UNIVERSITAT POLITÈCNICA DE VALÈNCIA

FACULTAD DE ADMINISTRACIÓN Y DIRECCIÓN DE EMPRESAS. UPV

Facultad de Administración y Dirección de Empresas
Universidad Politécnica de Valencia

Operaciones de localización de contenedores: desarrollo de métodos para problemas de localización de contenedores en terminales portuarias

TRABAJO FINAL DE GRADO

Grado en Administración y Dirección de Empresas

Autor: Jorge Feliu Escagüés

Tutoras: Eva Vallada Regalado
Mª Fulgencia Villa Juliá

Curso 2019-2020

Resumen

El transporte marítimo asume un papel fundamental a nivel mundial en el comercio y su volumen crece año tras año. Este aumento de los flujos de comercio provoca que los puertos deban hacer frente a unos niveles de trabajo cada vez mayores.

Con el fin de mejorar la productividad de las terminales y para hacer frente a una carga de trabajo que aumenta progresivamente, las terminales buscan alternativas que les ayuden a ser más competitivas y productivas. Una forma habitual de lograr unos mejores rendimientos consiste en diseñar métodos capaces de optimizar las actividades que se realizan dentro del puerto. Una de estas actividades, la localización de contenedores, es capaz de mejorar notablemente toda la cadena de actividades si se gestiona adecuadamente.

Mediante este trabajo, se ofrece una visión de conjunto de los puertos y de las principales operaciones e infraestructuras de una terminal de contenedores, exponiendo en profundidad el problema de localización de contenedores. También se proponen distintos algoritmos con los que resolver este problema. A través de los métodos propuestos se busca obtener soluciones que minimicen los tiempos necesarios para la ubicación de contenedores. Por último, se realiza una serie de experimentos computacionales para evaluar la eficiencia de estos algoritmos.

Palabras clave: contenedores, heurísticas, Investigación Operativa, localización, optimización, puerto

---

Resum

El transport marítim assumeix un paper fonamental a nivell mundial en el comerç i el seu volum creix any rere any. Aquest augment dels fluxos de comerç provoca que els ports hagen de fer front a uns nivells de treball cada vegada majors.

Amb la finalitat de millorar la productivitat de les terminals i per a fer front a una càrrega de treball que augmenta progressivament, les terminals busquen alternatives que els ajuden a ser més competitives i productives. Una forma habitual d'aconseguir uns millors rendiments consisteix a dissenyar mètodes capaços d'optimitzar les activitats que es realitzen dins del port. Una d'aquestes activitats, la localització de contenidors, és capaç de millorar notablement tota la cadena d'activitats si es gestiona adequadament.

Mitjançant aquest treball, s'ofereix una visió de conjunt dels ports i de les principals operacions i infraestructures d'una terminal de contenidors, exposant en profunditat el problema de localització de contenidors. També es proposen diferents algorismes amb els quals resoldre aquest problema. A través dels mètodes proposats es busca obtindre solucions que minimitzen els temps necessaris per a la ubicació de contenidors. Finalment, es realitza una sèrie d'experiments computacionals per a avaluar l'eficiència d'aquests algorismes.

Paraules clau: contenidors, heurístiques, Investigació Operativa, localització, optimització, port

---

Abstract

Maritime transport assumes a fundamental role worldwide in trade and its volume grows year after year. This increase in trade flows means that ports have to face increasing levels of work.

In order to improve the productivity of the terminals and to face a workload that is progressively increasing, the terminals are looking for alternatives that help them to be more competitive and productive. A common way to achieve better performance is to design methods capable of optimizing the activities carried out within the port. One of these activities, the location of containers, is capable of significantly improving the entire chain of activities if properly managed.

Through this work, an overview of the ports and the main operations and infrastructures of a container terminal is offered, exposing in depth the problem of container location. Different algorithms with which to solve this problem are also proposed. Through the proposed methods, we seek to obtain solutions that minimize the time necessary for the location of containers. Finally, a series of computational experiments are performed to evaluate the efficiency of these algorithms.

Key words: containers, heuristics, Operational Research, location, optimization, port

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

Los puertos marítimos son el motor de la economía mundial. Con un control de más del 80 % del volumen de todo el comercio que se realiza a nivel mundial, y con unas expectativas de crecimientos en alza, los puertos juegan un papel clave en el comercio internacional.

Organizar, gestionar y optimizar todo el tráfico de mercancías es un reto fundamental al que tienen que enfrentarse diariamente los trabajadores del puerto. La competitividad de los puertos depende de multitud de variables, como los costes, la eficiencia o la calidad de los servicios que ofrece. Por lo tanto, ser capaces de minimizar los tiempos que tardan en realizar sus operaciones, puede traducirse como una mejora en la productividad de la terminal.

Asignar los muelles a los buques, elaborar planes de estiba o almacenar la mercancía son algunas de estas operaciones que se realizan a diario en todos los puertos. El problema de la localización de contenedores en un bloque, forma parte de esta cadena de operaciones y, gestionarlo adecuadamente, tiene un gran impacto en la eficiencia del puerto.

Los contenedores llegan desde los barcos y desde los camiones y trenes para ser almacenados dentro de los patios de las terminales. Después, las grúas de patio cogen cada contenedor y lo ubican en aquellos espacios que le hayan asignado según sus características. Esta actividad, que a priori parece sencilla, ha sido un tema recurrente de estudio en las últimas décadas, planteando diversos algoritmos y metodologías con los que cubrir el problema.

1.1 Objeto y objetivos

El objeto del presente Trabajo de Final de Grado consiste en presentar cómo funciona una terminal portuaria de contenedores, centrándose específicamente en la etapa de localización de contenedores en un bloque, y diseñar algoritmos y métodos que aporten soluciones factibles dentro de unos rangos de tiempo aceptables.

El objetivo principal será definir el problema de la localización de contenedores dentro de un bloque y formular distintas aproximaciones algorítmicas mediante las cuales sea posible minimizar el tiempo necesario para realizar la localización, así como identificar el orden que se sigue para localizar cada contenedor en la posición asignada. También se analizan los algoritmos propuestos y se comparan los resultados obtenidos por cada uno de ellos.

Para lograr estos objetivos, se enunciarán los principales puntos a seguir en la planificación del trabajo:

- Ilustrar la historia y la situación actual del comercio y transporte marítimo, sirviendo como punto de partida para conocer el funcionamiento de la terminal de contenedores.
- Identificar las principales áreas de una terminal portuaria, el equipamiento e infraestructura que se utiliza dentro de esta y establecer las operaciones que se encarga de realizar una terminal de contenedores.
- Definir el problema de la localización de contenedores dentro de un bloque y establecer los problemas que surgen a la hora de realizar estos movimientos de contenedores portuarios, así como proponer soluciones a estos.
- Desarrollar e implementar distintas aproximaciones algorítmicas buscando mejorar los tiempos de operación para la localización de los contenedores dentro del bloque.
- Evaluar e interpretar los resultados que proporciona cada algoritmo y establecer cuáles son aquellos capaces de ofrecer una solución factible dentro de un rango temporal adecuado.
- Relacionar las conclusiones extraídas con el paradigma actual y establecer posibles líneas de investigación futuras referentes al problema de localización de contenedores dentro del bloque.

1.2 Motivación

Los problemas de localización de contenedores son un tema relevante dentro de las operaciones que se realizan en las terminales portuarias. En los últimos años, gran cantidad de estos problemas han sido tratados en profundidad y se han propuesto diversas aproximaciones con las que hacer frente a estos. Administrar de manera correcta cómo y dónde se colocarán los contenedores dentro del puerto, ofrece ciertas ventajas que permiten al puerto ahorrar recursos y agilizar sus operaciones.

Conocer cómo funciona una terminal de contenedores y qué entraña la actividad de localizarlos, ayuda a entender mejor la relación que existe entre el puerto y el comercio y, cómo es posible mejorar los niveles de eficiencia de las terminales mediante la reducción de los tiempos asignados a operar sobre los contenedores.

Tras estudiar diversos artículos relacionados con las terminales portuarias, se ha podido establecer como, especialmente a lo largo de estas últimas décadas, ha ido creciendo el interés por la optimización de las operaciones portuarias. Este tema supone un campo clave sobre el que realizar futuras investigaciones debido al impacto económico que implica una correcta gestión de la localización. Conocer cómo podemos desarrollar métodos para ahorrar recursos resulta interesante y útil, ya que esta clase de problemas de optimización no solo aparecen en los puertos, sino en prácticamente cualquier lugar.

El interés de este problema reside en conocer qué es el problema de localización de contenedores y demostrar el potencial que tiene en la actualidad, en un contexto en el que la tecnología tiene una influencia cada vez mayor en todas las operaciones que se realizan en los puertos.

El estudio de la optimización de recursos de las terminales portuarias puede relacionar estas operaciones con los Objetivos de Desarrollo Sostenible. Mediante las innovaciones y avances tecnológico que se aplican en los puertos, se promueve una industriali-

zación inclusiva y actual. Además, también contribuyen positivamente a una mejora del ecosistema acuático y la calidad de las aguas, gracias al uso de energías limpias.

