la capacidad del sistema.

  - Utilización de los Agentes: Los resultados muestran que a mayor número de robots para una carga de trabajo fija, la utilización media de los robots (ARU) disminuye. Este comportamiento es lógico y esperado, ya que el mismo trabajo se reparte entre más agentes, resultando en más tiempo de inactividad por robot.

- Análisis de Sensibilidad y Pruebas con Valores Extremos: Siguiendo la práctica estándar de validación de simulaciones, se realizaron pruebas con valores extremos. Por ejemplo, en el escenario con 4 robots y una tasa de llegada de pedidos con factor 3, el sistema colapsó y los tiempos de espera de pedidos se dispararon a casi la mitad del turno de simulación, un resultado esperado que valida la capacidad del modelo para reflejar condiciones de sobrecarga.

La combinación de estas técnicas, y en especial la fuerte alineación de los parámetros y tendencias del modelo con los de investigaciones validadas y publicadas, proporciona un alto grado de confianza en que el modelo de simulación es una herramienta creíble y robusta para responder a las preguntas de investigación de esta tesis.

6. Resultados

En este capítulo, en primer lugar, se explicará la forma en que se determinó el número de ejecuciones, con el fin de disponer de resultados representativos. A continuación, se presentarán los distintos escenarios objeto de análisis. Por último, se describirán los principales resultados de los indicadores expuestos para cada análisis.

6.1. Número de corridas

Para determinar el número de ejecuciones se sigue paso a paso el algoritmo que se describe a continuación. Para ello, es necesario elegir un indicador para analizar; en este caso, la medida seleccionada es el Tiempo Medio de Procesamiento de Pedidos (AOTT). El algoritmo consiste en:

1. Fijar el ancho medio deseado c*, en este caso es $0,01 * AOTT_{run}$.

2. Elegir un nivel de confianza. Un $\alpha = 0,05$, es decir, un nivel de confianza del 95 % elegido para esta simulación.

3. Establecer un valor inicial para el número de ejecuciones N. Se ha elegido $N = 10$ ejecuciones.

4. Calcular el percentil de la distribución t con $N - 1$ grados de libertad y nivel de confianza de $1 - \alpha$.

5. Ejecutar N réplicas y calcular la varianza del $AOTT_{run}$: $\hat{\sigma}(N)$.

6. Calcular $c(N) = percentile \cdot \frac{\hat{\sigma}(N)^{1/2}}{N}$.

7. Si $c(N) \leq c^*$, entonces N es el número de ejecuciones.

8. En caso contrario, $N = N + 5$ e ir al paso 4.

Tras realizar este procedimiento iterativo en Python para un caso específico, en concreto 4 robots, 10 recogedores y una tasa de llegada de pedidos de 160 / 60 / 60 pedidos por segundo, se representó gráficamente el Tiempo medio de producción de pedidos entre ejecuciones, que puede verse en la Figura 19. El número necesario de ejecuciones determinado fue de 45 ejecuciones; con esta cantidad de ejecuciones, los resultados pueden considerarse significativos dentro de un intervalo de confianza del 95 %.

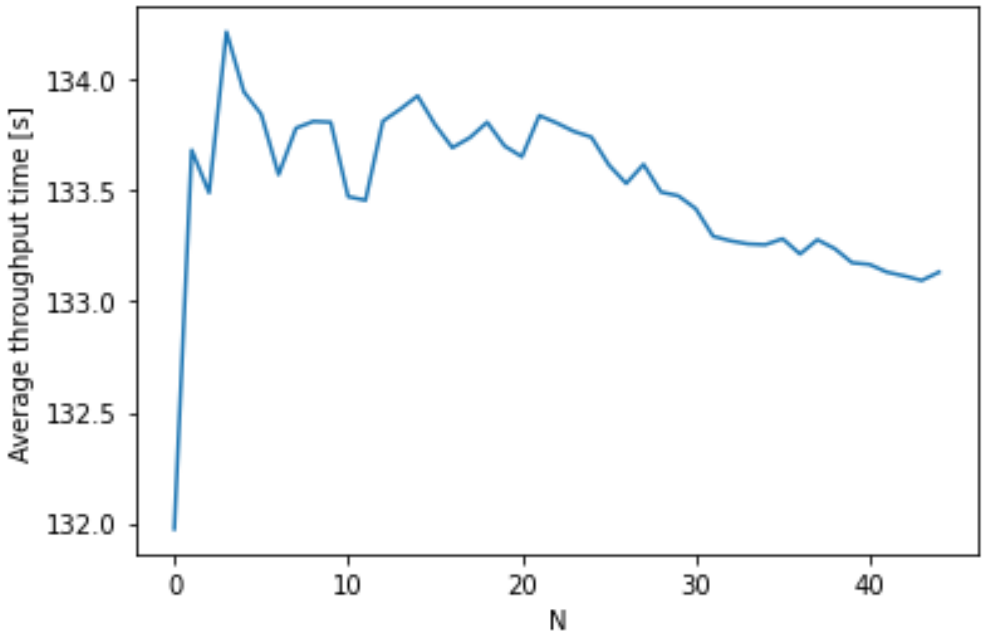

Figura 19: Número de corridas

6.2. Escenarios

La simulación se realizó considerando 2 conjuntos de escenarios. El primer conjunto de escenarios (Escenarios 1) considera una tasa de llegada de pedidos (OAR) fija y varía el número de robots (NR) y el número de recolectores (NP), sumando un total de 15 escenarios. Cada escenario

se ejecutó 45 veces para obtener resultados significativos. Los resultados corresponden a la media entre ejecuciones de cada agente. Los escenarios 1 se diseñaron para responder a las RQ1 y RQ2. Este conjunto de escenarios considera los siguientes parámetros:

- OAR = 160 / 60 / 60 órdenes por segundo.
- NP= [2, 3, 4].
- NR= [6, 7, 8, 9, 10].

El segundo conjunto de escenarios (Escenarios 2) varía tanto la tasa de llegada de pedidos como el número de robots y recolectores. El OAR cambia a través de un factor (Factor OAR). En este caso, lo que se fija es la relación entre el número de robots y el número de recolectores. De forma análoga, se propusieron los Escenarios 2 para responder a los RQ3 y RQ4. Este conjunto de escenarios suma un total de 12 escenarios como sigue:

- NR / NP = 2.

(NR, NP) = [(4, 2), (6, 3), (8, 4), (10, 5)].

- Factor OAR= [1, 2, 3].
- OAR= Factor OAR * 80 / 60 / 60 órdenes por segundo.

El resultado se dividirá en los dos conjuntos de escenarios, y en cada escenario, se dividirá en los indicadores relacionados con cada agente, el Tiempo Medio de Procesamiento del Pedido, la Utilización Media del Recogedor y la Utilización Media del Robot. Se proporcionarán tablas, gráficos y comentarios para una mejor comprensión de los resultados.

6.3. Escenarios 1

6.3.1. Tiempo Medio de Procesamiento de Pedidos

El Tiempo Medio de Tramitación de los Pedidos (AOTT) puede entenderse, tal y como se ha definido anteriormente, como el tiempo medio empleado por los pedidos desde que llegan al sistema hasta que se depositan en el almacén. También es interesante entender cómo cambian los componentes de este indicador entre los distintos escenarios. En primer lugar, se expondrán los resultados de cada componente y, a continuación, se explicará el resultado de la AOTT.

Los resultados del tiempo medio de asignación de pedidos en espera (AOWAT) se representan en el gráfico de la Figura 20. Manteniendo fijo el número de recolectores, el AOWAT siempre disminuye a medida que aumenta el número de robots. Considerando como referencia los escenarios en los que el número de robots es 6, al introducir un robot más en el sistema, el AOWAT disminuye entre un 46 % y un 70 % en función del número de preparadores. Además, para todos los números de recolectores, la adición de un robot más mejora el AOWAT a un ritmo decreciente: la adición del octavo robot disminuye el AOWAT con respecto a la línea de base en un intervalo del 21 % al 33 %, la adición del noveno la reduce en un intervalo del 5 % al 9 %, y la adición del décimo en un intervalo del 3 % al 7 %. También es relevante que las reducciones más significativas en el AOWAT se obtienen cuando el número de recolectores es menor.

Aunque puede resultar evidente que manteniendo fijo el número de robots y variando únicamente el número de recogedores, se observa que la AOWAT también disminuye, es interesante ver en qué medida influye la adición de recogedores. Considerando como referencia los escenarios en los que el número de recogedores es 2, la adición del tercer recogedor reduce el AOWAT en un rango del 76 % y el 95 %, mientras que la adición del cuarto recogedor al sistema lo reduce en un

rango del 2 % y el 10 %.

El resultado del Tiempo medio de espera de movimiento (AWMT) está disponible en la Figura 21. Como en la AOWAT, al mantener fijo el número de uno de los agentes y aumentar el del otro, la AWMT disminuye. No obstante, para este indicador el índice no es tan elevado como en el caso de la AOWAT. Manteniendo fijo el número de recolectores, la adición del séptimo robot disminuye el AWMT en un rango del 32 % y el 47 %, el octavo lo reduce en un rango del 24 % y el 35 %, el noveno en un rango del 11 % y el 16 %, y el décimo en un rango del 6 % y el 11 %.

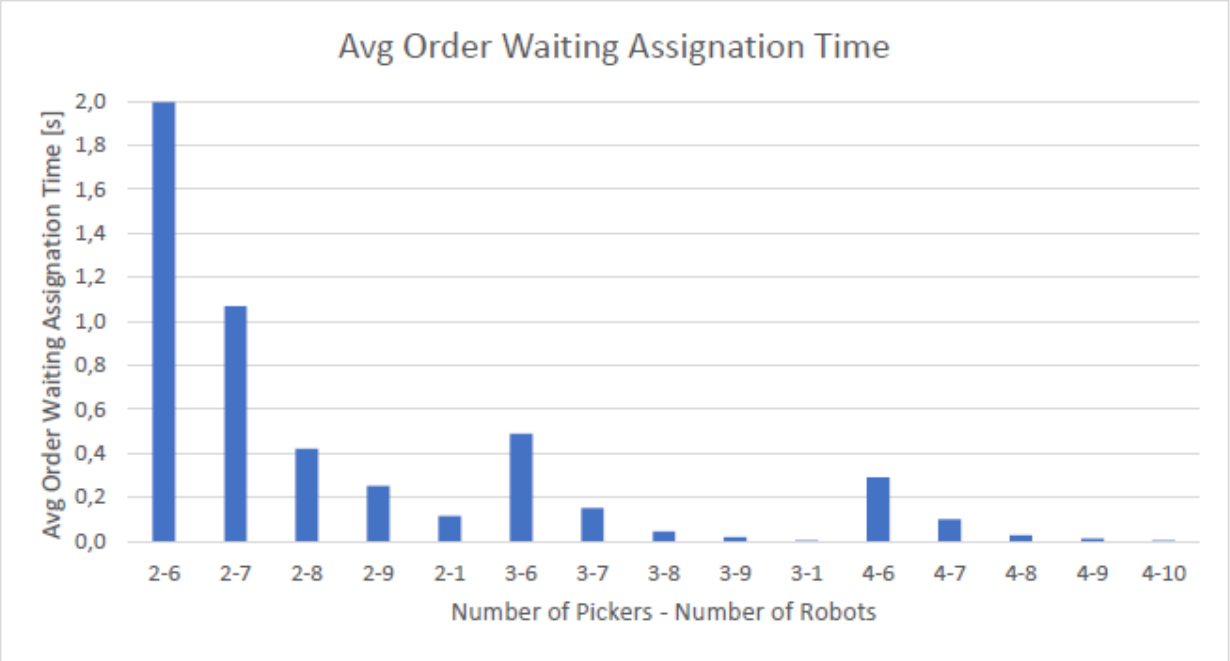

Figura 20: Tiempo medio de espera de asignación de un pedido

Ahora, manteniendo fijo el número de robots y aumentando el número de recogedores, la adición del tercer recogedor disminuye el AWMT en un rango del 26 % al 65 %, y el cuarto lo reduce en un rango del 7 % al 8 %.

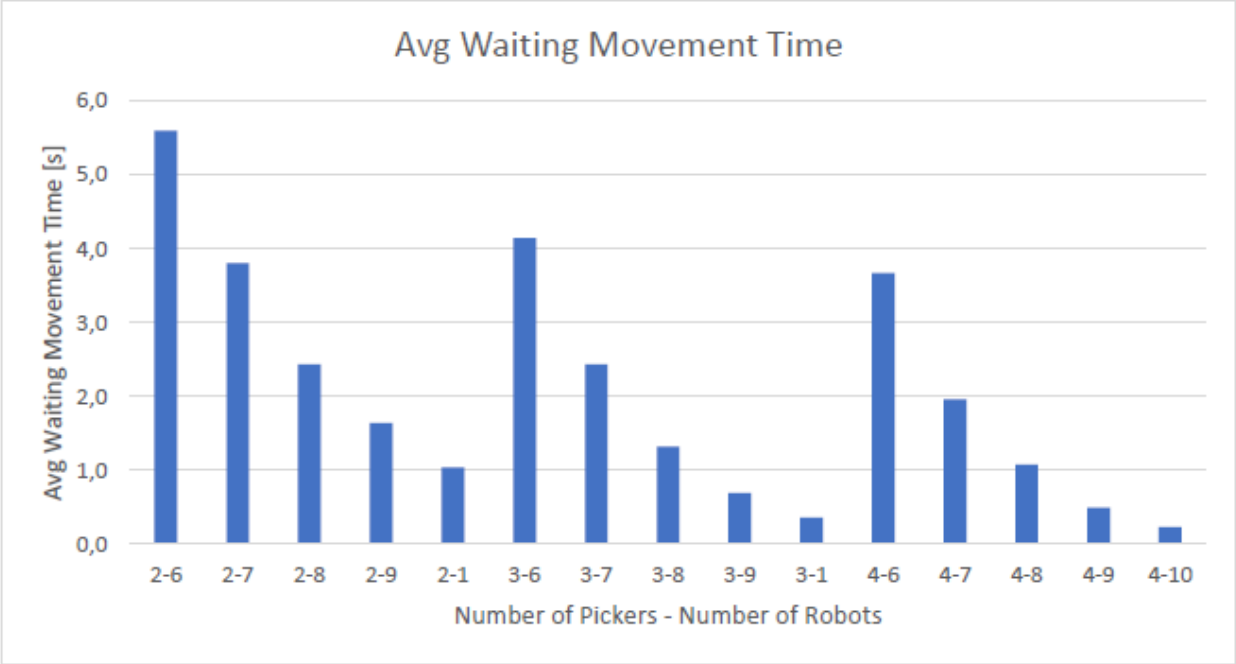

Figura 21: Tiempo medio de espera de movimiento de un pedido

El tiempo medio de proceso (APT), el tiempo medio que emplean los pedidos desde que se asignan, se representa en el gráfico de la Figura 22. El mínimo entre todos los escenarios es de 58 segundos, mientras que el máximo es de 111 segundos, casi el doble del valor mínimo, por lo que el rendimiento está muy influido por el número de agentes. A primera vista, puede observarse que aumentar el número de un agente mientras se mantiene fijo el otro disminuye el APT.

En primer lugar, fijando el número de recolectores mientras se aumenta el número de robots, los resultados son los siguientes: añadir un robot más disminuye el APT en un rango del 6 % al 9 %, añadir dos robots más lo reduce en un rango del 5 % al 7 %, añadir el noveno lo reduce en un rango del 2 % al 3 %, y añadir el décimo robot lo disminuye en un rango del 1 % al 2 %.

Ahora bien, fijando el número de robots y aumentando el número de recogedores, los resultados obtenidos son los siguientes: añadir un recogedor más al sistema mejora el rendimiento en un rango del 28 % y el 30 %, y añadir el cuarto recogedor lo hace en un rango del 7 % y el 8 %. La

velocidad a la que disminuye el APT no cambia significativamente cuando se mantiene un número fijo de robots y se aumenta el número de recogedores. Además, los resultados de los escenarios en los que el número de recogedores es 3 ó 4 son inferiores a los escenarios en los que el número de recogedores es 2, pero no son significativamente diferentes entre sí.