Desde un punto de vista académico, este trabajo profundiza en conceptos estudiados durante la carrera de Administración y Dirección de Empresas. Una de las bases sobre las que se sustenta el trabajo es aquella relacionada con el campo de la Investigación Operativa, ya que aborda la resolución del problema aplicando la metodología y técnicas estudiadas en esta asignatura además de utilizar herramientas como el *Solver*.

Las nociones aprendidas en asignaturas como Econometría o Estadística también han resultado importantes. Gracias a ellas, se ha podido estudiar los rendimientos de los distintos métodos y realizar análisis de las desviaciones porcentuales relativas. También se han empleado programas estadísticos como el Statgraphics, empleados en estas asignaturas.

Las bases establecidas en asignaturas como Economía Mundial y Economía Española también han servido para relacionar conceptos del comercio marítimo, el peso que tiene en el comercio internacional, la globalización o el papel que ejercen los distintos países dentro del transporte marítimo.

1.3 Estructura del documento

La organización de este Trabajo de Final de Grado se especificará a continuación con el fin de proporcionar una idea general al lector, así como ayudarle a localizar los distintos apartados:

- En el presente capítulo 1 se exponen los objetivos generales y específicos de este trabajo, cuál ha sido la motivación que ha llevado a plantear este tema y qué estructura sigue el trabajo.
- Un capítulo 2 en el que se expone el contexto histórico del transporte marítimo, se introduce el concepto de los contenedores marítimos y se expone una visión general de la situación del comercio marítimo en la actualidad.
- El capítulo 3 define el concepto de puerto marítimo y los equipos e infraestructuras que lo componen y recoge las principales actividades y procedimientos que se realizan dentro de las instalaciones de un puerto. También muestra los problemas que surgen en cada una de sus partes.
- En el capítulo 4 se presenta el problema de la localización de contenedores: se realiza una descripción del problema, se muestra una versión simplificada de su formulación matemática y se diseñan y desarrollan distintos algoritmos para la aproximación a la resolución del problema.
- El capítulo 5 recoge las principales fuentes bibliográficas sobre las que se basa el trabajo y cómo se relacionan con los distintos apartados.
- Por último, el capítulo 6 expone la conclusión extraída en lo referente al problema de localización y se muestran posibles líneas de investigación futuras hacia las que puede enfocarse el problema de localización de contenedores.

CAPÍTULO 2
Contexto histórico y situación actual del transporte marítimo

El transporte marítimo ha jugado un papel fundamental no solo en el comercio internacional, sino en el desarrollo cultural y social de la civilización desde principios de los tiempos. Dada la relevancia que han tenido a lo largo de la historia algunos conceptos como *puerto* o *comercio marítimo*, resultará útil enmarcar el contexto histórico y los antecedentes principales que han propiciado la trascendencia de dichos conceptos.

2.1 Historia

Cuando se habla de las civilizaciones antiguas, puede observarse que todas ellas presentan un nexo común: su proximidad con el agua. El hecho de que Egipto estuviese bañada por el río Nilo y el Mediterráneo, que Mesopotamia estuviese entre los ríos Éufrates y Tigris o que los pilares de China se asentasen junto al río Huang He es probablemente una de las causas fundamentales de su prosperidad. Puede considerarse que estos puertos primigenios, datados en períodos anteriores al siglo X a.C., fueron los primeros centros neurálgicos de estas civilizaciones. Estas civilizaciones fueron las pioneras del transporte marítimo mediante aquellos primitivos navíos. Este flujo, tanto de mercancías como de personas, supuso el nacimiento del comercio entre las distintas culturas: este intercambio económico, social y cultural supuso un punto clave en el desarrollo de la humanidad.

A lo largo de los siguientes siglos, el papel que jugaban los barcos y los puertos era fundamental en todos los niveles, por lo que su mejora y evolución estaba estrechamente ligada con la capacidad estratégica. Durante la Edad Media, surgieron muchos núcleos de población junto a los principales puertos, los barcos experimentaron diversas mejoras de diseño y se inventaron instrumentos como la brújula. Todo esto encaminó a la civilización hacia la época de las exploraciones, a principios del siglo XV. Es en esta época cuando surgen las grandes exploraciones hacia Oriente y América, unos hechos que provocarán una expansión inmensa de las ya existentes redes comerciales que existían en Europa junto con el establecimiento de unas doctrinas mercantilistas. En relación con las instalaciones portuarias, destacan las dificultades que presentaban los puertos para abastecer un número cada vez mayor de barcos y que se reflejará en unas primeras inversiones en las infraestructuras de estos.

EL siguiente hito que marcará el desarrollo marítimo será la invención de la máquina de vapor hacia finales del siglo XVIII, cuya incorporación en los barcos cambiaría radicalmente el transporte marítimo. Gracias a ella, los nuevos navíos disponían de un gran

tonelaje, mayores dimensiones y unas velocidades nunca antes vistas. La ingeniería naval siguió perfeccionando tanto puertos como buques optimizando cada vez más todas estas características.

Durante la primera mitad del siglo XX, ocurrió una gran especialización en el sector portuario y aparecieron los buques de tamaños inmensos (tanto de carácter militar como comercial). Aunque ahora los barcos podían transportar cantidades ingentes de carga, el proceso de carga y descarga de estos barcos era tedioso y generaba grandes retrasos en los puertos. En la Primera y Segunda Guerra Mundial, el papel de la ingeniería naval fue fundamental. Se revolucionó el concepto y el diseño de la construcción de buques, y surgieron numerosos buques militares con unas características que no se habían visto hasta ese momento.

Tras la Segunda Guerra Mundial y empujado por un sentimiento de globalización que había resurgido gracias a la creación de diversas organizaciones mundiales y la fortaleza de las tecnologías de la comunicación que comenzaban a asentarse, el comercio marítimo juega un papel fundamental. Sin embargo, aún quedaba por encontrar una solución que permitiese descongestionar los puertos y que terminase con la ineficacia en aquel momento.

2.2 Contenedores

Con la aparición de los contenedores y la proliferación de su uso, el transporte marítimo experimentó un cambio radical. Anteriormente, la carga de los barcos se distribuía en todo tipo de cajones, bidones o recipientes variados, con un grado de estandarización considerablemente bajo y que ralentizaba el hecho de cargar y descargar, unidad a unidad, unos barcos que cada vez podían transportar más carga. Pero gracias al uso de los contenedores, se solventaron gran parte de los problemas. La pionera en emplear unos contenedores que pudiesen ser transportados por barcos fue la empresa de Malcom McLean a principios de 1956. McLean llegó a la conclusión de que *«un barco sólo gana dinero cuando está en el mar»*, y basándose en esta premisa, adaptó sus buques para que pudiesen transportar remolques apilados. El rotundo éxito logrado en estos primeros viajes con mercancía contenedorizada estableció el camino por el que se desarrollaría la industria naval.

Aunque con estos primeros contenedores se conseguía reducir el coste de la carga en torno al 16 % por cada tonelada que se cargaba y se lograba reducir el espacio desaprovechado incluso en un tercio, su implementación fue un proceso lento que se extendió hasta principios de los años sesenta. Con unos barcos que eran capaces de cargar más de 200 de estos contenedores, aún no había un modelo de contenedor unificado. Las grandes inversiones en infraestructuras como grúas era algo necesario para aprovechar la carga de contenedores, y dado el desembolso y el espacio que suponían, el desarrollar un modelo único de contenedor era fundamental. Los primeros modelos de estas nuevas grúas adaptadas al movimiento de contenedores permitían operar con 400 toneladas de contenedores por hora, productividad que llegaba a ser cuarenta veces superior a la que presentaban de media las grúas tradicionales [21].

La clave para automatizar el proceso de carga y descarga residía en un contenedor estandarizado. Una primera aproximación vino de manos de la Junta Marítima Federal de Estados Unidos en 1961, que ofrecía subsidios en la fabricación de contenedores de 10, 20, 30 y 40 pies de largo, 8 pies de ancho y 8 pies de alto. En 1968 estas medidas se normalizan gracias a la norma ISO 668 sobre la categorización de estos.

Gracias a este contexto de estandarización que se había establecido, surgen los **TEU**, *Twenty-Foot Equivalent Units*. Este término es la medida estándar que se emplea en el transporte marítimo a nivel mundial y hace referencia a un contenedor de 20 pies de largo, 8 de ancho y 8 alto: el tamaño de contenedor más popular. Sus dimensiones en el sistema internacional equivalen a aproximadamente 6 x 2,8 x 2,8 metros y pueden admitir un carga de hasta 28 toneladas. Los TEU y los 2 TEU (equivalente a los contenedores de 40 pies de largo) son el tamaño en el que se basan los conceptos actuales de capacidad y medida de los buques.

El tonelaje de los primeros barcos que comenzaron a adoptar la medida TEU en los años sesenta era de 1.500 TEU y la tendencia no ha hecho más que crecer. A mediados de los años ochenta, este tonelaje se había triplicado hasta un tonelaje de unos 6.000 TEU; en la actualidad se tienen buques como el *MSC Gülsün* (figura 2.2) que alcanzan una capacidad de 23.000 TEU y de las previsiones futuras indican que esto seguirá creciendo con el tiempo. Este aumento del tamaño de los buques supone un reto para los puertos más competitivos que tratan de dar servicio a los buques más grandes, y estará estrechamente ligado al desarrollo de las infraestructuras y la expansión de los puertos.

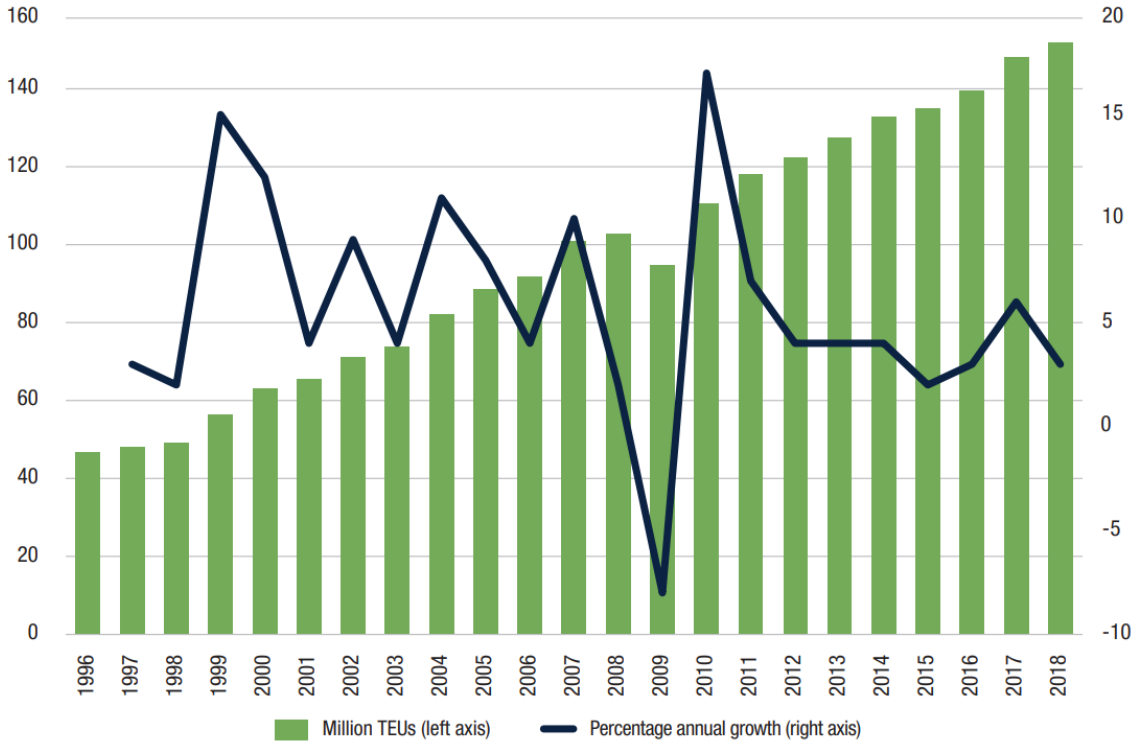


Figura 2.1: Evolución del comercio global de contenedores entre 1996 y 2018.
Figura de la UNCTAD [28], basado en datos de World Cargo Database, mayo 2019.

Mediante la figura 2.1 puede visualizarse un constante crecimiento del flujo de los contenedores TEU comerciados, con una evolución porcentual positiva a lo largo de todos los años excepto en el año 2009. Las previsiones de la UNCTAD[1] sobre la evolución del crecimiento del trafico de contenedores global se encontraban sobre el 4,7 %, aunque debido a la situación provocada por el COVID-19 se estima que sufrirá una evolución similar a la del año 2009, debido al estancamiento de gran parte de las actividades comerciales a nivel mundial, provocando un aumento del flujo de contenedores vacíos.

[1]Conferencia de las Naciones Unidas sobre Comercio y Desarrollo.

Figura 2.2: Buque portacontenedores MSC Gülsün.
Uno de los mayores portacontenedor de última generación del mundo. Figura capturada por Carlos Duclos.

Los contenedores pueden clasificarse según tres tipos: el contenedor de importación, que es aquel que llega por la parte de mar y es transportado hacia las áreas donde comunican con los transportes terrestres; el contenedor de exportación, que funciona al contrario que el de importación, llega desde tierra y es transportado hacia la parte del mar; y el contenedor de transbordo o *transshipment*, que es aquel que se carga y descarga en la parte marítima de la terminal. Los contenedores vacíos o que portan algún producto que tiene cierto riesgo se suelen tratar de forma distinta a estos tipos principales y se les almacena en lugares reservados especialmente para estos.

2.3 Situación actual

Durante los últimos años, el comercio marítimo internacional ha evolucionado de manera notable y gran parte del comercio internacional se basa en el uso de barcos y puertos para intercambiar productos. El transporte marítimo supone alrededor del 80-85 % de todo el tráfico internacional. Siguiendo la tendencia de una cierta ralentización del PIB a nivel mundial, el crecimiento de las importaciones y exportaciones cayó en el año 2018 un 2,8 %, situándolo ligeramente por debajo de lo esperado si se compara con el crecimiento del 4,5 % que experimentó en el 2017. Se estimaba que el volumen de este creciese con una tasa anual del 3,5 % entre los años 2019 y 2024 [27], aunque debido a la inestabilidad provocada por la pandemia en el 2020, se espera que estas estimaciones mermen considerablemente a partir de dicho año.

Los principales flujos de comercio marítimo se encuentran en los países en desarrollo con un 58,8 % del total de las exportaciones y un 64,5 % de las importaciones, concentrándose principalmente en Asia, donde China y los países adyacentes, juegan un papel destacable dentro de las cadenas de valor mundiales. Por otra parte, los países desarro-

llados están experimentando una disminución en su tráfico tanto de importaciones como de exportaciones. Las tendencias del año 2019 muestran un mercado en el que se aprecia el auge del comercio de contenedores, aunque más lento que años anteriores, mientras que la carga seca tanto de graneles[2] principales como secundarios se ha expandido en un 2,6 % en ese mismo año, un ritmo inferior al presentado en 2018 (4 %) [28].

El crecimiento del tráfico marítimo de los últimos años se apoya en la magnitud que están generando las rutas transpacíficas e intra-asiáticas, que resaltan el papel clave que tiene el continente asiático en el desarrollo del negocio marítimo. Este dominio asiático también se aprecia comparando el volumen de contenedores TEU que manejan sus puertos concentrando un 65 % del flujo total, mientras que los puertos de Europa, con un 25 % del volumen se sitúan en segunda posición. Puertos como el de Shanghái (figura 2.3), Singapur o Hong Kong son líderes a nivel mundial y dentro de los 20 puertos que más tráfico generan a nivel mundial tan solo se encuentran cinco fuera de Asia, como la terminal de Hamburgo, Los Ángeles o Róterdam.

Figura 2.3: Terminal de Yangshan, puerto de Shanghái.
Figura de Rib42, vía Wikimedia Commons.

La tendencia hacia una producción y unos mercados globalizados ha supuesto un gran desafío para el desarrollo del comercio marítimo. Todos aquellos puertos que buscan situarse en una posición competitiva dentro de este mercado global, han tenido que experimentar una serie de cambios para adaptarse a este nuevo contexto. Han tenido que realizar desde inversiones en nuevas infraestructuras para las terminales, hasta cambiar la gestión y organización del propio puerto. La idea de los puertos tradicionales, con unos sistemas más clásicos y manuales, ha virado hacia los *smart ports*, una nueva forma de afrontar los problemas que surgen en los puertos en este contexto de globalización apoyándose en soluciones como el *Internet of Things* o en el *Big Data*.

Mediante esta interconexión de la cadena logística que aportan los *smart ports* así como la automatización de operaciones en los puertos, los puertos consiguen mejores ren-

[2]Mercancías que se transportan sin embalar, como el grano, el hierro o el carbón.

dimientos y presentan mayor competitividad. Todo esto se fundamenta en las dos bases [12] que presenta el concepto de los *smart ports*:

- Una interconexión basada en el intercambio de datos. Se apoya en una informatización de los sistemas de los puertos que agilice y asegure las operaciones. Gracias a esta informatización se consigue recopilar datos relevantes a la hora de optimizar y flexibilizar la estructura.

- La automatización de la infraestructura y las operaciones, que además de ofrecer unos rendimientos superiores a los conseguidos en un puerto tradicional, minimiza el gasto de recursos innecesario. Esta automatización se basa en monitorizar los contenedores gracias a la implementación de sensores que indiquen las características de estos [31] a la vez que permiten realizar trazabilidad. Existen multitud de sistemas de sensores como los RFID[3], los sistemas GPS o la tecnología láser.

El crecimiento que han experimentado los puertos en los últimos años también ha supuesto que la sostenibilidad tanto económica como medioambiental juegue un papel relevante en la gestión de los puertos marítimos. La Organización Marítima Internacional, IMO, establece un marco regulatorio a nivel internacional que cubre aspectos relacionados con la eficiencia, el impacto medioambiental o la seguridad de los puertos y es la base sobre la que se desarrollan la gran mayoría de políticas marítimas a nivel nacional. Las regulaciones medioambientales recogidas por la IMO incluye medidas como reducir progresivamente las emisiones de gases de efecto invernadero en un 30 % en el 2025 o reducir notablemente las emisiones de dióxido de carbono de los buques para el año 2030 [20].