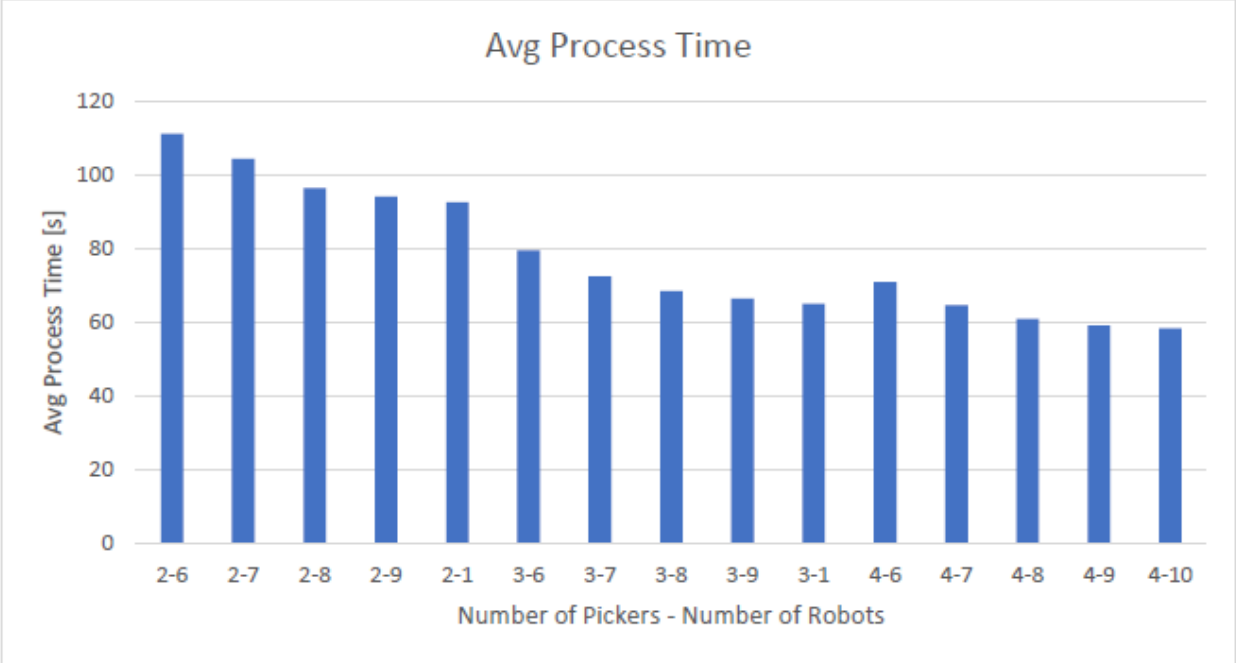

Figura 22: Tiempo medio de tiempo de proceso de un pedido

Los resultados del AOTT se ilustran en la Figura 23. Los resultados de este indicador siguen las mismas pautas que los del APT, ya que el AOTT es igual al APT más el AOWAT, y el APT toma valores mucho más altos que los del AOWAT; mientras que el APT tiene valores en el intervalo de 58 segundos y 111 segundos, el AOWAT está entre 0 y 2 segundos. Así, el mínimo de la AOTT es de 59 segundos, y su máximo es de 113 segundos. Como en el caso del APT, los escenarios en los que los AOTT son más bajos son cuando el número de recolectores es 4, pero estos rendimientos no son significativamente mejores que los obtenidos con 3 recolectores. Ambos rendimientos son significativamente mejores que los obtenidos con sólo 2 recolectores.

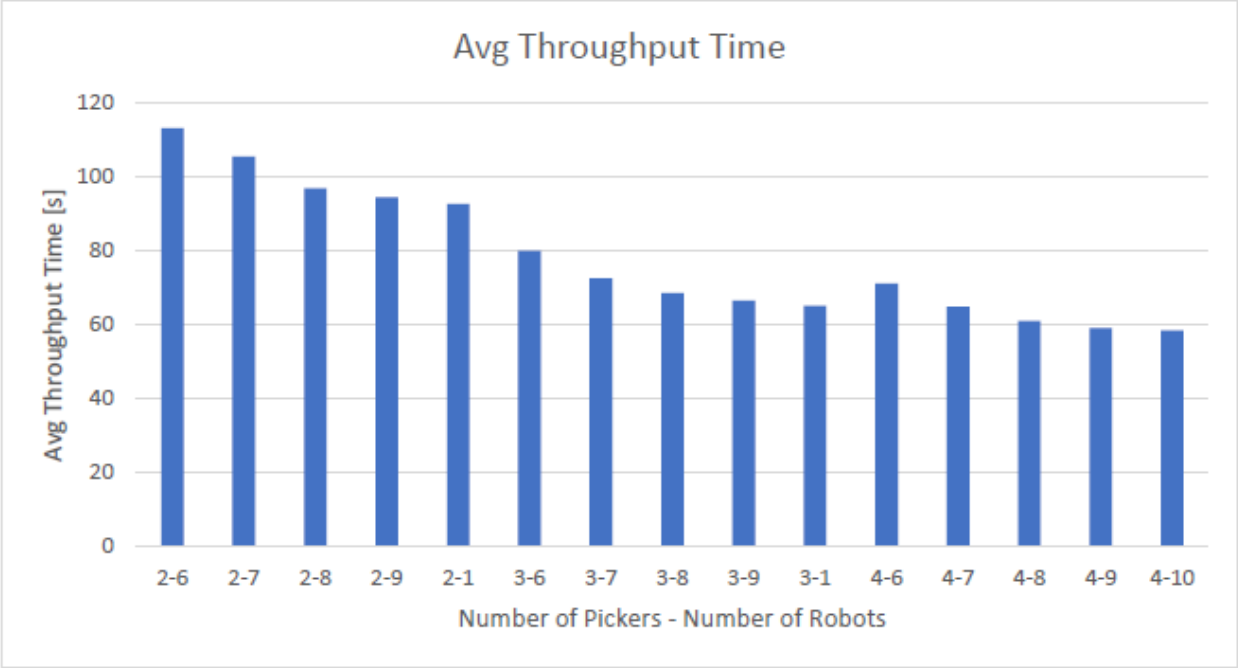

Figura 23: Tiempo medio de procesamiento de un pedido

6.3.2. Utilización media de los recogedores

La Utilización Media del Recogedor es un indicador que permite conocer el tiempo en el que este recurso está siendo efectivamente utilizado. En este sistema, como se ha presentado anteriormente, los recolectores pueden estar esperando a ser asignados a un robot, viajando o haciendo picking.

En la Figura 24 se representa el porcentaje de tiempo medio que los recolectores dedican a cada actividad. El gráfico muestra que, para un número fijo de recolectores, el tiempo medio de recogida de los recolectores (APPT) no cambia significativamente. De hecho, cuando el número de preparadores se fija en 4 y el número de robots es de 6, el APPT es del 15,0 %; mientras que al aumentar los robots a 10, el APPT sólo aumenta en un 0,3 %, hasta el 15,3 %. La situación es similar cuando el número de recogedores es 3 (del 10,0 % al 10,1 %) y cuando el número de

recogedores es 2 (en el intervalo del 7,5 % y el 7,6 %. Esto se debe a que la cantidad de pedidos a recoger depende principalmente de la Tasa de Llegada de Pedidos, que es fija para los Escenarios 1. De esta forma, hay una cantidad constante de pedidos en cada escenario a repartir entre un número variable de preparadores, por lo que el número de robots no es relevante para este indicador.

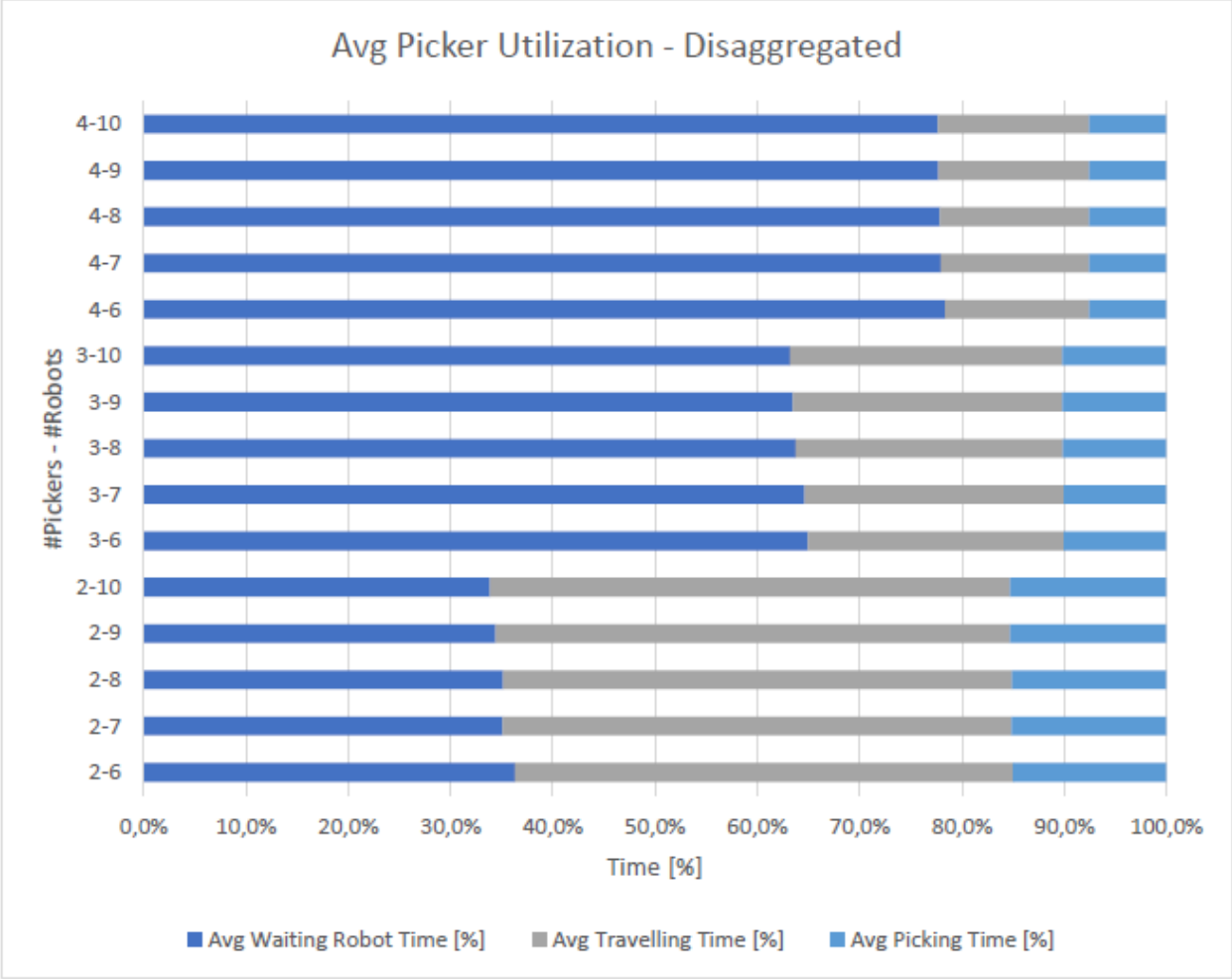

Figura 24: Utilización media de los recogedores - Desagregada

Sin embargo, lo más interesante es cómo cambia el APPT cuando varía el número de recolectores. Considerando la media de los valores para un número fijo de recolectores y distinto número de robots, se construyó el gráfico de la Figura 25. En este gráfico se puede observar que tener más recogedores implica menos tiempo por recogedor de media realizando la actividad de

picking, ya que, como se ha explicado, un número constante de pedidos en el tiempo de simulación se divide en más recogedores.

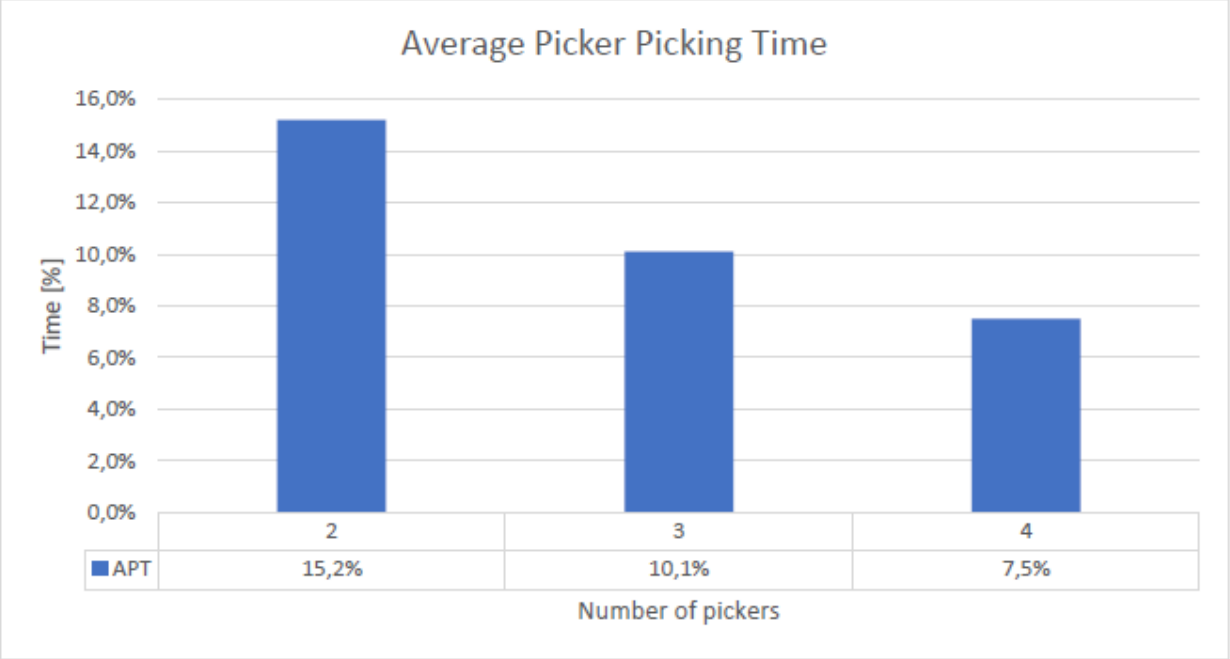

Figura 25: Tiempo medio de picking de los recogedores

El mismo análisis es válido para el Tiempo Medio de Recorrido del Recogedor (APTT), es decir, el aumento del número de robots no afecta significativamente a este indicador, ya que el número de pedidos entre escenarios es constante, por lo el número de ubicaciones de recogida será similar. Así, la distancia a recorrer por escenario no variará significativamente y habrá que dividirla entre el número de recolectores. Por lo tanto, el tiempo empleado de media por los recolectores será el mismo independientemente del número de robots que haya. En cuanto al APTT, es más interesante ver cómo varía al aumentar el número de recolectores. La figura 26 muestra los promedios para un número fijo de recolectores. Se puede observar también para este indicador que al aumentar el número de recolectores, el APTT disminuye, ya que más recolectores deben recorrer, en total, una distancia casi constante. Es relevante mencionar que en el caso de 2 recolectores, en

promedio la mitad del tiempo del turno están viajando.

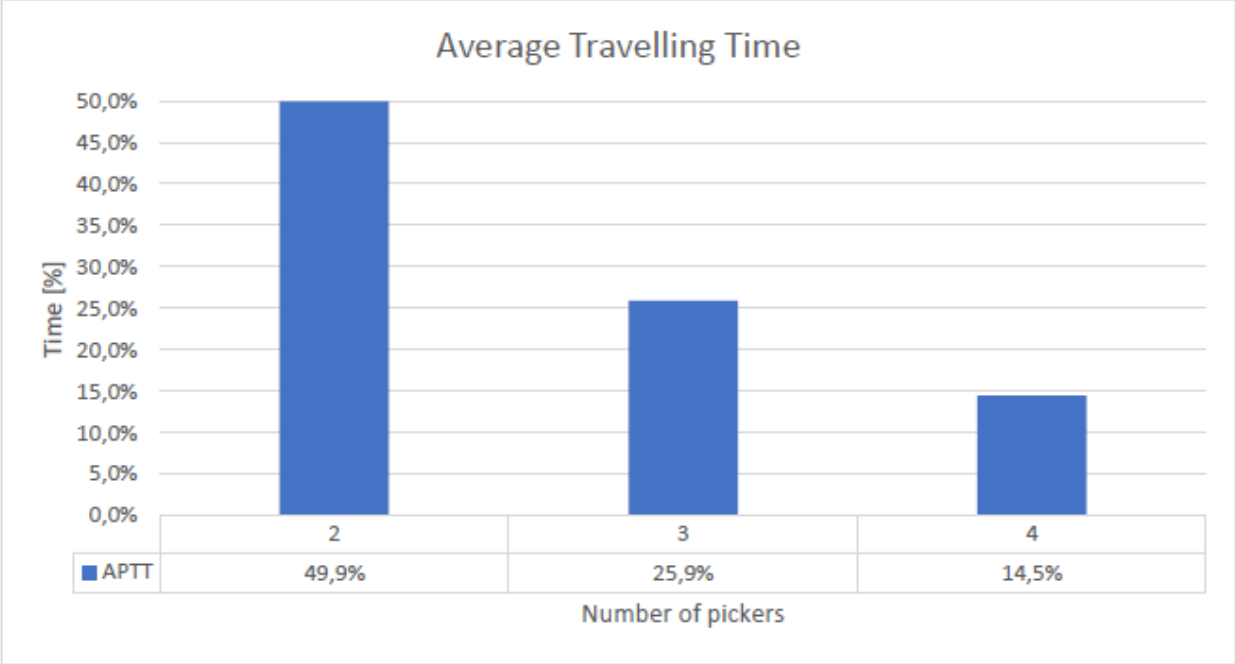

Figura 26: Tiempo medio de viaje de los recogedores

El último indicador es el tiempo medio de espera del robot (AWRT). En este caso se puede observar una ligera disminución del AWRT al aumentar el número de robots mientras se mantiene un número fijo de recogedores: para 2 recogedores y 6 robots el AWRT es del 36,3 % y con 10 robots disminuye en un 6,7 % hasta un 33,9 %; para 3 recolectores y 6 robots la AWRT es del 65 % y con 10 robots, con una disminución del 2,7 %, es del 63,2 %; y para 4 recolectores y 6 robots este indicador toma el valor de 78,4 %, disminuyendo sólo un 0,9 % al aumentar el número de robots a 10, con una AWRT del 77,7 %. Aquí se identifican dos patrones, uno es que a mayor número de robots menor AWRT, lo cual tiene sentido ya que al haber más robots disponibles es más probable que los recolectores sean necesitados por alguno de ellos. El segundo patrón es que a medida que aumenta el número de recolectores, la tasa de disminución de la AWRT por añadir robots es menor. En otras palabras, cuando hay más recolectores en el sistema, no importa cuántos robots haya si

son suficientes teniendo en cuenta la tasa de llegada de pedidos actual.

Aunque se produce una disminución del indicador con el aumento del número de robots, esta reducción no es muy elevada, por lo que un análisis como los realizados para el APPT y el APTT puede ser útil para comprender el impacto de añadir más recolectores al sin dar importancia al número de robots. La Figura 27 muestra la media del AWRT para cada número de recolectores. Al añadir más preparadores, el tiempo que pasan de media esperando a ser asignados a un robot aumenta significativamente. Esto es lógico, ya que la tasa de llegada de pedidos es fija, por lo que los robots necesitarán para los preparadores una cantidad casi constante. Por lo tanto, tener más preparadores disponibles crea una cola virtual, lo que produce más tiempo de espera.

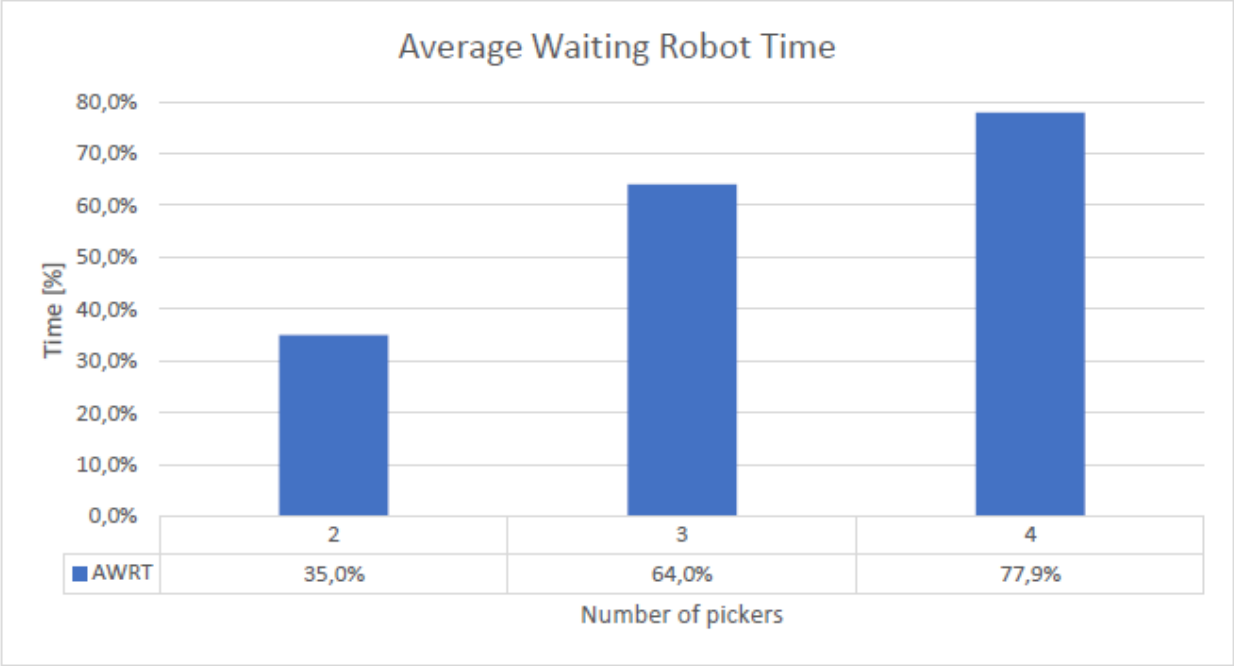

Figura 27: Tiempo medio de espera del robot de los recogedores

Ahora, consolidando los resultados en una medida más completa para entender la utilización de los recolectores, se utilizará el Tiempo Medio Acoplado y Desacoplado. Como se explicó en capítulos anteriores, el tiempo acoplado representa el tiempo en que la recolectora está aco-

plada o vinculada a un robot, en este caso, mientras se desplaza o recolecta, que coincide con los momentos en que está siendo utilizada activamente. El Tiempo Acoplado puede entenderse como la Utilización Media del Recolector (APU). Por el contrario, el tiempo desacoplado es cuando no está vinculado a un robot, es decir, cuando está esperando a ser asignada a un robot, por lo que no está siendo utilizada de manera productiva.

La Figura 28 es el mismo gráfico de la Figura 24 pero agregando los indicadores en Tiempo Acoplado y Desacoplado. No es necesario volver a explicar el tiempo desacoplado, ya que es equivalente a la AWRT. En cambio, el Tiempo Acoplado es la suma del APPT y el APTT. Como ambos indicadores tienen exactamente el mismo comportamiento, los análisis son válidos también para el Tiempo Acoplado. Así, los únicos escenarios en los que el APU está por encima del 50 % son los que tienen 2 recolectores, independientemente del número de robots.

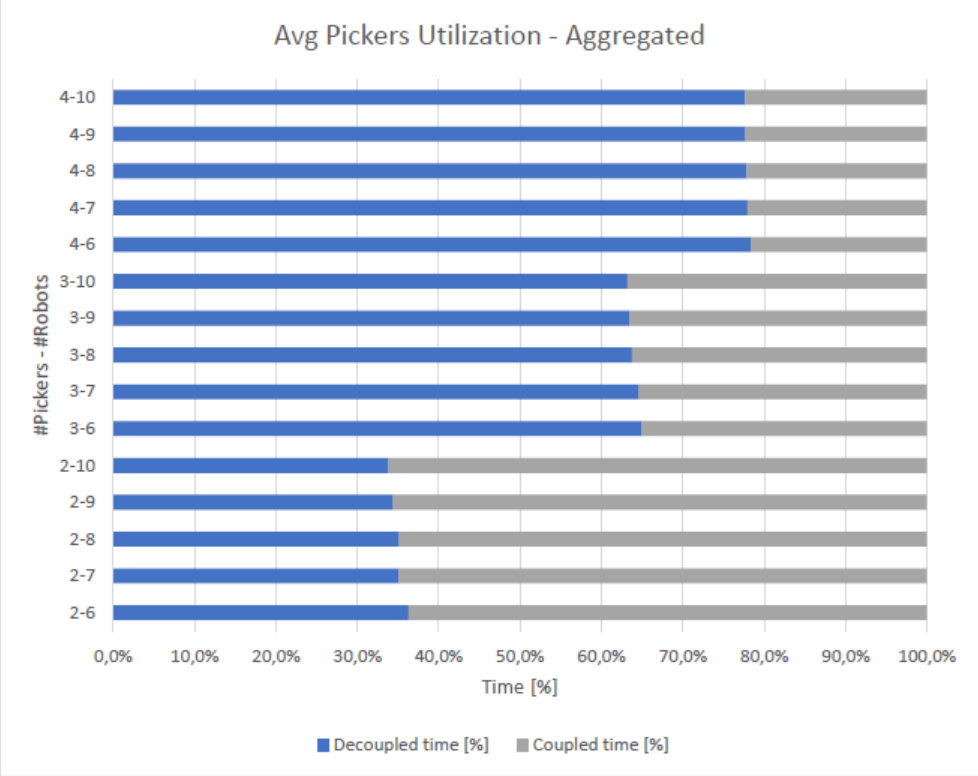

Figura 28: Utilización media de los recogedores - Agregada

6.3.3. Utilización media de los robots

Para analizar la Utilización Media del Robot (UAR), al estar compuesta por varios componentes, se desglosarán los resultados en Tiempo Activo y Tiempo Pasivo, comentando cada uno de los componentes de estos subgrupos. Al final de la sección se analizará el ARU de forma general.

El Tiempo Activo se compone del Tiempo Medio de Desplazamiento del Robot (ARTT) y Tiempo Medio de Recogida del Robot (ARPT). La Figura 29 toma el Tiempo Activo de cada escenario como el 100 % y representa la división del ARPT y el ARTT. En el gráfico se observa que las proporciones del Tiempo Activo no cambian significativamente entre los distintos escenarios.

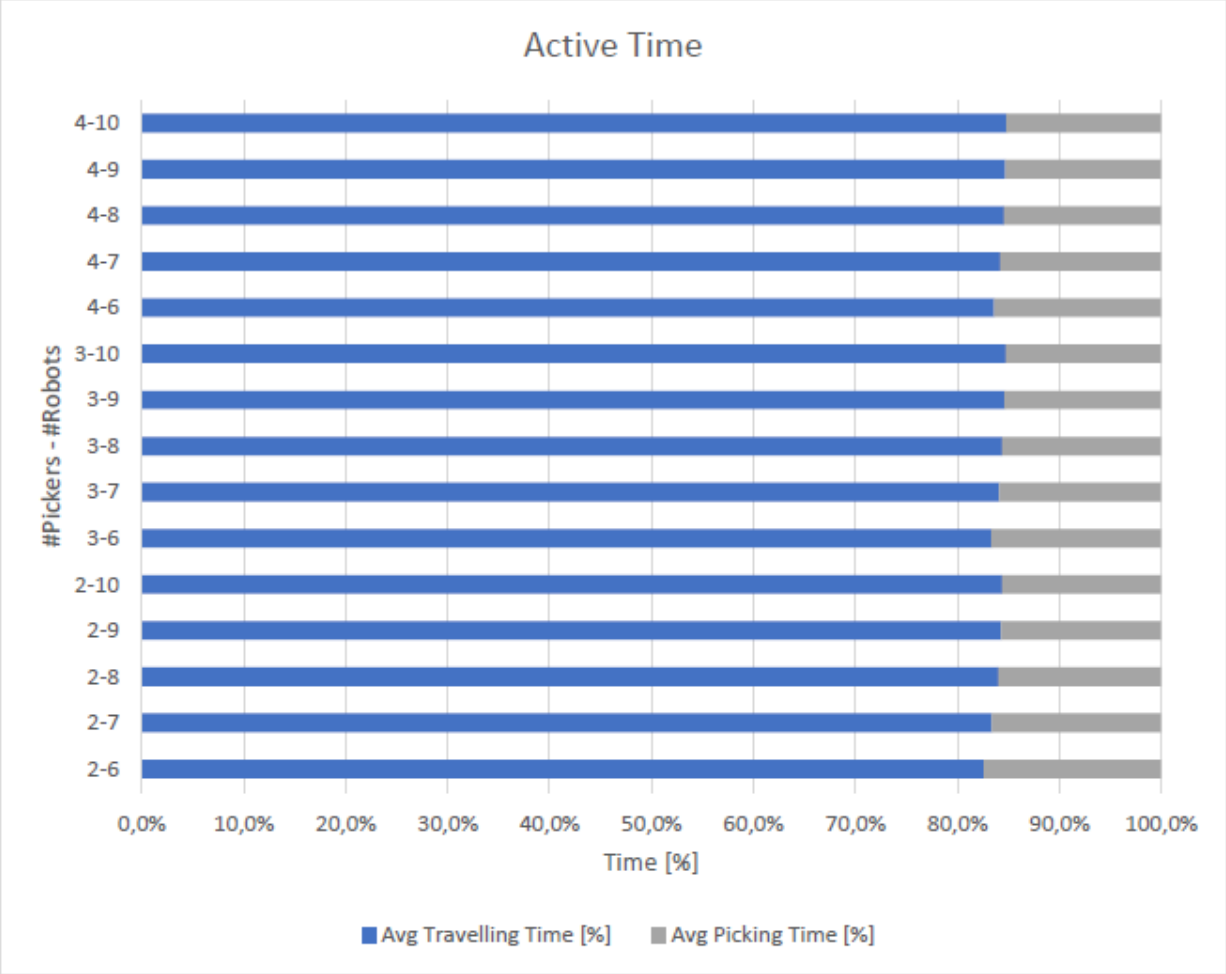

Figura 29: Tiempo activo medio de los robots

Esto puede confirmarse además en la Tabla 4 donde se calculan las medidas de tendencia central de ARTT y ARPT y ARTT - ARPT. Las medias están muy cercanas a sus máximos y mínimos y la desviación ronda el 1 %.

Tabla 4: Medidas de Tendencia Central para el Tiempo Activo

Indicator	**ARTT**	**ARPT**	**Difference**
Average	84.1 %	15.9 %	68.3 %
Max	84.8 %	17.4 %	69.6 %
Min	82.6 %	15.2 %	65.2 %
Std Deviation	0.6 %	0.6 %	1.2 %

El análisis es diferente para el Tiempo Pasivo, compuesto por el Tiempo Medio de Espera de Pedido (TMPO), el Tiempo Medio de Carga (TMPC) y el Tiempo Medio de Espera de Recogida (TMPR). En cuanto al Tiempo Activo, la Figura 30 se construye considerando el Tiempo Pasivo de cada escenario como el 100 %. A primera vista, se puede observar que más del 60 % del Tiempo Pasivo se dedica a la espera de pedidos, una parte menor del tiempo se dedica a la espera de recolectores, y sólo una pequeña parte se dedica a la carga de los robots.

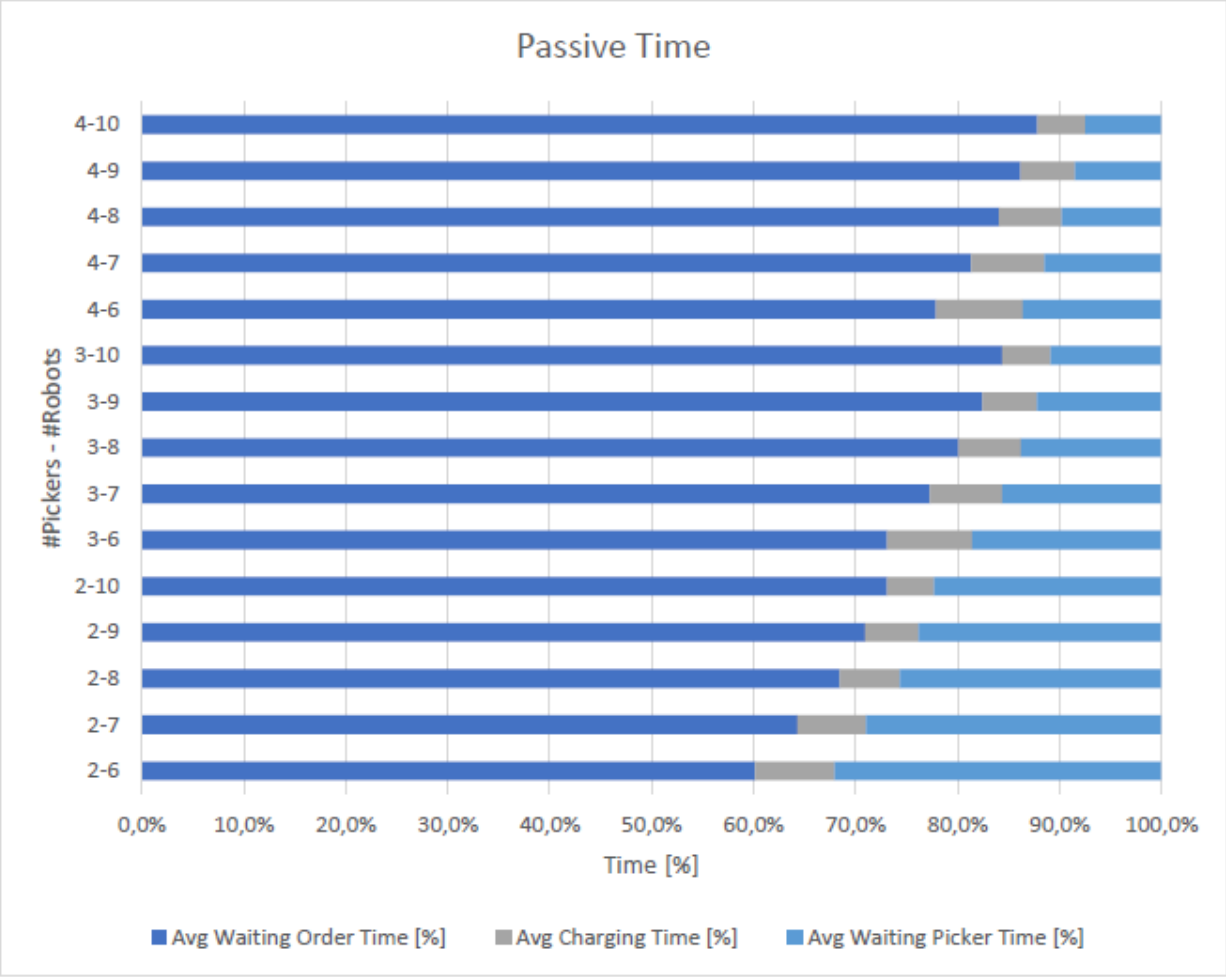

Figura 30: Tiempo medio de carga

Profundizando ahora en cada indicador, para el ACT, como no se percibe fácilmente en la Figura 30, considere la Figura 31 en la que se representan los mismos datos pero en detalle. Se puede observar que para este indicador el número de recolectores no es relevante, lo cual es lógico ya que la carga de los robots no tiene nada que ver con los recolectores, también porque el modelo asume que no se gasta energía mientras el robot está esperando; los robots sólo gastan energía mientras se desplazan. La segunda observación es que al aumentar el número de robots, el ACT disminuye. Teniendo en cuenta el mismo supuesto de cuándo los robots gastan energía y cuándo no, este resultado es consecuencia de tener una tasa de llegada de pedidos fija, porque

esto significa un número plano de pedidos en cada escenario que tiene que ser gestionado por un número creciente de robots que, por tanto, viajarán menos de media, es decir, gastarán menos energía.