El concepto de puerto ecológico, o *green port* se relaciona con la idea de los *smart ports*. Esta clase de puertos se basa en pautas para prevenir contaminación ambiental que producen y reducir el impacto ecológico al ecosistema marino. Esta tendencia de desarrollo sostenible se encarga de integrar métodos respetuosos con el medio ambiente a la hora de realizar su gestión y actividades y reducir el impacto negativo del uso de los recursos [1].

Los retos que deben afrontar los puertos que buscan seguir unos estándares de puertos ecológicos están relacionados con la reducción de sedimentos y vertidos como el petróleo, que deben ser controlados y eliminados de forma adecuada. El control de la calidad del agua y el aire es otro de los puntos fundamentales de los puertos verdes. El uso de combustible de baja calidad, las emisiones de hidrocarburos, plomo o sulfuro y el uso de infraestructuras no adaptadas a las nuevas normativas son los principales causantes de estos problemas.

Las soluciones que se están aplicando en los principales puertos se basan en el desarrollo de programas de prevención de polución basados en la instalación de sensores que analizan parámetros como la cantidad de elementos químicos en las aguas portuarias, la densidad y tipo de fitoplancton o el sólido suspendido [23]. La utilización de infraestructura adecuada a las normativas, cómo el uso de grúas eléctricas para carga y descarga, es otra de las iniciativas más llamativas, que logra un ahorro de hasta un 86,6 % de energía y reduce las emisiones de CO_2 un 67,8 % respecto a las grúas tradicionales [30].

Estas tendencias actuales se relacionan directamente con los Objetivos de Desarrollo Sostenible. El concepto de los *smart ports* y su aplicación en las terminales puede impulsar el progreso y crear empleos de calidad. Así como los puertos ecológicos ayudan a proteger el ecosistema submarino, reducen la contaminación de las aguas e impulsan el uso de energias alternativas y no contaminantes.

[3]Sistemas basados en radiofrecuencia o *Radio Frequency Identification.*

CAPÍTULO 3
Estructura de la terminal de contenedores marítima

Las terminales portuarias constituyen el núcleo del comercio marítimo. Estas instalaciones experimentan diariamente el flujo constante de portacontenedores, buques de carga, petroleros... que aprovechan los puertos para cargar y descargar sus mercancías. A continuación se define la terminal portuaria de contenedores, así como sus infraestructuras y las principales actividades que se llevan a cabo dentro de estas instalaciones.

3.1 Descripción

El puerto puede definirse como el área geográfica donde se encuentran las terminales, en las que se almacenan, cargan y descargan contenedores provenientes de barcos y de otra clase de vehículos[1] y que conecta la parte del transporte marítimo con la del terrestre.

Otra aproximación al concepto de puerto es la definición que establece el Real Decreto Legislativo 2/2011 del 5 de septiembre, donde se recoge que un puerto marítimo es el «conjunto de espacios terrestres, aguas marítimas e instalaciones que, situado en la ribera de la mar o de las rías, reúna condiciones físicas, naturales o artificiales y de organización que permitan la realización de operaciones de tráfico portuario, y sea autorizado para el desarrollo de estas actividades por la Administración competente».

La tendencia al alza del empleo de contenedores dentro del comercio marítimo supone que las terminales de puertos tradicionales están especializándose y adaptándose a esta contenerización progresiva. La competitividad que ofrecen está basada en factores como los bajos tiempos de carga y descarga o la reducción del tiempo que pasan los barcos en la terminal. Estos factores suponen que el tránsito de contenedores sea considerablemente superior al presentado en las terminales más tradicionales.

Dentro de las terminales de contenedores, se pueden diferenciar generalmente cuatro zonas. Su correcta delimitación y localización supone una ventaja en la gestión de las terminales. Las cuatro zonas son las siguientes:

– **Área de operaciones marítimas:** Es el lugar situado en contacto con el agua, donde amarran temporalmente los barcos durante las operaciones encargadas de cargar y descargar los contenedores. La administración adecuada de esta zona reducirá el tiempo muerto que pasan los buques en el muelle. También se conoce por su denominación inglesa, *Ship Operation Area*.

[1] Camiones y trenes suelen ser los vehículos encargados de transportar los productos en la parte de tierra, como se verá a continuación.

- **Área de operaciones del patio de contenedores:** Lugar en el que se acopian los contenedores, tanto los de importación, los de exportación o los vacíos. Es la zona intermedia que pone en contacto la parte terrestre con la parte marítima de la terminal. Su correcta gestión es fundamental para lograr, mediante la ayuda de grúas y vehículos especializados, que los contenedores se distribuyan de forma eficaz en los patios.

- **Área de entrega y recepción:** Punto de conexión con los medios terrestres que se encargan de transportar los contenedores, camiones y trenes principalmente, que cargan y descargan en esta zona. También es llamada *Receipt/ delivery Operation Area* o área terrestre.

- **Área de servicios:** Es más conocida como *CFS Area*, o *Container Freight Station Area* y recoge la zona de talleres, almacenes u oficinas donde se controla el tráfico portuario, las operaciones comerciales o servicios aduaneros.

Esta estructura, sin embargo, puede diferir entre los distintos puertos. Por ejemplo, algunos puertos están especializados en el intercambio de contenedores entre barco y barco, por lo que el papel de la zona de tierra no suscita tanta importancia como en aquellos puertos dedicados principalmente a distribuir los productos hacia dentro del territorio. En otros, no es posible identificar una única área de servicios, ya que se encuentran distribuidos entre toda la terminal.

La construcción de las terminales de contenedores es un proyecto que supone una enorme inversión y que presenta restricciones tanto económicas como espaciales o medioambientales. Cabe destacar la importancia del factor espacial con la eficacia de la terminal, ya que el tamaño que presente será proporcional al número de buques y operaciones que puedan atenderse, por lo que se tendrá que planificar desde el comienzo de la construcción las posibles expansiones futuras que realizaría la terminal. El tamaño de las terminales de contenedores también está relacionado en cómo se organizan los contenedores dentro de ella.

Los contenedores normalmente se apilan unos sobre otros organizándose en distintos bloques o *stacks* que se distribuyen a lo largo de cada patio de las terminales de contenedores. Estos bloques son uno de los principales patrones que se siguen dentro de las operaciones con contenedores, y a su vez se diferencian en filas, bahías y alturas o niveles[2]. Las áreas más próximas a mar y a tierra se las conoce como los puntos de entrada/salida o puntos *Input/Output*, abreviados como E/S, y suele ser el punto donde los contenedores suelen ser cargados y descargados en los transportes terrestres y buques. En la figura 3.1 puede visualizarse esta organización de los bloques.

El cómo se trabaja en la terminal está condicionado por la organización de los contenedores a lo largo del patio. Mientras que algunas distribuciones requieren de infraestructuras complejas que controlan de forma automática el movimiento de los contenedores, otras tienden a utilizar más mano de obra. Estas diferencias en las configuraciones de las terminales permiten clasificarlas [19], en dos configuraciones principales: terminales asiáticas o terminales europeas. Factores como el nivel de automatización de la terminal, la posición y orientación de los bloques o la situación de los puntos de entrada/salida permiten clasificar qué tipo de configuración es la que sigue la terminal [5].

Los puertos asiáticos generalmente disponen de una extensión mayor que los europeos y con un sistema más enfocado a la mano de obra que a la automatización, siendo esta relativamente baja, dónde se trabaja transportando los contenedores mediante

[2]También se emplea normalmente sus traducciones inglesas *rows, bays* y *tiers*

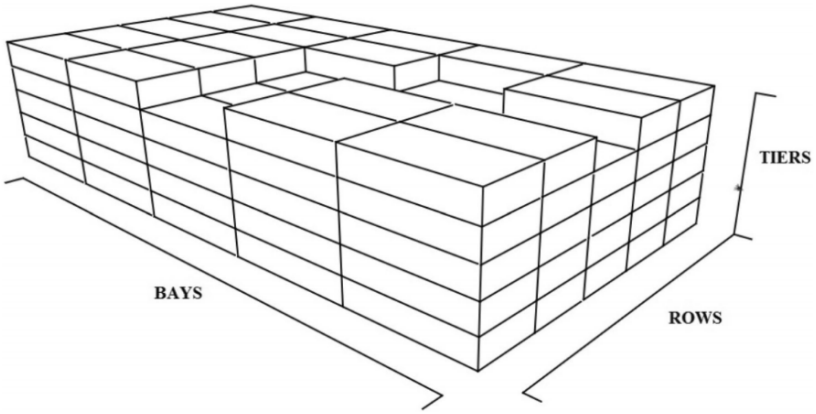


Figura 3.1: Representación de un bloque y de sus subdivisiones.
Figura extraída del artículo [16].

vehículos internos y externos. Las bahías se sitúan paralelamente al muelle, y las grúas recorren los bloques para colocar los contenedores en los vehículos.