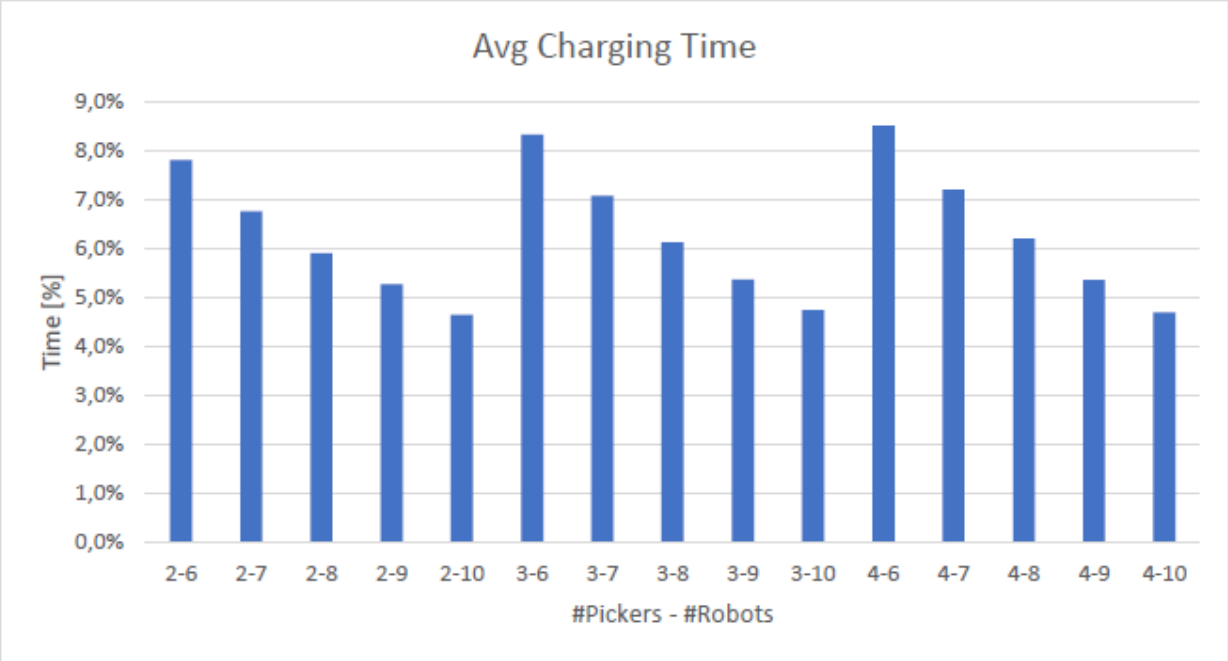

Figura 31: Tiempo pasivo medio de los robots

En cuanto al AWPT, el resultado muestra que al aumentar el número de recolectores y robots se produce una disminución del indicador. Es interesante ver en el gráfico que desde el escenario 2-6 hasta el escenario 3-10, el AWPT pasa del 32 % a un 10,9 % a un ritmo casi constante, pero luego cuando los escenarios son con 4 recogedores, el patrón se rompe. Esto se puede explicar de la siguiente manera: considerando una tasa de llegada de pedidos constante, tener menos recolectores implica que será más probable que se acoplen a un robot, recordemos que para escenarios con 2 recolectores de media se acoplan más del 70 % de las veces. Pasando a 3 recolectores, el efecto persiste ya que, aunque se acoplen sólo alrededor del 35 % del tiempo, el Tiempo Medio de Rendimiento del Pedido sigue mejorando. Sin embargo, pasar de 3 a 4 preparadores no afecta

significativamente al AOTT, por lo que la situación no es muy diferente incluso considerando el preparador adicional.

Por último, en cuanto al AWOT, se observa que aumenta tanto al aumentar el número de recolectores como de robots. Manteniendo fijos los recogedores y aumentando el número de robots, se observa un ligero aumento del AWOT, a medida que aumenta el número de recogedores, la tasa de aumento del AWOT disminuye con el aumento de robots. Por ejemplo, pasar de 6 a 10 robots con 2 recogedores supone un aumento del 21,5 % (del 60,2 % al 73,1 %) mientras que la misma situación pero para 4 recogedores supone un aumento de sólo el 12,7 % (del 77,9 % al 87,9 %).

El mismo fenómeno se produce si el análisis se realiza manteniendo fijo el número de robots y aumentando el número de recogedores. En este caso el patrón más significativo es que pasar de 2 a 3 recogedores implica un incremento en el AWOT mucho mayor que el obtenido al pasar de 3 a 4 recogedores. Tomando como ejemplo el caso de 6 robots y el caso de 8 robots, considerando 2 recogedores como base para calcular las tasas, en el primer caso, pasar de 2 a 3 recogedores supone un incremento del 21,6 % y de 3 a 4 uno del 29,5 %. En el segundo, pasar de 2 a 3 recogedores supone un aumento del 17,1 %, mientras que pasar de 3 a 4 sólo un 22,8 %.

Estos comportamientos pueden explicarse si se considera de nuevo que la tasa de llegada de pedidos es constante, por lo que en cada escenario llega un número fijo de pedidos que tienen que ser atendidos por más agentes. En este caso, tener más robots disponibles para recibir pedidos implica que algunos robots tendrán que esperar más tiempo para ser asignados a un pedido, aumentando la media.

Combinando todos los indicadores para analizar el ARU, que se entenderá como el Tiempo Activo empleado por el robot durante el tiempo de simulación, se construyó la Figura 32. En este gráfico, el tiempo de simulación se tomó como el 100 % y las proporciones de Tiempo Activo

y Pasivo se representan en porcentajes. Puede observarse que en todos los escenarios el Tiempo Pasivo representa al menos el 70 %. Otra observación es que el número de recolectores no influye significativamente en el ARU, el factor principal es únicamente el número de robots. El tercer resultado es que al aumentar el número de robots, el ARU disminuye, ya que más robots deben gestionar el sistema considerando una tasa de llegada de pedidos constante.

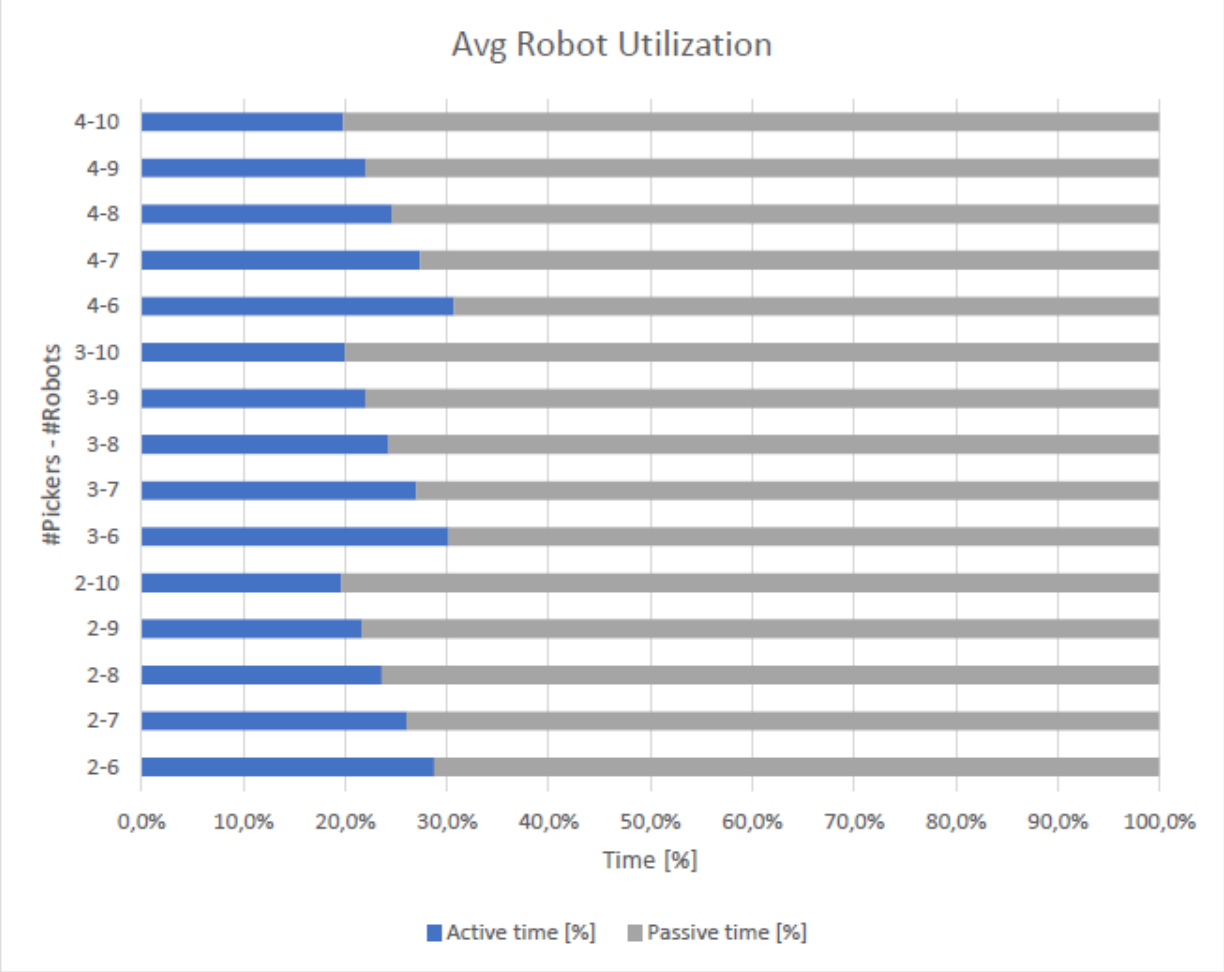

Figura 32: Utilización media de los robots

6.4. Escenarios 2

La nomenclatura utilizada en los gráficos de este apartado para identificar los diferentes escenarios es Número de Robots ; Factor de Tasa de Llegada de Pedidos, recordando que lo que es constante es la relación entre el número de robots y el número de recogedores. Así, si por ejemplo, el es 6 ; 2, implica que el número de recogedores es 3, o si el escenario es 10 ; 1, entonces el número de recogedores es 5, manteniendo siempre el ratio igual a 2.

6.4.1. Tiempo medio de procesamiento de pedidos

En cuanto a los Escenarios 2, se explicará cada componente del Tiempo Medio de Procesamiento de Pedidos (ATT) por separado. En la figura 33 se representa el tiempo medio de asignación de pedidos en espera (AOWAT) para cada escenario. Este gráfico sólo es útil para identificar un gran valor atípico, el escenario 4 ; 3 con 1367 segundos, casi la mitad del tiempo de simulación. Esto se produce porque la cantidad de recolectores y robots no es suficiente para gestionar esta considerable tasa de llegada de pedidos, por lo que los pedidos se acumulan y tienen que esperar hasta que los agentes estén disponibles para ser asignados. Para analizar mejor el resto de escenarios se propone el gráfico de la Figura 34. Este gráfico tiene los mismos datos pero el escenario 4 ; 3 no está representado. Aquí se puede observar que el AOWAT es muy bajo para todos los escenarios (menos de 15 segundos). Además, al aumentar el número de recolectores y robots, el indicador disminuye a medida que aumenta la capacidad de gestión de pedidos. Por último, el AOWAT aumenta al aumentar el factor OAR y mantener el mismo número de robots y recolectores, ya que los mismos recursos deben gestionar más pedidos, por lo que se generan colas de pedidos.

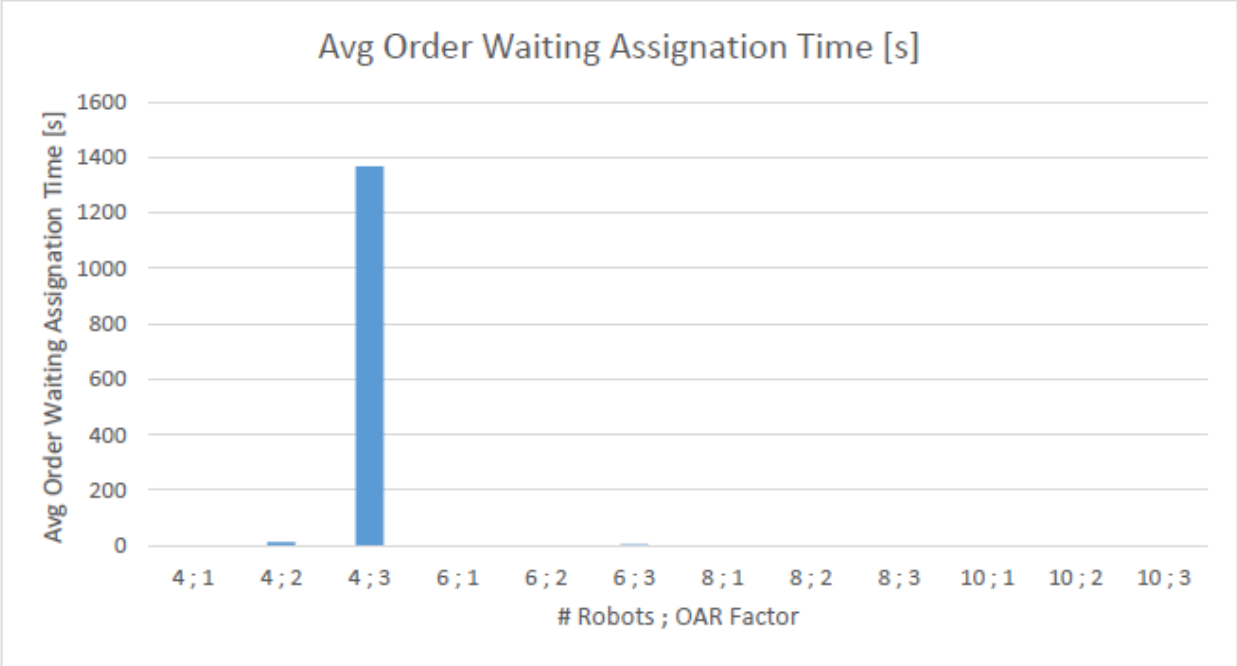

Figura 33: Tiempo medio de asignación de pedidos (Escenarios 2)

El Tiempo medio de movimiento en espera (AWMT) también es muy bajo, como puede verse en la Figura 35 es inferior o igual a 9 segundos en todos los escenarios. En términos generales, para un número fijo de robots y recolectores, el AWMT aumenta al aumentar el Factor OAR, ya que llegan más pedidos y se asignan a los robots, por lo que tienen que esperar tanto si reciben 3 pedidos como si esperan el tiempo máximo de espera. El único escenario en el que esto no ocurre es el de 4 ; 3. En este caso, como la capacidad es demasiado baja para la tasa de llegada de pedidos, en cuanto un robot está disponible recibe inmediatamente todos los pedidos disponibles (máximo 3) que están en cola, por lo que el AWMT se reduce casi a 0.

A continuación, en la Figura 36 se muestra el resultado del Tiempo medio de proceso (APT). El patrón es claro, manteniendo fijo el número de recolectores y robots, al aumentar el Factor OAR, el APT aumenta, ya que hay que procesar más pedidos. En esta línea, una observación valiosa es que cuando los recursos son escasos, este incremento es mucho mayor que en los

escenarios en los que los recursos son suficientes para satisfacer la demanda. Por ejemplo, en los escenarios con 4 robots, el aumento del APT es muy elevado, del Factor OAR 1 al 3 el APT aumenta un 112 % (de 82 segundos a 175 segundos), mientras que para los escenarios con 10 robots, el aumento es del 13 % (de 54 segundos a 61 segundos).

Otra observación digna de mención es que los APT de los escenarios con 8 robots no son significativamente diferentes de los de 10 robots, ya que a partir de 8 robots los recursos son suficientes para satisfacer los pedidos, por lo que los beneficios sobre el rendimiento por añadir más robots son cada vez menos importantes.

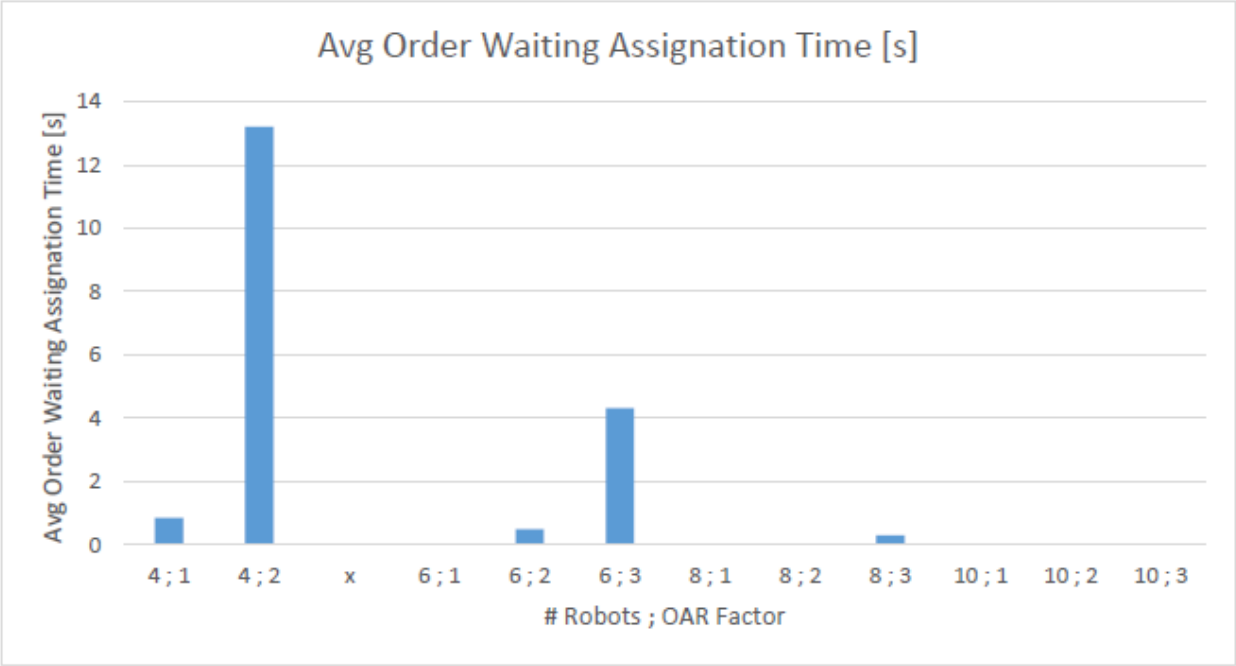

Figura 34: Tiempo medio de asignación de pedidos (Escenarios 2) - Sin el escenario 4 ; 3

Por último, en la Figura 37 se representa el tiempo medio de rendimiento como la suma del APT y el AWOT. Dado que el AWOT del escenario 4 ; 3 induce a error en el análisis debido a su elevado valor, se propone otro gráfico con los mismos datos pero sin este escenario en la Figura 38. En este caso, todo el análisis ya explicado para el APT puede utilizarse para el ATT, ya que para

todos estos escenarios (excluyendo 4 ; 3) el AWOT es muy inferior al APT, por lo que no afecta significativamente a los comentarios realizados.