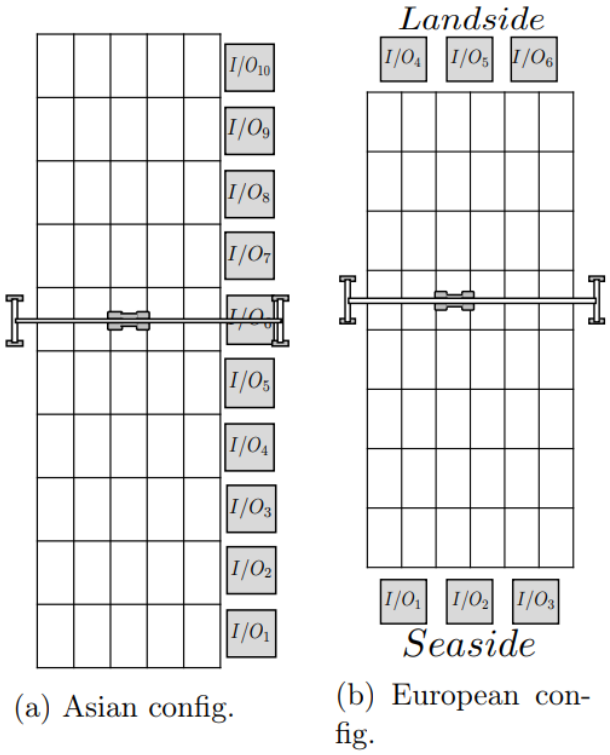
Por otra parte, la configuración europea se centra en la automatización de la terminal. Generalmente, los bloques se sitúan en posición perpendicular respecto al muelle, y las grúas de pórtico se encargan de situar los contenedores en los punto de entrada/salida de mar o tierra de cada bloque. Mediante el uso de vehículos guiados automatizados, se transporta el contenedor desde el punto de entrada/salida hasta su posición asignada en el bloque. Esta configuración optimiza considerablemente el espacio del patio de contenedores y es muy común en puertos europeos con ciertas restricciones de espacio pero con elevado flujo de contenedores TEU. Puertos como el de Róterdam o Amberes son líderes en esta clase de terminales.

En la figura 3.2 pueden observarse los esquemas de estas dos configuraciones. En la asiática se observa como los contenedores se sitúan paralelamente con los puntos de entrada y salida. En cambio, la configuración europea cuenta con puntos de entrada y salida en los dos extremos de los bloques, y se sitúan perpendicularmente respecto a estos.

Las terminales asiáticas no requieren realizar inversiones de infraestructuras, pero requieren mucho coste operacional. Por otra parte, la configuración europea necesita un gran desembolso inicial para automatizar los procesos, pero consigue coste operacionales inferiores. Son más productivas gracias a unos movimientos planificados, capaces de aligerar el transporte de contenedores, y presentan una mayor capacidad de almacenamiento por superficie al ahorrarse el espacio que necesitan los camiones para moverse entre los bloques en las asiáticas. La tendencia global apuesta cada vez más por enfocar las configuraciones de las nuevas terminales siguiendo un estilo europeo, debido a una tecnología para la automatización cada vez más asequible y las oportunidades que ofrece una terminal automatizada a la hora de recoger datos sobre las operaciones que realiza, que con tratamiento adecuado permiten mejorar considerablemente la productividad de las terminales.

3.2 Equipamientos e infraestructuras

Los puertos atienden niveles de tránsito de contenedores cada vez más elevados. Con esto nace la necesidad disponer de unas instalaciones y unas herramientas que les permitan realizar adecuadamente las operaciones sobre los contenedores, adecuándose a las



(a) Asian config. (b) European config.

Figura 3.2: Esquema de un bloque de contenedores en una terminal asiática y en una terminal europea.
Figura extraída de *Optimization Models and Methods for Storage Yard Operations in Maritime Container Terminals*, de Virgile Galle [14].

características del puerto y ofreciendo a estos unos niveles de competitividad acordes a su actividad. Gracias a la estandarización que supone el uso de contenedores, gran parte de las actividades del puerto se han visto simplificadas debido al uso de instrumentos generalizados que facilitan la comunicación y adaptabilidad entre puertos.

Desde el punto de vista logístico, estas infraestructuras aportan una gestión más eficiente del flujo de contenedores, centrándose fundamentalmente en reducir el tiempo de las operaciones y los movimientos que se realizan. Los equipamientos de los puertos pueden clasificarse, basándose en [26] en dos grupos:

3.2.1. Tipos de grúas

Dentro de las terminales se emplean distintos tipos de grúas. Destaca la conocida como grúa de muelle, o *quay crane* (figura 3.3), cuyo trabajo consiste en la carga y descarga directa de los contenedores sobre los buques. Estas grúas pueden ser de carro sencillo o carro doble. Las más comunes son las de carro sencillo, manejadas por un operario, mientras que las de doble carro aún no están muy extendidas al ser una tecnología relativamente reciente. Se sitúan en el muelle y tienen un rendimiento que permite operar entre 22 y 30 contenedores por hora operando a unos niveles de trabajo frecuentes.

Figura 3.3: Serie de grúas de muelle.
Figura extraída de la página web de AACNI.

Figura 3.4: Grúas de patio RTG.
Figura extraída de Nauticexpo.

El otro tipo corresponde a las grúas de almacenamiento o de patio, conocidas como *gantry crane*, cuyo papel consiste en mover los contenedores dentro de los bloques. Se clasifican en: grúas con neumáticos de goma (RTG)(figura 3.4), son las más flexibles ya que pueden desplazarse a distintos lugares; las grúas montadas sobre raíles (RMG), capaces de desplazarse a través de los carriles sobre los que están montadas, y suelen emplearse dos RMGs por bloque para evitar posibles fallos técnicos y además proporcionar una mayor productividad; y los puentes grúa (OBC), instalados sobre plataformas fijas de

hormigón. Las RTG y las RMG son capaces de cubrir en torno a 8-12 filas y apilar de 4 a 8 alturas en los bloques donde se instalan.

3.2.2. Medios de transporte horizontal

– **Vehículos internos:** Son los transportes que operan dentro de las terminales. Se encargan de trasladar las cargas dentro de las áreas de operaciones de contenedores. Pueden clasificarse según sean o no capaces de levantar los contenedores por cuenta propia. Los llamados "pasivos" no pueden levantar el contenedor por su cuenta, son camiones con tráiler, multitráiler y los AGV, o vehículos de guiado automático. Los AGV son capaces de cargar dos TEU simultáneamente y proporcionan niveles de automatización muy elevados al combinarse con grúas de pórtico automatizadas, pero requieren inversión elevada.

  Por otra parte están los vehículos capaces de levantar los contenedores por sí mismos. Destaca el uso de los conocidos como *straddle carriers*, o SC, vehículos manejados normalmente por un operario que permiten transportar simultáneamente hasta cuatro contenedores. La flexibilidad que ofrecen los SC permite tanto transportar los contenedores como apilarlos. También se utiliza una versión automatizada conocida como *automated lifting vehicles*, los ALV.

– **Vehículos externos:** Engloba todos aquellos transporten que no trabajan directamente dentro de las terminales, sino que operan fundamentalmente en los puntos de entrada/salida terrestres. Son los camiones, capaces de trabajar generalmente con una capacidad de 2 o 3 TEU, y ferrocarriles de mercancías, que manejan un flujo de contenedores más elevado que los camiones pero requieren una infraestructura adaptada.

En la figura 3.5 puede observarse las infraestructuras fundamentales que han sido expuestas en esta sección, así como las distintas zonas de la terminal y dónde se opera con los distintos tipos de contenedores.

3.3 Modelos de investigación de las operaciones

Durante los últimos años, el desarrollo de modelos de optimización sobre la organización y las operaciones que realizan las terminales de contenedores se ha convertido en un campo fundamental que proporciona grandes ventajas competitivas sobre la logística de estos puertos. Debido a que el grado de complejidad que presentan los flujos de contenedores es cada vez más elevado, es necesario apoyarse en estos modelos a la hora de adoptar decisiones operacionales y estratégicas.

Hechos como el retraso de algún buque o un funcionamiento no adecuado de alguna grúa afectan notablemente al funcionamiento de las terminales. En este horizonte de planificación temporal relativamente pequeño, es primordial trabajar con aproximaciones y modelos de decisión en tiempo real que permitan predecir el funcionamiento del puerto y actuar acorde a todas las variables que lo rodean.

A continuación, se exponen los principales retos que surgen en esta clase de terminales, clasificados según se localicen en la parte del mar, en el área de operaciones de contenedores o en la área terrestre. Se recomienda al lector el artículo de D. Steenken et al. [26] para profundizar en estos modelos de optimización.