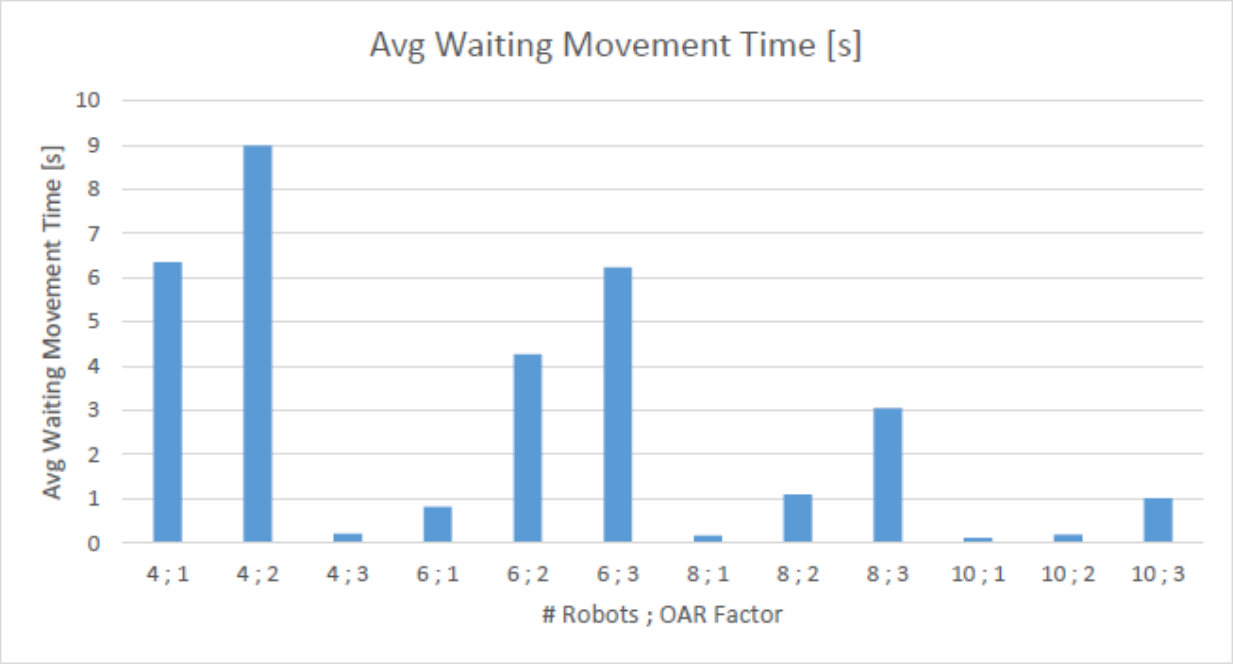

Figura 35: Tiempo medio de espera de movimiento (Escenarios 2)

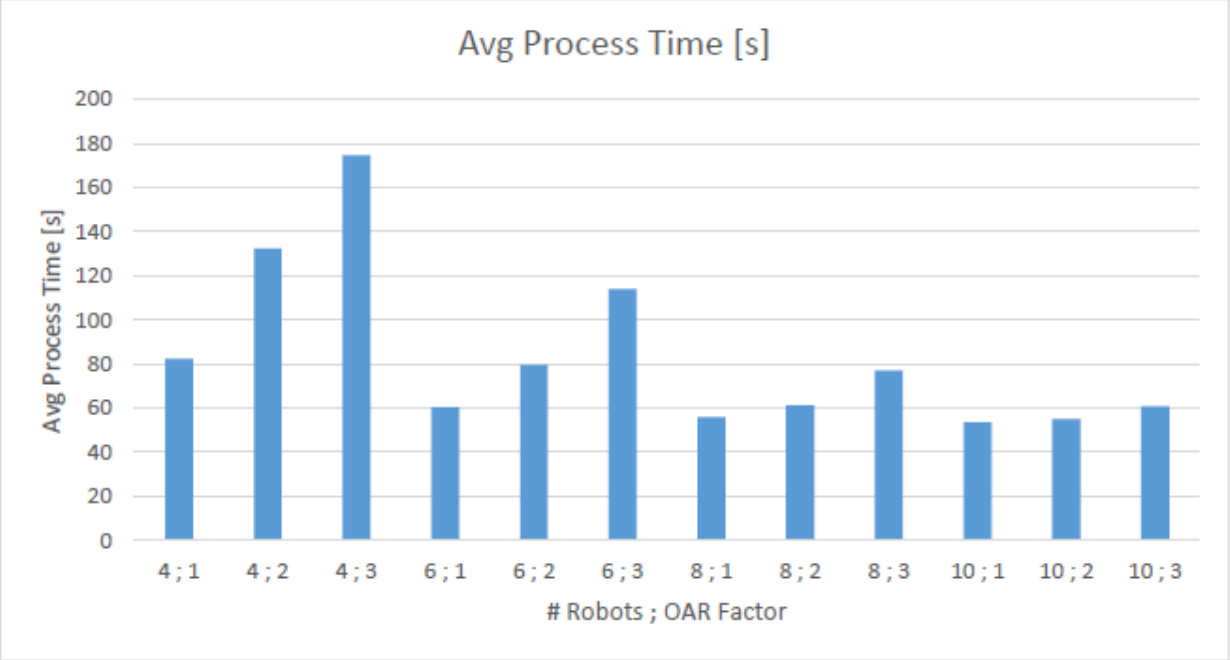

Figura 36: Tiempo medio de proceso de pedidos (Escenarios 2)

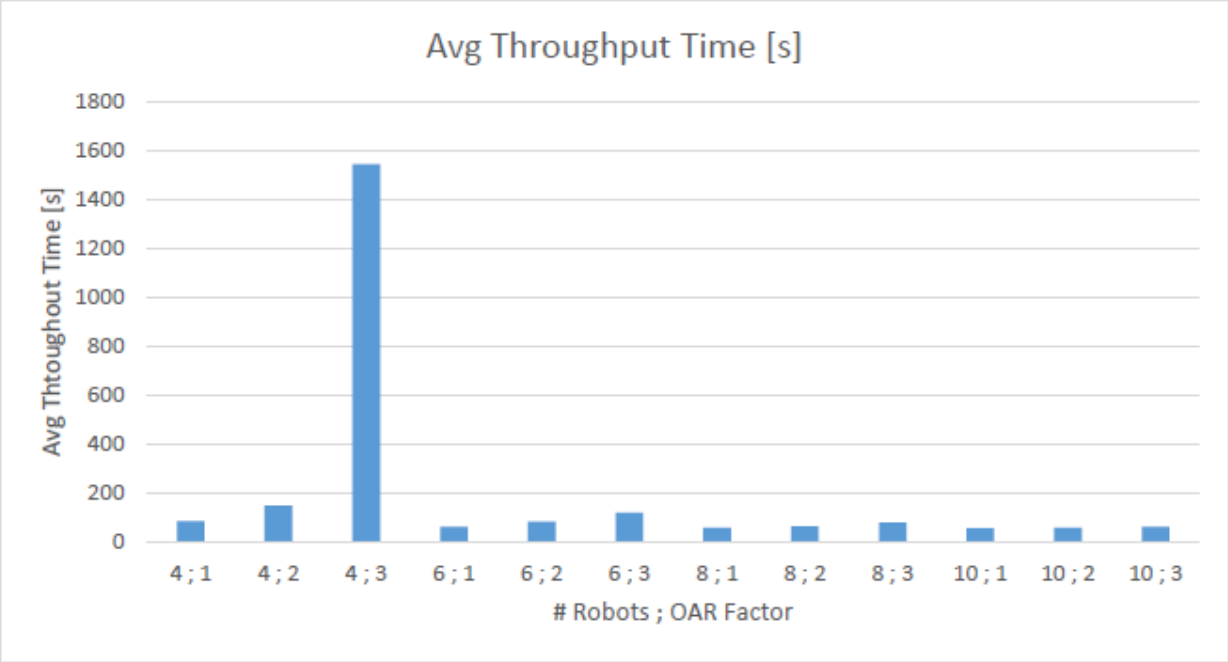

Figura 37: Tiempo medio de procesamiento de pedidos (Escenarios 2)

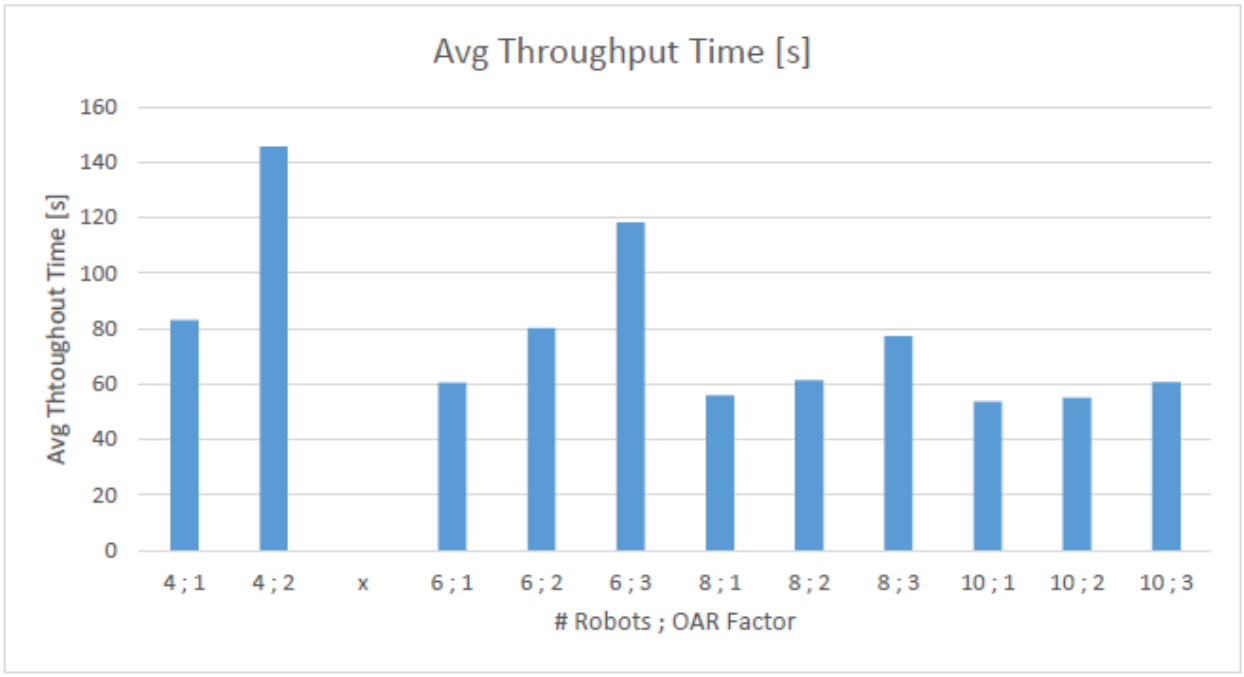

Figura 38: Tiempo medio de procesamiento de pedidos (Escenarios 2) - Sin el escenario 4 ; 3

6.4.2. Utilización media de los recogedores

Bajo una vista general, es decir, al nivel de tiempo acoplado y desacoplado, para entender la Utilización Media de los Recogedores (APU), fue diseñada la Figura 39. Esta muestra que solo en 3 escenarios el APU es mayor a 50 %: en el escenario 4 ; 3 con un 88.5 %, el 4 ; 2 con un 61.8 % y en el 6 ; 3 con un 57.7 %. También el APU crece al incrementar el factor de la tasa de llegada de pedidos, ya que se usan los mismos recursos para manejar más órdenes. Además, al mantener este factor constante, el APU decrece cuando se incrementa el número de recursos, pues más recursos satisfacen un mismo número de pedidos.

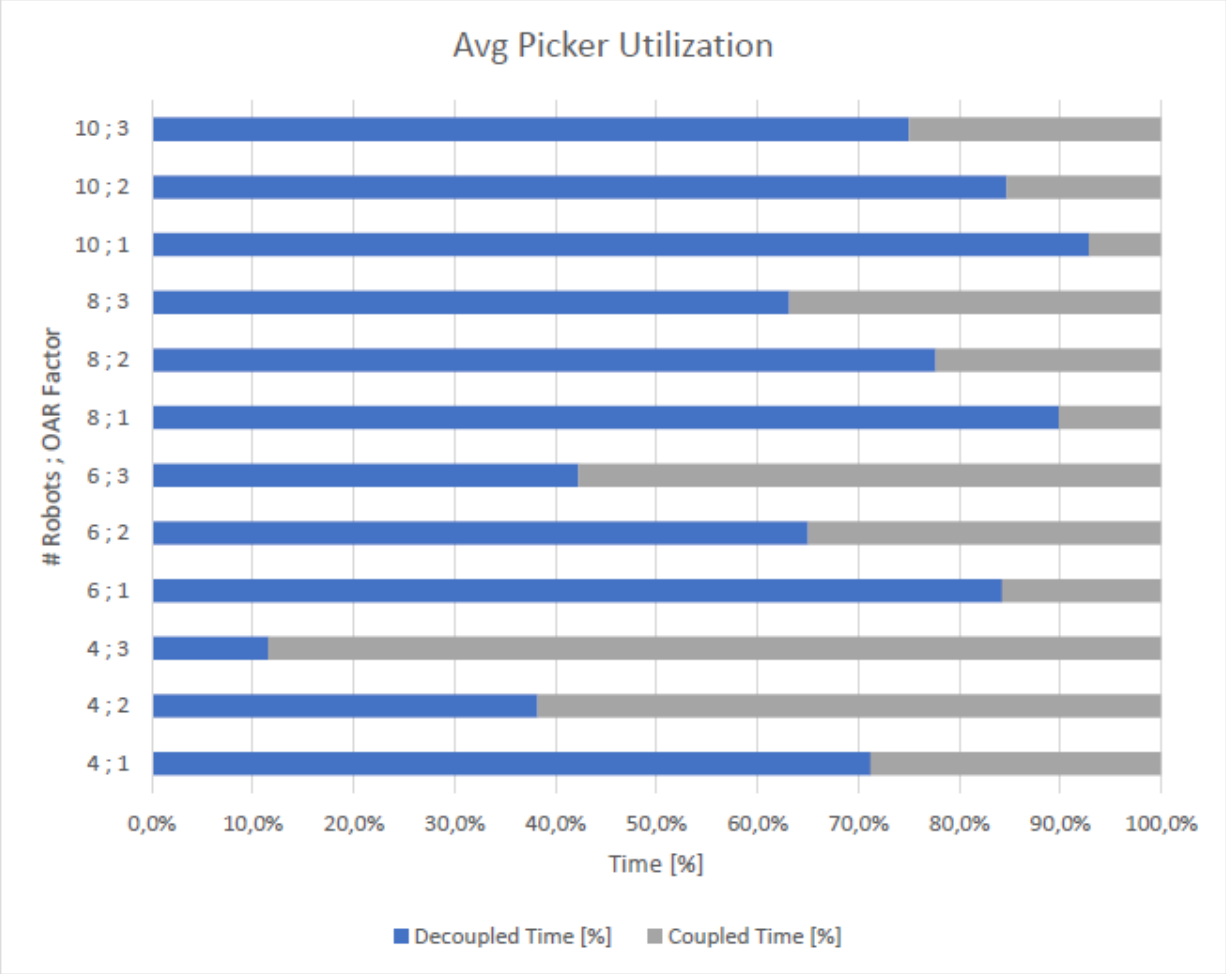

Figura 39: Utilización media de los recogedores (Escenarios 2)

A continuación se detallan sus componentes. En primer lugar, el Tiempo Medio de Espera del Robot AWRT) es equivalente al Tiempo Desacoplado. En la Figura 40 puede observarse que el aumento del factor OAR reduce el AWRT, ya que hay más pedidos que gestionar y, por tanto, más ubicaciones de picking que visitar. Además, entre los casos de 8 robots y 10 robots las diferencias no son significativas, ya que los recursos con los que se cuenta son suficientes para satisfacer los pedidos a esas tasas de llegada de pedidos.

En cuanto al tiempo medio de recogida (APPT), el resultado se muestra en la Figura 41 y muestra que el tiempo empleado en la preparación de pedidos aumenta a medida que hay más pedidos que preparar. Además, al aumentar el número de robots y recolectores manteniendo fijo el Factor OAR, disminuye el APPT. Esto se debe a que el mismo nivel de pedidos es gestionado por más agentes, por lo que cada preparador debe gestionar de media menos pedidos.

Por último, el tiempo medio de desplazamiento del recolector (APTT) aumenta a medida que lo hace el factor OAR, ya que al haber más pedidos que gestionar, los recolectores deben desplazarse a más ubicaciones de recogida. Por el contrario, manteniendo fijo el Factor OAR, el APTT disminuye al añadir más robots y recolectores. Todo esto se puede visualizar en la Figura 42.

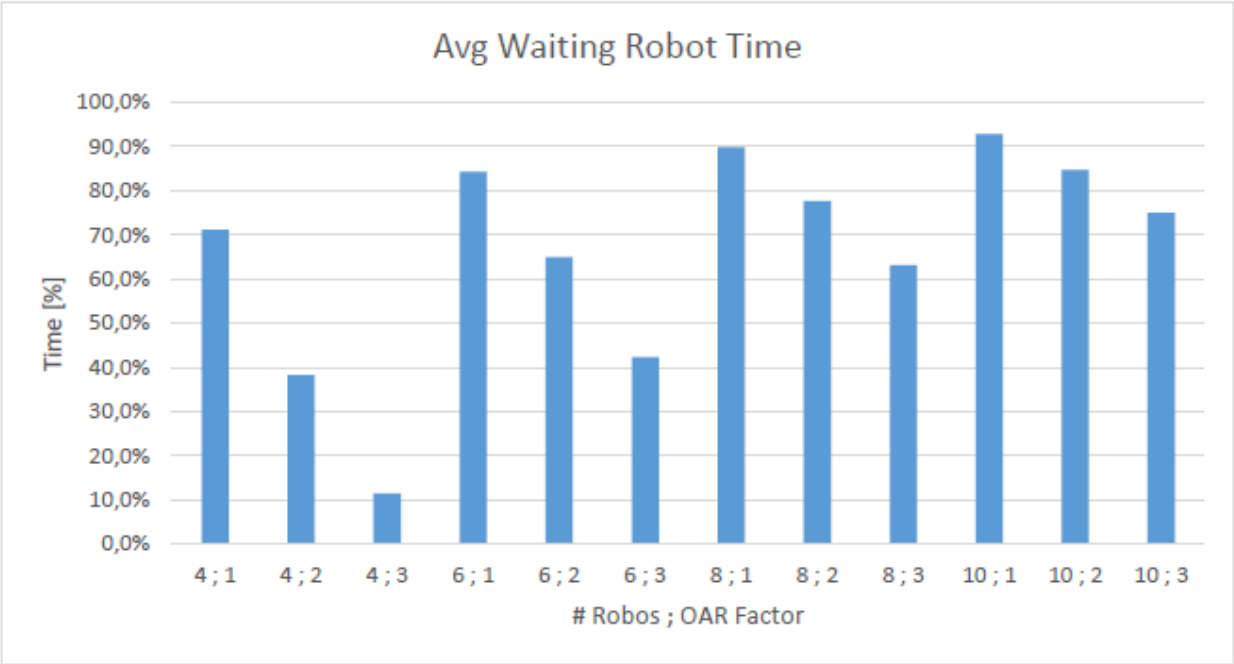

Figura 40: Tiempo medio de espera de un robot (Escenarios 2)

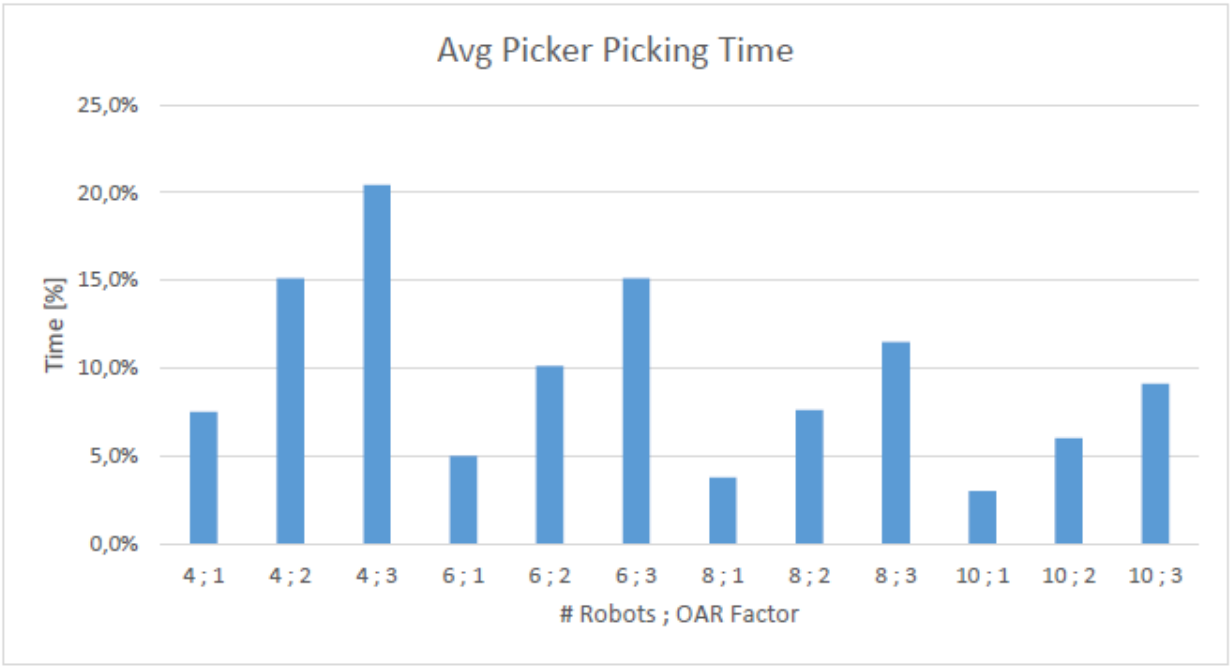

Figura 41: Tiempo medio de picking (Escenarios 2)

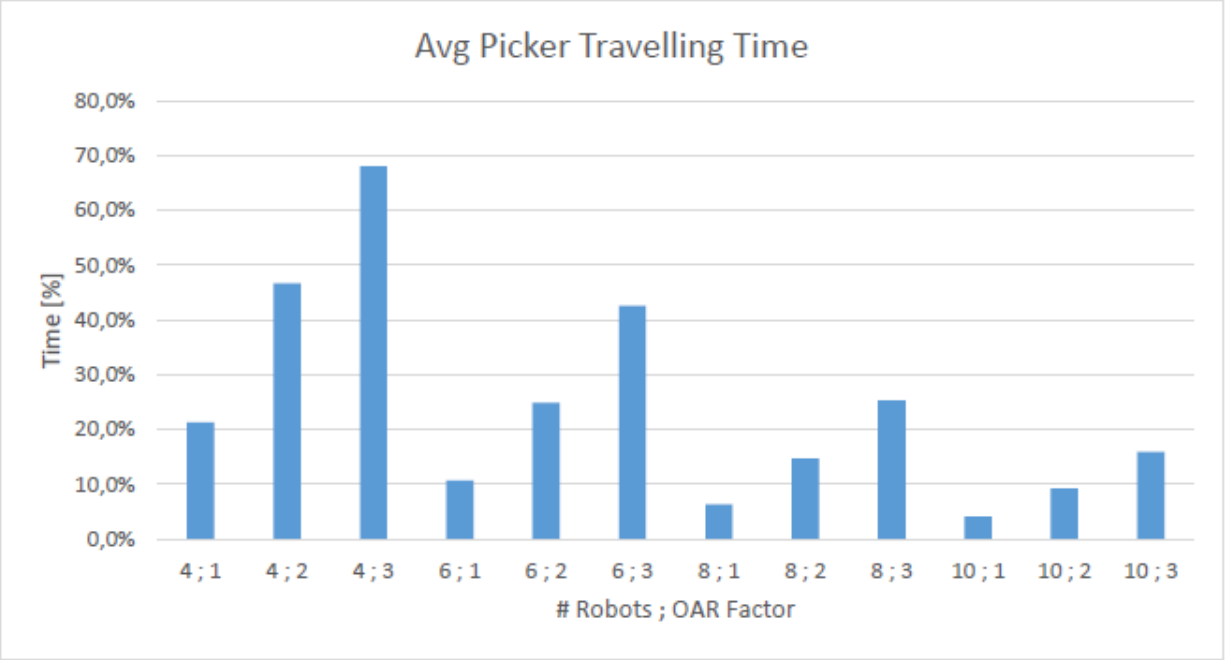

Figura 42: Tiempo medio de viaje de los recogedores (Escenarios 2)

6.4.3. Utilización media de los robots

En primer lugar, considerando el Tiempo Activo como el 100 %, tal y como se ilustra en la Figura 43, cambiar el número de recolectores y robots, manteniendo su proporción fija, no afecta significativamente a la proporción de Tiempo Medio de Desplazamiento del Robot (ARTT) y Tiempo Medio de Recogida del Robot (ARPT); siempre está en torno a un 80:20, respectivamente. En cambio, si se modifica el factor OAR manteniendo un número fijo de recolectores y robots, se produce un ligero cambio en la proporción: a medida que aumenta el factor OAR, aumenta ligeramente la proporción de ARPT. Sin embargo, esto sólo es cierto para los escenarios en los que hay 4 y 6 robots.

Para entenderlo mejor, consideremos la Figura 44 y la Tabla 5 en las que se muestran los valores de tiempo absolutos de estos dos indicadores de rendimiento. De hecho, el cambio en el

ARPT es proporcionalmente mayor que el del ARTT para los escenarios con 4 y 6 robots, pero son muy similares en los escenarios con 8 y 10 robots. La razón es que cuando haya suficientes robots para satisfacerlos, habrá ineficiencias (como se mostrará en el análisis del tiempo pasivo), por lo que, de media, cada robot no dedicará más tiempo al picking ni a los desplazamientos.

Tabla 5: Tiempo Promedio de Viaje de Robot y Tiempo de Picking (Escenarios 2)

NR - OAR Factor	Average Travelling Time [s]	Change	Average Picking Time [s]	Change
4 ; 1	5552		1082	
4 ; 2	8249	49 %	2176	101 %
4 ; 3	8707	6 %	2939	35 %
6 ; 1	4004		721	
6 ; 2	7258	81 %	1456	102 %
6 ; 3	8843	22 %	2178	50 %
8 ; 1	3033		545	
8 ; 2	6009	98 %	1096	101 %
8 ; 3	8212	37 %	1653	51 %
10 ; 1	2437		432	
10 ; 2	4844	99 %	866	100 %
10 ; 3	7126	47 %	1313	52 %

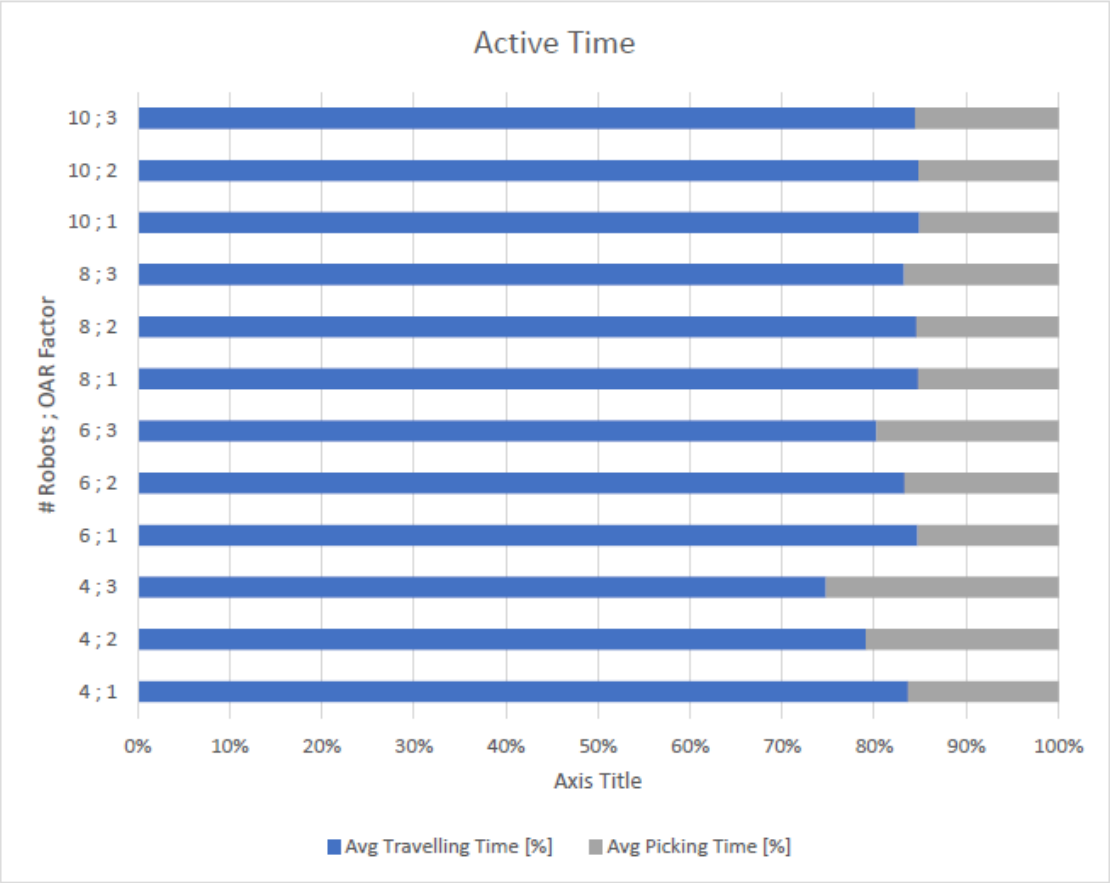

Figura 43: Tiempo Activo de los robots (Escenarios 2)

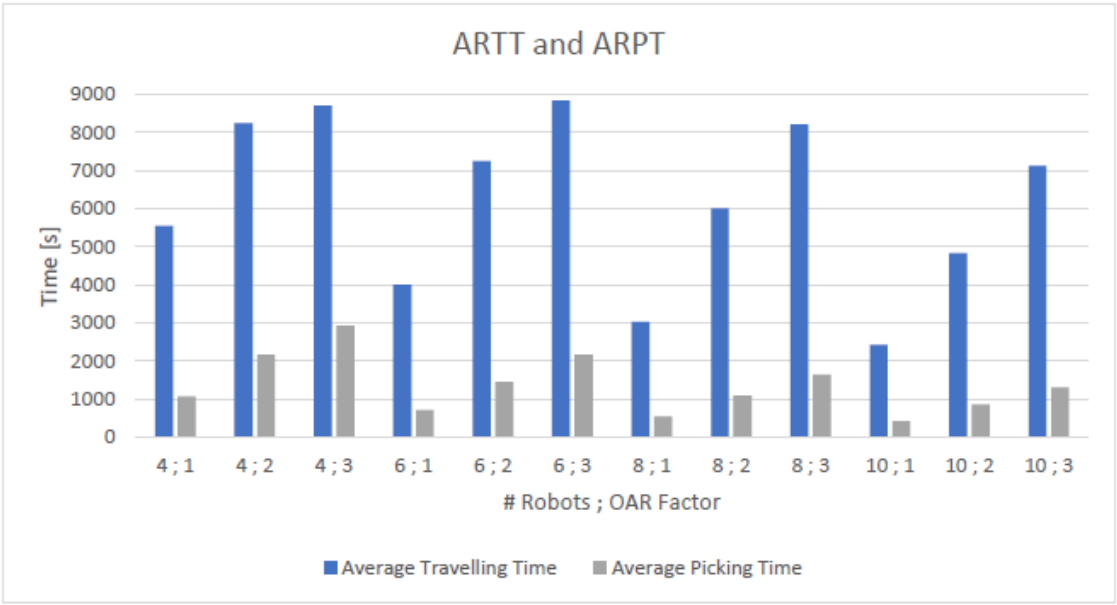

Figura 44: Tiempo medio de viaje y de picking de los robots (Escenarios 2)

En la Figura 45 se ilustra la proporción de los distintos tiempos pasivos. La primera conclusión es que, a medida que hay más pedidos, aumenta la proporción de tiempo dedicado a la carga, ya que los robots tienen que desplazarse más. Asimismo, la proporción del Tiempo medio de espera del recolector (AWPT) aumenta a medida que aumenta el Factor OAR, para un número fijo de robots y recolectores, ya que la proporción es siempre la misma; al haber más pedidos que gestionar, es lógico que los robots tengan que esperar más tiempo a los recolectores que estarán más ocupados. Como la proporción del AWPT y el ACT aumentan cuando el Factor OAR aumenta, el AWOT debe disminuir. Al haber más pedidos disponibles, los robots tendrán que esperar menos tiempo para recoger los 3 pedidos y empezar a desplazarse. Otra observación es que, para los escenarios en los que hay 6, 8 y 10 robots, la proporción de AWOT aumenta a medida que aumenta el número de robots, pero a un ritmo equilibrado. Sin embargo, en el escenario 4 ; 3, el AWOT es demasiado bajo, ya que el número de robots no es suficiente para satisfacer los pedidos, como se muestra en el análisis anterior.