– **Área marítima:** La zona de contacto de la terminal de contenedores con el agua es un punto fundamental en la productividad del puerto. La gestión tanto de los

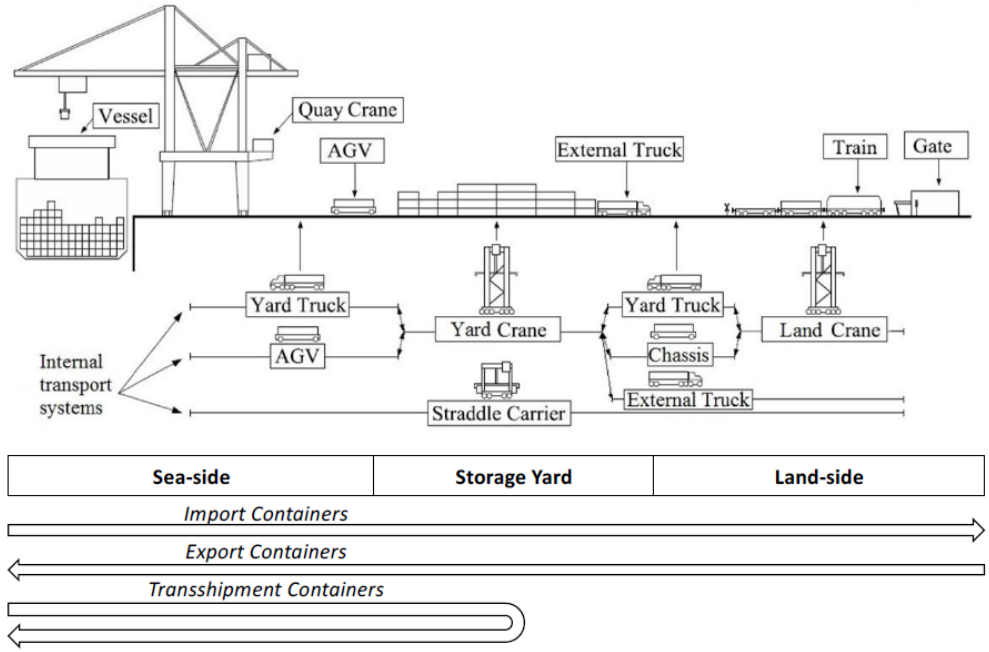


Figura 3.5: Esquema de una terminal de contenedores y sus principales equipamientos. Figura extraída de *Optimization Models and Methods for Storage Yard Operations in Maritime Container Terminals*, de Virgile Galle.

transportes marítimos como la de las grúas de muelle será clave para determinar el nivel de productividad que ofrece la terminal. Programar el cómo y el dónde de los barcos que van llegando al puerto aportará medidas con las que se podrá reducir el cuello de botella habitual en esta zona.

+ **Asignación del atraque.** Este problema también se conoce como *berth allocation*. Busca minimizar el tiempo de espera de los barcos a la hora de atracar en el puerto. A la hora de asignar estas localidades, hay que tener presente restricciones como el calado del barco, el tipo de grúas de muelle que operan en ese lugar o comprobar que el tamaño del barco es adecuado. Debe tenerse en cuenta que la terminal disponga de la información de cuándo va a llegar un barco con anterioridad suficiente, pero dada la incertidumbre que generalmente presentan estos horarios previos, es común que se recalcule las posiciones que se asignarán a los barcos conforme vayan pasando por la terminal los buques previos.
+ **Asignación de grúas de muelle.** Este problema, conocido como *quay crane assignment* ha sido analizado en [15]. Se centra en minimizar el número de grúas de muelle asignadas a cada buque a la vez que minimiza el número de movimientos que tiene que hacer la grúa para cargar y descargar contenedores. De esta forma se asigna la configuración de grúas que más productividad ofrezca. Una embarcación de tamaño habitual suele necesitar entre 3 a 5 grúas simultáneamente. Este problema está ligado al de asignación de atraque, ya que antes de la llegada del barco se tiene que conocer qué grúas son las más adecuadas para operar con él.
+ **Planificación de estiba**[3] o *stowage planning*. Trata la programación de una distribución de los contenedores que tenga en cuenta criterios de peso, tamaño y tipo de contenedores que permita reducir el tiempo que se tarda en realizar

[3]La estiba corresponde a la actividad de colocar y distribuir las cargas de una embarcación.

esta actividad y maximice el uso de las grúas de muelle. La estiba adecuada también debe garantizar la estabilidad y la flotabilidad de la embarcación tanto antes como después del proceso de carga/descarga de contenedores. Autores como Ding D. et al. [11] presentan algunos modelos de planificación de la estiba en los puertos.

- **Área de almacenamiento de contenedores:** Normalmente los contenedores se almacenan en los patios apilándose en los diversos bloques con los que cuenta la terminal. Este almacenamiento supone el reservar espacio exclusivamente para depositar los contenedores. Teniendo en cuenta que el desarrollo que han sufrido las terminales en los últimos años se traduce generalmente en un aumento considerable del número de los contenedores que pasan por sus instalaciones, puede establecerse la relevancia que tiene el controlar el espacio del que dispone el puerto. Estas restricciones de espacio se estudian mediante la asignación y la localización de los contenedores:

  + **Asignación de contenedores.** Tras la llegada de los contenedores a los puntos de entrada/salida marítimos y terrestres, el siguiente paso consiste en identificar cuál será el bloque en el que se colocarán. A la hora de seleccionar el bloque más adecuado, se tendrán en cuenta factores como las fechas de salida, el destino de los contenedores, si se trata de un contenedor de *input* o de *output* o el tamaño y peso de dichos contenedores. Mediante la evaluación de estos factores, se asigna el bloque más eficiente.

    Los problemas de asignación se han enfocado desde puntos de partida muy distinto y acorde a las necesidades del puerto. Kim K. et al. [17] centra en un modelo de asignación centrado en buscar los costes óptimos de almacenar los contenedores en los bloques. En Chen L. et al. [7], la asignación se subdivide en dos problemas: primero se establece el patio y la cantidad de localizaciones disponibles y en segundo lugar se determina el bloque y la posición exacta de cada contenedor.
  + **Localización de contenedores.** Una vez es asignado el bloque en el que se situará un contenedor, lo siguiente será establecer cuál será la ubicación exacta dentro de dicho bloque que facilite los procedimientos de carga y descarga de los contenedores. Tras conocer la localización de cada contenedor, se deberá mover desde un punto de entrada/salida sobre el que se encuentre inicialmente hasta el lugar asignado, buscando el minimizar el tiempo que se tarda en mover el contenedor a la vez que se asegure que los contenedores se encuentran accesibles sin realizar movimientos adicionales.

    El concepto de *pre-marshalling* hace referencia al reposicionamiento de los contenedores que tiene como finalidad reducir posibles futuras relocalizaciones que implicarían un aumento del número de movimientos de las grúas de almacenamiento. Los problemas de localización de contenedores dentro de los bloques están estrechamente relacionados con este concepto, ya que la gestión de los reposicionamientos de los contenedores tiene un impacto notable en el tiempo que se tarda en colocar el contenedor en el lugar asignado.

- **Área terrestre:** Aunque el área del mar juega un papel más importante ya que cualquier retraso de los barcos supone unos costes mucho más elevados que el ligado a los retrasos de ferrocarriles o camiones, también conviene plantear aproximaciones adecuadas para evitar la formación de cuellos de botella en la parte de tierra.

  En esta zona tenemos problemas relacionados con los transportes internos y con los vehículos de exterior que pasan por el puerto a recoger los contenedores en los puntos de entrada/salida. Respecto al transporte interno, estos vehículos se utilizan de

manera constante a lo largo de la terminal, principalmente en las terminales de tipo asiático, y muchas veces siguen caminos redundantes y poco productivos que afectan negativamente al flujo de la terminal. En estos problemas se determinarán las rutas más cortas y que más posibilidades de sincronización entre todos los vehículos operativos ofrece. También se controla la flota de vehículos, comprobando que no se caiga en sobrecostes u ociosidad.

En relación a los ferrocarriles y camiones que conectan con el puerto, se busca fundamentalmente el evitar la congestión en los puntos de entrada/salida. Se suele trabajar gestionando adecuadamente las ventanas de tiempo de carga y descarga de estos medios de transporte.

CAPÍTULO 4
Diseño de métodos: localización de contenedores

En este capítulo se trata el problema de localización de contenedores dentro de un bloque. En la sección 4.1 se define formalmente el problema, sus principales características y la notación utilizada. Una primera aproximación matemática relacionada con el problema de asignación aparece reflejada en la sección 4.2. Después, se muestran los algoritmos propuestos para resolver el problema y se estudia, mediante la aplicación de unos ejemplos sencillos, las soluciones que ofrece cada una de ellas en la sección 4.3. Por último, en la sección 4.4, se analiza la eficiencia de cada algoritmo a partir de una serie de instancias generadas aleatoriamente.

4.1 Descripción del problema

4.1.1. Características del problema de localización de contenedores

El problema de localización de contenedores consiste en determinar la ubicación más adecuada para cada contenedor en un bloque concreto de la terminal. Los contenedores serán ubicados en la localización disponible determinada más factible, y el contenedor se alojará temporalmente en ella. Queda a la espera de cargarse posteriormente en otro transporte marítimo o terrestre. La localización de contenedores constituye uno de los problemas más comunes que aparece a diario en cualquier terminal portuaria del mundo.

Mediante una gestión adecuada, se ofrece una planificación capaz de reducir el tiempo necesario de ubicación, de forma que no se produzcan movimientos adicionales innecesarios.

Generalmente, la terminal portuaria dispone, a priori, de una información aproximada sobre los contenedores previstos. Con esta información, se pueden desarrollar posibles planes de estiba y asignaciones de bloques previos que determinarán la posterior localización dentro de los bloques. Sin embargo, según se explica en Steenken et al. [26], hay cierta carencia a la hora de que las terminales puedan recopilar información precisa. Se estima que entre un $30-40\,\%$ de los contenedores de exportación y, únicamente, entre un $10-15\,\%$ de los contenedores de importación aportan datos completos antes de llegar. Además, contratiempos como los retrasos de los buques o la congestión de la infraestructura, pueden provocar una planificación de localizaciones considerablemente distinta. Las terminales elaboran diferentes planificaciones para reducir la incertidumbre de estos contratiempos. La terminal deberá tener en cuenta toda la información que disponga, a fin de aumentar la productividad del patio.