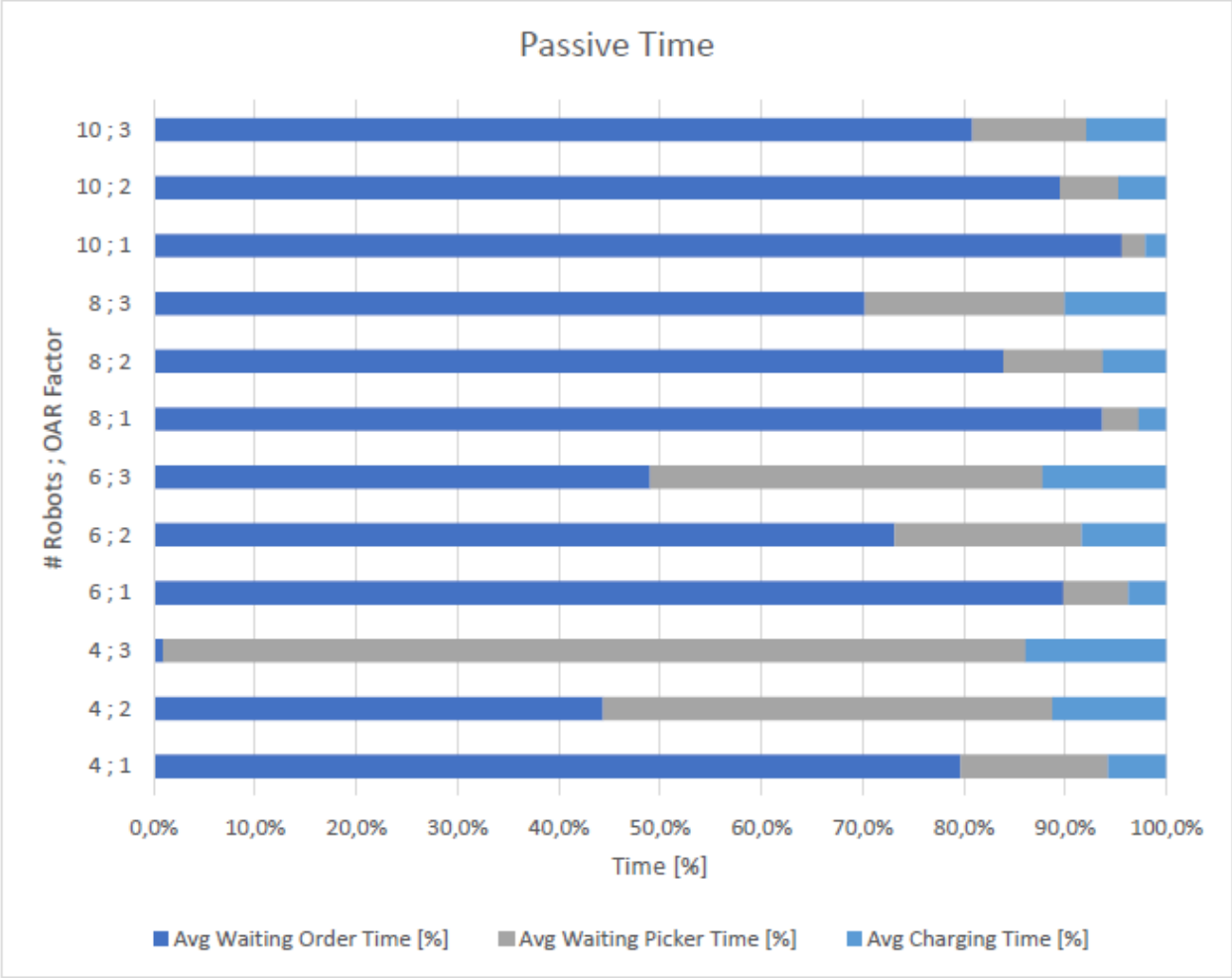

Figura 45: Tiempo pasivo de los robots (Escenarios 2)

Por último, la Figura 46 muestra la utilización media del robot (ARU) en porcentaje. La ARU es mayor cuando aumenta el Factor OAR, ya que hay más pedidos que gestionar, por lo que los robots tienen que dedicar más tiempo a desplazarse y hacer picking. Para un Factor OAR específico, como se analiza en los Escenarios 1, al aumentar el número de agentes, la ARU disminuye, ya que hay más robots gestionando un número constante de pedidos. Para un Factor OAR igual a 1, la ARU oscila entre el 10 % y el 20 %, cuando es 2 oscila entre el 20 % y el 35 %, y cuando toma el valor de 3, la ARU se mueve en torno al 30 % y el 40 %.

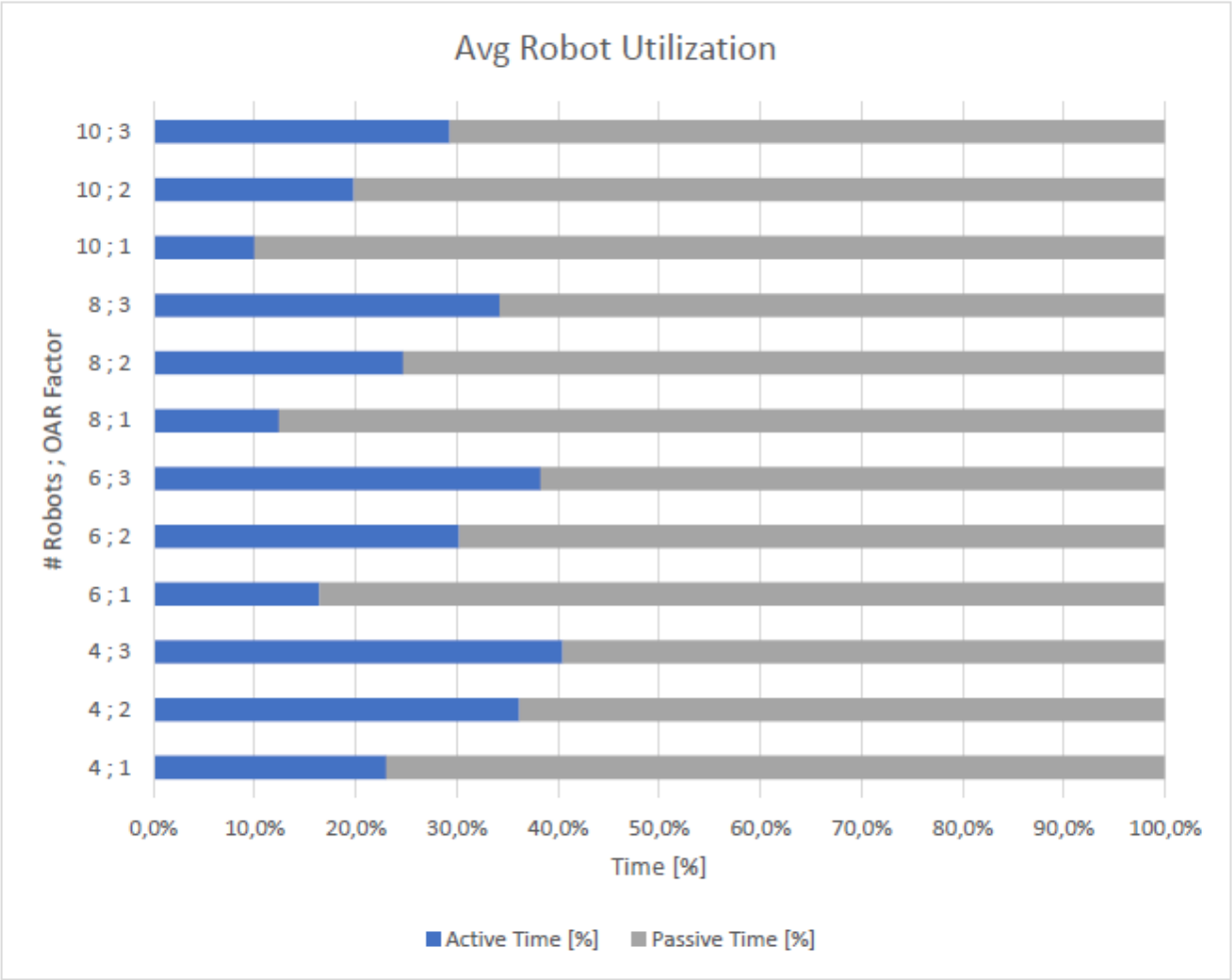

Figura 46: Utilización media de los robots (Escenarios 2)

7. Conclusiones

En esta sección se ofrecen las respuestas a la pregunta de investigación y se formulan observaciones finales. Tras el análisis de los resultados de la simulación proporcionado en el capítulo anterior, es posible responder a las Preguntas de Investigación. Los Escenarios 1 se diseñaron para responder a las RQ1 y RQ2, mientras que los Escenarios 2 se propusieron para dar respuesta a las RQ3 y RQ4:

- En cuanto al impacto en el rendimiento del sistema al cambiar el número de recogedores y el número de robots pero fijando la tasa de llegada de pedidos (OAR) (RQ1), se seleccionó el tiempo medio de procesamiento de pedidos (AOTT) indicador de rendimiento a analizar. Se observó que al aumentar el número de agentes en general, robots y/o recolectores, el AOTT disminuye, por lo que mejora el rendimiento del sistema. También se observó que la tasa de disminución de la AOTT al aumentar un robot adicional manteniendo un número constante de recolectores, no cambia significativamente para diferentes números de recolectores. Además, se concluyó que el impacto de añadir un recolector adicional es mayor que el impacto de añadir un robot hasta que hay suficientes recolectores para gestionar el sistema considerando el OAR dado.

- Para los mismos escenarios (Escenarios 1), pero respecto a la utilización de los agentes (RQ2), se observó que la Utilización Media de Recogedores (APU) aumenta ligeramente al aumentar el número de robots para un número fijo de recogedores, pero la APU disminuye considerablemente al añadir más recogedores, ya que más recogedores deben gestionar un número fijo de pedidos. Con los parámetros utilizados, se determinó que utilizando 2 recogedores la APU toma valores en torno al 60 % y 70 %.

- Siguiendo con la respuesta a la RQ2, los resultados también indicaron que la Utilización Media de Robots (UAR) no depende del número de recolectores, sino sólo del número de robots. La relación entre el APU y el número de robots es inversa: al añadir robots, el APU disminuye, ya que más robots deben gestionar un número plano de pedidos. Se ha visto que el APU toma valores en torno al 20 % y el 30 %. Por lo tanto, la utilización de los robots es muy baja para los escenarios observados.

- Sobre el impacto en el rendimiento del sistema pero ahora variando el OAR, el Número de Recogedores y el Número de Robots, mientras se fija el ratio entre el Número de Recogedores y el Número de Robots (RQ3), se observó que el AOTT aumenta al aumentar el OAR y mantener un número fijo de agentes, ya que los mismos agentes tienen que gestionar más pedidos. También se ha visto que este impacto es considerable, sobre todo en escenarios en los que el número de agentes no es suficiente para gestionar adecuadamente el número de pedidos, como el caso del escenario de 4 robots, 2 recolectores y 3 de Factor OAR, en el que el AOTT es casi la mitad de todo el tiempo de simulación dada la falta de capacidad de ese escenario. A medida que aumenta el número de agentes, el impacto de aumentar el OAR es menos importante.

- En los mismos Escenarios 2, pero con respecto a la utilización de los recolectores (RQ4), se encontró que tanto el número de agentes como el OAR son variables relevantes que afectan considerablemente al APU. Mientras que aumentando sólo el OAR y fijando el número de agentes aumenta el APU, por el contrario, añadiendo más agentes y manteniendo un OAR constante disminuye el APU. Se han encontrado tres escenarios en los que el APU es superior al 50 %, pero uno de ellos es el caso inviable destacado en el último párrafo. Los otros dos

son cuando el número de recolectores es 2, el número de robots es 4 y el Factor OAR es 2, y el otro cuando el número de recolectores es 3, el número de robots es 6 y el Factor OAR es 3.

- Por último, para completar la respuesta a la RQ4, analizando el ARU se comprobó que aumentar el OAR manteniendo constante el número de agentes implica un aumento considerable del ARU, es decir, para las diferentes cantidades de agentes, pasar del Factor OAR 1 al 3 duplica o triplica el ARU. Por otro lado, fijar el OAR, pero aumentar el número de agentes disminuye el ARU, pero el impacto no es tan fuerte como el impacto de cambiar el OAR. Por lo tanto, mantener una relación fija entre el número de recolectores y el número de robots produce un ARU estable. Entre todos los escenarios 2, el ARU oscila entre el 10 % y el 40 %.

8. Limitaciones

Este modelo, aunque flexible para afrontar investigaciones posteriores, se analizó sobre la base de varios supuestos y simplificaciones destinados a representar situaciones específicas que permitieran responder a las diversas Preguntas de Investigación. No obstante, estos supuestos y simplificaciones, así como el alcance de los análisis, dieron lugar a varias limitaciones. Las principales limitaciones identificadas que son interesantes para futuras investigaciones sobre este tema son:

- El modelo considera un conjunto específico de dimensiones del almacén, aunque el código fue escrito de forma que las dimensiones sean fácilmente modificables, en los análisis realizados el layout fue fijo. Muchas empresas pueden tener diferentes layouts o pueden estar evaluando la implementación de diferentes layouts para el picking de sus almacenes, por lo que se deben realizar análisis sobre el desempeño de este modelo en esos diferentes layouts.

- En este modelo se asumió que la reposición no era necesaria, por lo que la disponibilidad de los artículos está garantizada. Se trata de una gran simplificación que puede no ser cierta en algunos casos, por lo que incluir esta actividad en el modelo y cómo se llevaría a cabo (ya sea mediante una colaboración entre humanos y cobots, o solo por humanos) es interesante para nuevas investigaciones sobre este tema.

- También es interesante considerar las políticas de zonificación en las ubicaciones de los productos. Aunque una ubicación aleatoria de los artículos puede ser una simplificación justa para algunos almacenes que gestionan cierto tipo de productos, probablemente no sea una buena simplificación cuando existe una clara diferencia entre los productos almacenados, ya sean estos las diferencias se refieren al tamaño, el peso u otras características físicas de los

artículos, o incluso si las diferencias se refieren a las cantidades pedidas o a su valor relativo; en estos últimos casos puede utilizarse una política ABC. Todas estas consideraciones no se han tenido en cuenta en este estudio. El script diseñado para este estudio no ofrece la flexibilidad necesaria para incluir de forma sencilla estos escenarios.

- Para este estudio sólo se ha considerado la información sobre el rendimiento del sistema y los agentes, pero un resultado interesante sobre este tema sería un análisis coste-beneficio, considerando también la inversión necesaria para cada escenario y/o los costes operativos relacionados con cada escenario. El estudio realizado puede ser utilizado también como un primer paso para descartar escenarios y luego sólo utilizar algunos escenarios interesantes y añadir estos datos y realizar ya sea otra simulación más detallada u otro tipo de análisis, por ejemplo un Problema de Optimización incluyendo también los datos de Coste y Beneficio.

9. Anexo A: Código

9.1. Importar bibliotecas

```
import random
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Figura 47: Importar librerías.

9.2. Definición de parámetros

```
N = 45
simulation_time = 28800
width = 29
height = 15
avg_speed_AMR = 1
avg_speed_picker = 0.75
battery_size = 2.5
battery_consumption = 0.9 * battery_size / 4.5 / 60 / 60
power_charging_station = 0.9 * battery_size / 1.75
frequency_order_lines = [
    [1, 0.9542],
    [2, 0.0274],
    [3, 0.0054],
    [4, 0.0026],
    [5, 0.0012],
    [6, 0.0016],
    [7, 0.0005],
    [8, 0.0004],
    [9, 0.0067]
]
max_wait_order_limit = 25

layout = create_layout(width, height)
```

Figura 48: Definir parámetros.

9.3. Definición de funciones

```python
def create_layout(width, height):
    all_x = []
    for i in range(width+1):
        all_x.append(i)

    all_y = []
    for i in range(height + 1):
        all_y.append(i)

    a = 1
    x_picking_locations = []
    while a + 1 <= width:
        x_picking_locations.append(a)
        a += 3

    central_aisle1 = round(height/2) - 1
    central_aisle2 = round(height/2)
    central_aisles = [central_aisle1, central_aisle2]
    y_picking_locations = [0, 1, central_aisles[0], central_aisles[1], height - 1, height]

    depots = [[0, 0], [width, 0]]

    charging_stations = [[0, 0], [width, 0]]

    blocks_x = []
    for i in all_x:
        if i not in x_picking_locations:
            blocks_x.append(i)

    blocks_y = []
    for i in all_y:
        if i not in y_picking_locations:
            blocks_y.append(i)

    return [x_picking_locations, y_picking_locations, blocks_x, blocks_y, central_aisles, charging_stations, depots]
```

Figura 49: Definir funciones (1/7).

```python
def create_orders(arrival_rate_orders, x_picking_locations, y_picking_locations):
    time = 0
    orders = []
    id = 0
    while time <= simulation_time:
        prob = random.random()
        freq_acum = 0
        for n,f in frequency_order_lines:
            freq_acum += f
            if prob <= freq_acum:
                order_lines = n
                break
        picking_locations = []
        for i in range(order_lines):
            x = random.choice(x_picking_locations)
            y = random.choice(y_picking_locations)
            picking_locations.append([x, y])
        structure = [id, 0, time, picking_locations, order_lines, -1, 0, 0, 0, 0]
        orders.append(structure)
        time += int(np.random.exponential(1/arrival_rate_orders))
        id += 1
    orders.sort()
    return orders
```

Figura 50: Definir funciones (2/7).

```
def create_robots(num_robots):
    robots = []
    for i in range(num_robots):
        if i <= round(num_robots):
            inital_depot = [0,0]
        else:
            inital_depot = [width,0]
        structure = [i, 1, 0, [], 0, inital_depot, [], battery_size, -1, 0, 0, 0, 0, 0, 0, 0, 0]
        robots.append(structure)
    return robots
```

Figura 51: Definir funciones (3/7).

```
def create_pickers(num_pickers, layout):
    pickers = []
    for j in range(num_pickers):
        structure = [j, 0, -1, [random.choice(layout[0]), random.choice(layout[1])], [], 0, 0, 0, 0, 0]
        pickers.append(structure)
    return pickers
```

Figura 52: Definir funciones (4/7).

```
def calculate_distance(pos1, pos2):
    distance = abs(pos1[0] - pos2[0])
    if pos1[0] == pos2[0]:
        distance += abs(pos1[1] - pos2[1])
    elif (pos1[1] < layout[4][0] and pos2[1] < layout[4][0]) or (pos1[1] > layout[4][1] and pos2[1] > layout[4][1]):
        closest_aisle_y = min(layout[1], key = lambda z: abs(z - pos1[1]))
        distance += abs(pos1[1] - closest_aisle_y) + abs(pos2[1] - closest_aisle_y)
    else:
        distance += abs(pos1[1] - pos2[1])
    return distance
```

Figura 53: Definir funciones (5/7).

```
def next_position(pos, picking_locations, speed, flag):
    distances = []
    for x,y in picking_locations:
        if flag == True:
            if x + 1 in layout[0] or x == layout[0][-1]:
                x -= 1
            else:
                x += 1
        distance = calculate_distance(pos, [x,y])
        distances.append(distance)
    distance_min = min(distances)
    index_min = distances.index(distance_min)
    coord_min = picking_locations[index_min]
    travel_time = int(distance_min / speed)
    return [coord_min, travel_time]
```

Figura 54: Definir funciones (6/7).

```
def nearest_picker(pos, pickers, avg_speed_picker):
    travel_time = 100000
    id_picker = -1
    for picker in pickers:
        if picker[1] == 0:
            distance = int(calculate_distance(pos, picker[3]))
            candidate = int(distance / avg_speed_picker)
            if candidate <= travel_time:
                travel_time = candidate
                id_picker = picker[0]
    return id_picker
```

Figura 55: Definir funciones (7/7).

9.4. Preparación de la estructura de datos para el output

```
line_order = []
line_robot = []
line_picker = []

line_order.append('Order arrival factor')
line_order.append('Number robots')
line_order.append('Number pickers')
line_order.append('Run')
line_order.append('Number of orders')
line_order.append('Number of completed orders')
line_order.append('Waiting assignation time [s]')
line_order.append('Process time [s]')
line_order.append('Throughput time [s]')
line_order.append('Waiting movement time [s]')
info_order.append(line_order)

line_robot.append('Order arrival factor')
line_robot.append('Number robots')
line_robot.append('Number pickers')
line_robot.append('Run')
line_robot.append('Waiting order time total [s]')
line_robot.append('Charging time [s]')
line_robot.append('Travelling time [s]')
line_robot.append('Waiting picker time [s]')
line_robot.append('Picking time [s]')
info_robot.append(line_robot)

line_picker.append('Order arrival factor')
line_picker.append('Number robots')
line_picker.append('Number pickers')
line_picker.append('Run')
line_picker.append('Waiting robot time [s]')
line_picker.append('Travelling time [s]')
line_picker.append('Picking time [s]')
info_picker.append(line_picker)
```

Figura 56: Preparar el output.

9.5. Inicio de la simulación de los Escenarios 1

```
for NP in [2, 3, 4]:
    for NR in [6, 7, 8, 9, 10]:

        OR = 2
        arrival_rate_orders = OR * 80 / 60 / 60
        num_robots = NR
        num_pickers = NP
```

Figura 57: Iniciar simulaciones 1.

9.6. Inicio de la simulación de los Escenarios 2

```
for S in [[2, 4], [3, 6], [4, 8], [5,10]]:
    for OR in [1, 2, 3]:

        arrival_rate_orders = OR * 80 / 60 / 60
        NR = S[1]
        NP = S[0]
        num_robots = NR
        num_pickers = NP
```

Figura 58: Iniciar simulaciones 2.

9.7. Simulación

```
for r in range(N):

    line_order = []
    line_robot = []
    line_picker = []

    line_order.append(OR)
    line_order.append(NR)
    line_order.append(NP)
    line_order.append(r)

    line_robot.append(OR)
    line_robot.append(NR)
    line_robot.append(NP)
    line_robot.append(r)

    line_picker.append(OR)
    line_picker.append(NR)
    line_picker.append(NP)
    line_picker.append(r)

    orders = create_orders(arrival_rate_orders, layout[0], layout[1])
    robots = create_robots(num_robots)
    pickers = create_pickers(num_pickers, layout)
```

Figura 59: Simulación (1/7).

```
time = 0
count_orders = 0
while time < simulation_time:

    for order in orders:

        if order[1] == 0 and int(order[2]) <= int(time):
            order[1] = 1
            count_orders += 1

    for robot in robots:

        if robot[1] == 0:

            if int(robot[9]) <= int(time):
                robot[1] = 1
                robot[9] = int(time)

            else:
                robot[11] += 1
```

Figura 60: Simulación (2/7).

```
            if robot[1] == 1 and robot[2] < 3:

                for order in orders:

                    if order[1] == 2 and order[5] == robot[0]:
                        order[9] += 1

                    elif order[1] == 1:
                        order[1] = 2
                        order[5] = robot[0]
                        order[6] = int(time - order[2])
                        robot[2] += 1
                        robot[3].append(order[0])

                        for order_line in order[3]:

                            if order_line not in robot[6]:
                                robot[6].append(order_line)

                        if robot[2] == 3 or robot[16] >= max_wait_order_limit:
                            robot[1] = 2
                            robot[9] = int(time)
                            robot[16] = 0
                            break

                if robot[1] != 2:
                    robot[10] += 1
                    robot[16] += 1
```

Figura 61: Simulación (3/7).

```
if robot[1] == 5:

    if int(robot[9]) <= int(time):
        robot[4] += 1
        pickers[robot[8]][1] = 0
        pickers[robot[8]][2] = -1
        pickers[robot[8]][9] += 1
        robot[8] = -1

        if len(robot[6]) > 0:
            robot[1] = 2

        else:
            robot[1] = 6

    else:
        robot[14] += 1
        pickers[robot[8]][8] += 1

if robot[1] == 2:
    robot[1] = 3
    coord_min, travel_time = next_position(robot[5], robot[6], avg_speed_AMR, True)
    robot[5] = coord_min
    robot[6].remove(coord_min)
    robot[7] -= battery_consumption * travel_time
    robot[9] = int(time + travel_time)
```

Figura 62: Simulación (4/7).

```
if robot[1] == 3:

    if int(robot[9]) <= int(time):
        robot[1] = 4

    else:
        robot[12] += 1

if robot[1] == 4:

    if robot[8] == -1:
        id_picker = nearest_picker(robot[5], pickers, avg_speed_picker)

        if id_picker != -1:
            pickers[id_picker][1] = 1
            pickers[id_picker][2] = robot[0]
            coord_min, travel_time = next_position(pickers[id_picker][3], [robot[5]], avg_speed_picker, False)
            pickers[id_picker][3] = coord_min
            pickers[id_picker][5] = int(time + travel_time)
            robot[8] = pickers[id_picker][0]
            robot[9] = int(time + travel_time)

        else:
            robot[13] += 1
```

Figura 63: Simulación (5/7).

```
    if robot[1] == 4 and robot[8] != -1:

        if int(pickers[robot[8]][5]) <= int(time):
            pickers[robot[8]][1] = 2
            pickers[robot[8]][8] += 1
            picking_time = int(random.randint(2, 10))
            pickers[robot[8]][5] = int(time + picking_time)
            robot[1] = 5
            robot[14] += 1
            robot[9] = int(time + picking_time)

        else:
            robot[13] += 1
            pickers[robot[8]][7] += 1

    if robot[1] == 6:
        robot[1] = 7
        robot[2] = 0
        coord_min, travel_time = next_position(robot[5], layout[6], avg_speed_AMR, False)
        robot[5] = coord_min
        robot[7] -= int(battery_consumption * travel_time)
        robot[9] = int(time + travel_time)
```

Figura 64: Simulación (6/7).

```
    if robot[1] == 7:

        if int(robot[9]) <= int(time):

            for o in robot[3]:
                orders[o][1] = 3
                orders[o][7] = time - orders[o][2] - orders[o][6]
                orders[o][8] = time - orders[o][2]
                robot[15] += 1

            robot[3] = []
            robot[1] = 0
            charging_time = int((battery_size - robot[7]) / power_charging_station * 60 * 60)
            robot[7] = battery_size
            robot[9] = int(time + charging_time)
            robot[11] += 1

        else:
            robot[12] += 1

for picker in pickers:

    if picker[1] == 0:

        if picker[5] <= time:
            picker[6] += 1

time += 1
robots.sort(key = lambda x: x[9])
```

Figura 65: Simulación (7/7).

9.8. Exportar el resultado

```
aux_orders = []
done_orders = 0

for order in orders:

    if order[1] == 3:
        aux_orders.append([order[6], order[7], order[8], order[9]])
        done_orders += 1

array_orders = np.array(aux_orders)
line_order.append(len(orders))
line_order.append(done_orders)
line_order.append(array_orders[:, 0].mean())
line_order.append(array_orders[:, 1].mean())
line_order.append(array_orders[:, 2].mean())
line_order.append(array_orders[:, 3].mean())
info_order.append(line_order)

aux_robots = []
for robot in robots:
    aux_robots.append([robot[10], robot[11], robot[12], robot[13], robot[14]])

array_robots = np.array(aux_robots)
line_robot.append(array_robots[:, 0].mean())
line_robot.append(array_robots[:, 1].mean())
line_robot.append(array_robots[:, 2].mean())
line_robot.append(array_robots[:, 3].mean())
line_robot.append(array_robots[:, 4].mean())
info_robot.append(line_robot)
```

Figura 66: Exportar output (1/2).

```
            aux_pickers = []
            for picker in pickers:
                aux_pickers.append([picker[6], picker[7], picker[8]])

            array_pickers = np.array(aux_pickers)
            line_picker.append(array_pickers[:,0].mean())
            line_picker.append(array_pickers[:,1].mean())
            line_picker.append(array_pickers[:,2].mean())
            info_picker.append(line_picker)

            print('Run', r, 'with scenario:')
            print('Order factor =', OR)
            print('Number of robots =', NR)
            print('Number of pickers', NP)
            print('Done\n')

df1 = pd.DataFrame(np.array(info_order))
df2 = pd.DataFrame(np.array(info_robot))
df3 = pd.DataFrame(np.array(info_picker))

df1.to_excel('df1_v3.xlsx')
df2.to_excel('df2_v3.xlsx')
df3.to_excel('df3_v3.xlsx')
```

Figura 67: Exportar output (2/2).

10. Anexo B: Resultados

10.1. Escenarios 1

10.1.1. Indicadores de pedidos

Tabla 6: Output de los pedidos (Escenarios 1)

NP	NR	AOWAT [s]	APT [s]	AOTT [s]	AWMT [s]
2	6	2	111	113	6
	7	1	104	105	4
	8	0	96	97	2
	9	0	94	94	2
	10	0	93	93	1
3	6	0	79	80	4
	7	0	72	73	2
	8	0	68	69	1
	9	0	66	66	1
	10	0	65	65	0
4	6	0	71	71	4
	7	0	65	65	2
	8	0	61	61	1
	9	0	59	59	0
	10	0	58	58	0
Totals		0	78	78	2

10.1.2. Indicadores del recolector

Tabla 7: Output de los recolectores (Escenarios 1)

NP	NR	AWRT [s]	APTT [s]	APPT [s]
2	6	10454	14023	4323
	7	10115	14326	4360
	8	10129	14327	4344
	9	9916	14484	4400
	10	9753	14646	4402
3	6	18707	7199	2894
	7	18608	7310	2882
	8	18366	7520	2913
	9	18283	7598	2920
	10	18204	7674	2921
4	6	22572	4054	2174
	7	22454	4167	2178
	8	22421	4203	2176
	9	22386	4241	2173
	10	22374	4257	2169
Totals		16983	8669	3149

10.1.3. Indicadores del robot

Tabla 8: Output de los robots (Escenarios 1)

NP	NR	AWOT [s]	ACT [s]	ARTT [s]	AWPT [s]	ARPT [s]
2	6	12345	1602	6838	6574	1441
	7	13694	1440	6253	6168	1246
	8	15059	1298	5714	5644	1086
	9	16008	1189	5251	5373	978
	10	16917	1075	4772	5155	880
3	6	14704	1673	7247	3729	1447
	7	16252	1488	6539	3286	1235
	8	17465	1335	5909	2999	1093
	9	18527	1205	5360	2734	973
	10	19449	1094	4874	2507	876
4	6	15557	1698	7387	2709	1449
	7	17020	1506	6638	2391	1245
	8	18266	1348	5984	2114	1088
	9	19368	1204	5364	1898	966
	10	20292	1083	4836	1721	868
Totals		16728	1349	5931	3667	1125

10.2. Escenarios 2

10.2.1. Indicadores de pedidos

Tabla 9: Output de los pedidos (Escenarios 2)

NR	NP	OAR Factor	AOWAT [s]	AWMT [s]	APT [s]	AOTT [S]
4	2	1	1	6	82	83
		2	13	9	132	146
		3	1367	0	175	1542
6	3	1	0	1	61	61
		2	0	4	80	80
		3	4	6	114	118
8	4	1	0	0	56	56
		2	0	1	61	61
		3	0	3	77	77
10	5	1	0	0	54	54
		2	0	0	55	55
		3	0	1	61	61
Totals			116	3	84	199

10.2.2. Indicadores del recolector

Tabla 10: Output de los recolectores (Escenarios 2)

NR	NP	OAR Factor	AWRT [s]	APTT [s]	APPT [s]
4	2	1	20508	6128	2164
		2	11006	13442	4352
		3	3317	19604	5879
6	3	1	24265	3093	1442
		2	18719	7168	2913
		3	12181	12263	4356
8	4	1	25882	1829	1089
		2	22370	4238	2192
		3	18196	7299	3305
10	5	1	26744	1191	865
		2	24395	2673	1732
		3	21602	4572	2626
Totals			19099	6958	2743

10.2.3. Indicadores del robot

Tabla 11: Output de los robots (Escenarios 2)

NR	NP	OAR Factor	AWOT [s]	ACT [s]	ARTT [s]	AWPT [s]	ARPT [s]
4	2	1	17663	1274	5552	3229	1082
		2	8157	2065	8249	8154	2176
		3	161	2380	8707	14614	2939
6	3	1	21622	897	4004	1556	721
		2	14694	1677	7258	3715	1456
		3	8721	2169	8843	6889	2178
8	4	1	23627	680	3033	915	545
		2	18207	1355	6009	2132	1096
		3	13300	1899	8212	3736	1653
10	5	1	24790	545	2437	595	432
		2	20667	1086	4844	1337	866
		3	16458	1609	7126	2294	1313
Totals			15672	1470	6190	4097	1371

Referencias

Adriaensen, A. and B., N. (2022). Interdependence analysis in collaborative robot applications from a joint cognitive functional perspective. *International Journal of Industrial Ergonomics*.

Aloini, D. F. (2022). Enhancing operations management through smart sensors: measuring and improving well-being, interaction and performance of logistics workers. *The TQM Journal*, 34(2):303–329.

Bansal, V. R. (2021). Performance analysis of batching decisions in waveless order release environments for e-commerce stock-to-picker order fulfillment. *International Transactions in Operational Research*, 28:1787–1820.

Dehkordi, M. B. and M., R. (2021). Explainability in human-robot teaming. In *Procedia Computer Science*, pages 3487–3496.

Ghorashi Khalilabadi, M., Roy, D., and de Koster, R. (2022). Human-robot matching policies in order-picking systems using queuing models. In *Presentación académica*. El año es una estimación basada en el contenido y las referencias del documento, ya que no se especifica explícitamente.

Guerra, A. (2022). Centralized or decentralized system? analyzing intelligent order picking warehouses with autonomous mobile robots. Master's thesis, Politecnico di Milano, Milan, Italy. Tesis de Laurea Magistrale en Management Engineering.

Jacob, F. and H., E. (2023). Picking with a robot colleague: A systematic literature review and

evaluation of technology acceptance in human–robot collaborative warehouses. *Computers Industrial Engineering*.

Kudelska, I. P. (2020). Influence of assortment allocation management in the warehouse on the human workload. *Central European Journal of Operations Research*, 28:779–795.

Kumar, S. and Sheu, J.-B. (2023). Planning a parts-to-picker order picking system with consideration of the impact of perceived workload. *Transportation Research Part E: Logistics and Transportation Review*.

Lambrechts, W., Klaver, J., Koudijzer, L., and Semeijn, J. (2021). Human factors influencing the implementation of cobots in high volume distribution centres. *Logistics*, 5:32.

Lorson, F. and F., A. (2023). New team mates in the warehouse: Human interactions with automated and robotized systems. *IISE Transactions*, 55(5):536–553.

Loske, D. Empirical evidence on human learning and work characteristics in the transition to automated order picking. *Journal of Business Logistics*, page 302–342.

Maruyama, T., Ueshiba, T., Tada, M., Toda, H., Endo, Y., Domae, Y., and Suita, K. (2021). Digital twin-driven human robot collaboration using a digital human. *Sensors*, 21:8266.

Menti, F. and R., D. (2023). A technology assessment and implementation model for evaluating socio-cultural and technical factors for the successful deployment of logistics 4.0 technologies. *Technological Forecasting and Social Change*.

Pasparakis, A. and D., A. (2021). In control or under control? human-robot collaboration in wa-

rehouse order picking. http://dx.doi.org/10.2139/ssrn.3816533. SSRN preprint.

Pasparakis, A. and D., A. (2023). Assessing the impact of human–robot collaborative order picking systems on warehouse workers. *International Journal of Production Research*, 61(22):7776–7790.

Peruzzini, M. and Pellicciari, M. (2017). A framework to design a human-centred adaptive manufacturing system for aging workers. *Advanced Engineering Informatics*, 33:330–349.

Ross, S. (2007). Introduction to probability models. *Academic Press*.

Sheu, J.-B. and Chou, M.-L. (2023). Can we work more safely and healthily with robot partners? a human-friendly robot–human-coordinated order fulfillment scheme. *Production and Operations Management*, 32:794–812.

Sobhani, V. V. (2023). Performance optimisation of pick and transport robot in a picker to parts order picking system: a human-centric approach. *International Journal of Production Research*, 61(22):7791–7808.

Thylén, N. and W., C. (2023). Challenges in introducing automated guided vehicles in a production facility – interactions between human, technology, and organisation. *International Journal of Production Research*, 61(22):7809–7829.

Winkelhaus, S. and Zimmer, M. (2022). Hybrid order picking: A simulation model of a joint manual and autonomous order picking system. *Computers Industrial Engineering*.

Zhang, M. and W., S. (2021). Evaluation of human workload in a hybrid order picking system. *IFAC-PapersOnLine*, pages 458–463.