Esta falta de información lleva a que las terminales realicen el llamado *pre-marshalling*, previamente introducido en la sección 3.3. Aunque se hayan localizado adecuadamente, la llegada de nuevos contenedores a un bloque provoca que algunos de ellos no estén organizados adecuadamente. El *pre-marshalling* se producirá antes de la llegada de los nuevos contenedores, reorganizando aquellos ya colocados, a fin de reducir los posibles movimientos adicionales futuros que provocaría el operar sobre un bloque que no estuviese organizado.

La eficacia de la localización está estrechamente ligada a la gestión de otros problemas, que aparecen antes y después de este paso. La asignación de los contenedores en los distintos bloques debe asegurar que los contenedores se transportan a los bloques más adecuados. Así, evitaremos posibles congestiones y movimientos adicionales, como colocar el contenedor en un punto de entrada/salida (E/S) no contemplado inicialmente. Se considera que la planificación de la estiba y la asignación de bloques y vehículos, son las más adecuada, de forma que no se tendrá en cuenta la gestión de estas actividades.

Dentro de cada bloque de la terminal pueden observarse tres dimensiones. En este problema, un bloque B contará con X filas, Y bahías y Z alturas. Cada una de estas coordenadas (X, Y, Z) corresponde a una localización que puede ser ocupada por un contenedor, normalmente referido a aquellos de tamaño correspondiente a un TEU. En una terminal de tamaño estándar, los bloques tienen unas dimensiones entorno a $6 - 12$ filas(X), entre $14 - 40$ bahías (Y) y $2 - 5$ alturas (Z).

En este problema se establece que las tres dimensiones (X, Y, Z) se encuentran dentro del rango estándar con el que trabaja una terminal de contenedores. En la figura 4.1 se reflejan estas coordenadas sobre un bloque de la terminal en el que opera una grúa de patio.

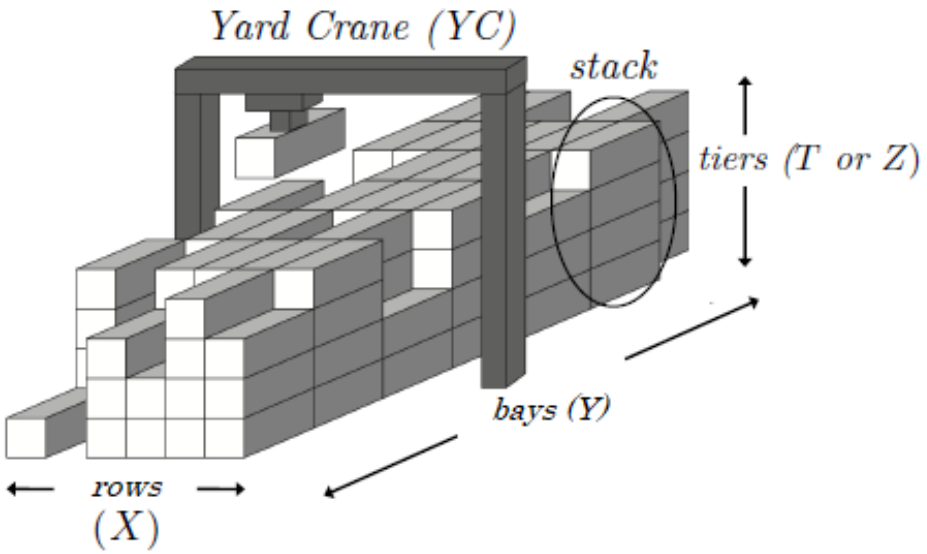


Figura 4.1: Geometría y notación de un bloque.
Figura extraída de [14], de Virgile Galle.

Apilar unos contenedores sobre otros supone que no haya acceso directo a los situados debajo. A la hora de apilar los contenedores, debe buscarse un equilibrio entre la altura y la accesibilidad que existe a los contenedores inferiores. Además de establecer unas alturas que aseguren la estabilidad, las grúas y vehículos adoptan un papel fundamental en la delimitación de la altura. Vehículos como las RTG, capaces de operar con alturas que no suelen superar los 8 contenedores, o las SC que soportan hasta cuatro ni-

veles de contenedores, son determinantes para delimitar el nivel máximo de los bloques de las terminales.

Las distintas características de los contenedores ayudan a determinar cuales son las posibles ubicaciones aptas para cada uno. El peso es un factor notable que influye en cómo apilar los contenedores. Se recomiendo apilar aquellos más pesados sobre los más ligeros para evitar futuros *rehandle*. Esto se debe, según expone Steenken et al. [26], a que los contenedores más pesados suelen ser colocados antes en los barcos y vehículos que aquellos de menor peso, ya que aporta una mayor estabilidad a los distintos medios de transporte. En este problema, se asume que los contenedores cumplen estos criterios de peso para cada una de las localizaciones que tienen disponibles.

Los puntos de E/S corresponden a los lugares del patio en los que se ubican los contenedores descargados de los barcos y de los vehículos terrestres, tal como se muestra en la sección 3.1. Hay que tener presente que la función de estos puntos no es la de almacenar contenedores como si se tratara de un bloque más, sino que son puntos intermedios sobre los que se sitúan temporalmente los contenedores a la espera de que sean transportados a la localización asignada. En este problema se considera que no existen problemas de congestión en estos puntos, por lo que siempre que un contenedor llega a un punto E/S, hay huecos suficientes para poder colocarlo.

Ya que el problema puede ser enfocado según se opere en la parte marítima y/o en la parte terrestre, resultará conveniente diferenciar entre los puntos de entrada/salida según se localicen en una u otra parte. De esta forma, los puntos de entrada/salida correspondientes a la parte marítima serán denominados IO_m. Los puntos de la zona terrestre serán IO_t. Para simplificar el problema, se asumirá un único punto IO en cada extremo del bloque, cuya capacidad será infinita a fin de poder colocar los contenedores que van llegando a lo largo del tiempo sin provocar retrasos.

Partiendo de la literatura de los problemas de localización de contenedores, el equipamiento más utilizado son las grúas RTG o las grúas RMG. Uno de los principales motivos es la flexibilidad que son capaces de ofrecer frente a las grúas fijas. Con estas grúas se puede acceder al interior de los bloques de forma más sencilla que con vehículos que operan alrededor de los bloques, por ejemplo los SC.

Conocer el tipo de configuración de grúas que sigue la terminal es un factor determinante a la hora de diseñar e implementar las metodologías en el puerto. La configuración de grúa única o simple trabaja con una sola grúa capaz de abarcar todas las localizaciones del bloque. Esta configuración permite operar unos $15 - 20$ contenedores por hora en un patio de tamaño estándar. Por otro lado, las grúas duales se sitúan en los dos extremos del bloque y ofrecen una mayor productividad. Estas pueden operar simultáneamente con más de un contenedor. Sin embargo, las grúas duales tienen un sistema de control más complejo que las simples: aunque sean capaces de realizar movimientos simultáneos, debe asegurarse que una grúa no se cruce con otra para evitar bloqueos.

En este problema se asume que la terminal operará mediante una configuración de grúa única, y será independientemente de tipo RTG o RMG. Esta grúa se sitúa en una posición inicial g_i correspondiente al punto IO_m. Al colocarse sobre dicho punto de E/S, puede acceder a los contenedores que van depositándose bajo él. La grúa del problema puede moverse a lo largo de todo el bloque y sobre los puntos IO_m y IO_t. Se establece que la terminal dispone de las grúas de muelle y vehículos de transporte necesarios, que permiten depositar los contenedores en el IO correspondiente, y se asegura que el contenedor que ya ha llegado a su correspondiente IO estará disponible en dicho punto siempre que la grúa de patio necesite acceder a él.

La posición de la grúa de muelle está definida siguiendo el conjunto de coordenadas (X, Y, Z). Para unificar la representación del problema, el movimiento de la grúa de patio

también se representa mediante coordenadas. Esto indica que la grúa puede desplazarse en dirección x y el carro podrá moverse sobre y y z. La grúa y el carro pueden moverse al mismo tiempo sobre x e y, y una vez se encuentre situado sobre la localización del contenedor sobre el que opera, se moverá sobre z.

Se establece que la velocidad y el movimiento de la grúa en todas las direcciones es constante y no sufre interferencias provocadas por operaciones ajenas al proceso de localización de contenedores dentro del bloque, ni por el peso del contenedor que se transporta. También se asume que el tiempo necesario para asegurar y soltar los contenedores se encuentra implícitamente incluido en los movimientos sobre z.

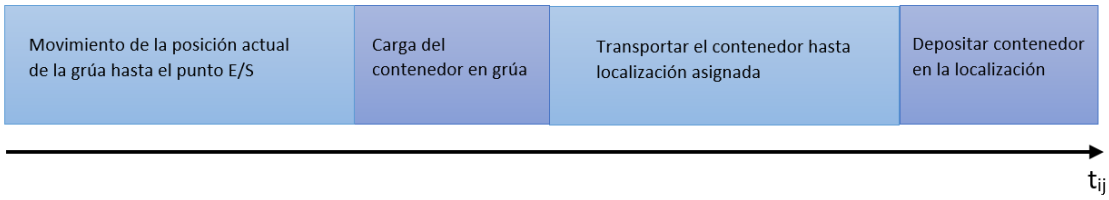


Figura 4.2: Esquema temporal de las etapas de localización de un contenedor.
Figura de elaboración propia.

El proceso que sigue la grúa para almacenar un contenedor consta de cuatro etapas, que ocurrirán cada vez que se reciba la petición de localizar un contenedor dentro de un bloque. Las etapas, esquematizadas en la figura 4.2, son las siguientes:

a) **Movimiento de la grúa desde su posición actual hasta el IO donde está el contenedor.** La grúa de patio se desplaza hasta estar situada sobre el punto IO del contenedor sobre el que operará. En el caso en que la grúa ya se encuentre situada en el punto de entrada/salida correspondiente, esta etapa tendrá un valor 0. Esta fase se formula en 4.1:

$$Max\{|x_g - x_{IO}|, |y_g - y_{IO}|\} \tag{4.1}$$

b) **Carga del contenedor en la grúa vacía.** Se parte desde la posición inicial g_i de la grúa, la cuál bajará hasta el punto IO que se encuentra bajo ella, enganchará el contenedor y volverá a subir hasta la altura inicial z de la grúa. De esta forma se evita provocar posibles colisiones con las pilas de contenedores al desplazarse. La fórmula 4.2 calcula la duración de este movimiento:

$$|z_g - z_{IO}| + |z_g - z_{IO}| \tag{4.2}$$

c) **Transportar el contenedor hasta la localización asignada.** La grúa y el carro son capaces de moverse de forma simultánea en las direcciones x e y. Por lo tanto, la distancia desde el punto de inicio hasta situarse sobre el punto en el que se ubicará el contenedor, corresponde a la distancia máxima recorrida en x o en y. Mediante la fórmula 4.3 se calcula esta fase.

$$Max\{|x_g - x_{L_i}|, |y_g - y_{L_i}|\} \tag{4.3}$$

d) **Depositar el contenedor en la localización.** Después de que el contenedor se encuentre situado sobre la localización asignada, la grúa desciende hasta depositarlo

en el nivel establecido y vuelve a subir hasta la altura inicial z de la grúa. Esta actividad corresponde a la siguiente fórmula:

$$|z_g - z_{L_i}| + |z_g - z_{L_i}| \tag{4.4}$$

Por lo tanto, el tiempo total (t_{ij}) que tardará la grúa en almacenar el contenedor i en la localización j será el resultado de sumar las ecuaciones 4.1, 4.2, 4.3 y 4.4:

$$\begin{aligned} t_{ij} = Max\{|x_g - x_{IO}|, |y_g - y_{IO}|\} + |z_g - z_{IO}| + |z_g - z_{IO}| + \\ Max\{|x_g - x_{L_i}|, |y_g - y_{L_i}|\} + |z_g - z_{L_i}| + |z_g - z_{L_i}| \end{aligned} \tag{4.5}$$

Según las maniobras que se realicen y, teniendo en cuenta el origen y el destino del bloque, los contenedores pueden ser de cuatro tipos distintos:

- **Sea-to-yard:** el contenedor llega por la parte marítima y se almacena en el patio de contenedores.
- **Land-to-yard:** el contenedor llega por la parte terrestre y se almacena en el patio de contenedores.
- **Yard-to-sea:** el contenedor sale del patio de contenedores hacia la parte de mar.
- **Yard-to-land:** el contenedor sale del patio de contenedores hacia la parte terrestre.

Se asume que los bloques del patio de la terminal portuaria no están inicialmente vacíos. A pesar de que partiendo de un bloque vacío se simplifica el problema de localización notablemente, no es realista asumir que los bloques de un patio operativo se encuentran sin ningún contenedor inicialmente. El bloque será definido por una configuración inicial en la que se indicarán cuáles son las coordenadas sobre las que ya hay situados contenedores previamente y, partiendo de dicha configuración, se establecen cuáles son las localizaciones que se tienen en cuenta en el problema.

Se supone que durante todo el tiempo que dura el proceso de localizar los contenedores en el bloque, únicamente entrarán contenedores conforme vayan llegando. En un bloque pueden entrar contenedores que vengan de mar o tierra, pero también pueden salir del bloque aquellos contenedores que vayan a ser cargados en barcos o vehículos terrestre o ser transportados hacia un segundo bloque (por ejemplo, aquellos contenedores vacíos o cargados). Estas operaciones de salida de contenedores son realizadas fuera del tiempo en el que se están procesando las localizaciones de contenedores. De esta forma, durante la localización solo se va a gestionar la entrada de contenedores al bloque, sin que influya en el problema aquellos que pudiesen salir.

La gestión de los contenedores que llegan al punto de E/S seguirá una política *First In First Out*, también conocida como FIFO. Esta política indica que la gestión de los contenedores se realizará atendiendo primero aquellas peticiones que lleguen antes a la terminal. De esta forma, la fecha de llegada al punto E/S representará el orden en el que serán atendidas las peticiones de localizar los contenedores. Los contenedores que tengan un menor tiempo de llegada serán atendidos antes que aquellos con un tiempo de llegada superior. En caso de que el tiempo de llegada a E/S sea el mismo para varias peticiones, se elegirán siguiendo un orden aleatorio entre ellas. Gracias a esta política FIFO, se representa el carácter temporal del problema en la gestión de los contenedores y los puntos de entrada/salida. Para que esta gestión FIFO pueda ejecutarse, se acepta que la fecha de llegada de todos los contenedores es conocida antes de comenzar a localizar las peticiones.

El objetivo es, teniendo en cuenta las características comentadas anteriormente y dada una lista de contenedores, almacenar cada contenedor en la localización asignada, de tal manera que este proceso dure lo menos posible.

4.1.2. Notación del problema

Antes de comenzar a diseñar los algoritmos con los que aproximar la solución del problema, se establece la notación fundamental con la que se identificarán los parámetros principales de forma clara. La tabla 4.1 muestra la notación necesaria para identificar algunos parámetros útiles para explicar, en posteriores secciones, algunos métodos de resolución del problema.

Notación	Significado
C	conjunto de contenedores a localizar: $1, 2, \ldots, C$.
L	conjunto de localizaciones disponibles: $1, 2, \ldots, L$.
(x, y, z)	coordenadas: x filas, y bahías, z alturas.
$\tau : \{m, t\}$	lado marítimo (m) o lado terrestre (t).
$\gamma : \{1, 2\}$	1 (sea-to-yard), 2 (land-to-yard).
IO_τ	punto de entrada/salida del lado τ sobre el que se trabaja.
T_γ	contenedor de tipo γ
r_i	momento a partir del cual el contenedor i está disponible en el IO_τ.
F_i	instante en el que finaliza el proceso de localización del contenedor i
i	contenedor $i \in C$
j	localización disponible $j \in L$
t_{ij}	tiempo que tarda la grúa en realizar la operación de localizar el contenedor $i \in C$ en la localización $j \in L$

Tabla 4.1: Notaciones principales del problema de localización.

Tabla de elaboración propia.

4.1.3. Ejemplo de estudio

A continuación, proponemos un ejemplo que servirá de apoyo a la hora de explicar los distintos algoritmos propuestos en este trabajo.

Las tablas 4.2, 4.3 y 4.4 recogen las características del problema. La tabla 4.2 muestra el número de contenedores a localizar, las dimensiones del bloque y las coordenadas de los puntos de E/S así como de la grúa.

La tabla 4.3 muestra la lista de contenedores del problema y sus características.

En la tabla 4.4 se muestran las localizaciones libres que tiene asignadas el ejemplo de estudio.

El objetivo de esta clase de problemas consiste en devolver el orden en el que cada uno de los contenedores es localizado, de forma que se minimice el tiempo necesario en realizar todas las operaciones. Esto supone que la representación de la solución del problema muestre un tiempo total F_i que recoja todos los t_{ij} hasta el momento y tenga en cuenta los tiempos de llegada a los puntos de entrada o salida (r_i).

Además de conocer este tiempo total, también resulta conveniente disponer de una lista que indique el orden en el que se han tratado los contenedores, que localización se les ha asignado y su coste t_{ij} correspondiente. De esta forma, no solo se conoce el coste

Número de contenedores a localizar C	5
Número de filas X	10
Número de bahías Y	42
Número de alturas Z	4
Posición inicial de la grúa	$(5, 0, 5)$
Punto IO_m	$(5, 0, 1)$
Punto IO_t	$(5, 43, 2)$

Tabla 4.2: Información del problema para el ejemplo de estudio.

Tabla de elaboración propia.

Contenedor	T_γ	r_i
A	1	29
B	2	6
C	1	1
D	2	8
E	1	18

Tabla 4.3: Lista de contenedores a localizar del ejemplo de estudio.

Tabla de elaboración propia.

Localización	**Fila** (x)	**Bahía** (y)	**Altura** (z)
$l1$	1	5	1
$l2$	4	6	1
$l3$	2	12	1
$l4$	6	13	2
$l5$	3	14	2
$l6$	5	23	1
$l7$	8	10	1
$l8$	6	8	2
$l9$	4	9	2
$l10$	5	10	1

Tabla 4.4: Lista de localizaciones disponibles iniciales para el ejemplo de estudio.

Tabla de elaboración propia.

temporal total que supone el localizar todos los contenedores, sino que también se se conoce el orden en el que se debe realizar este proceso. Por tanto, la solución de este problema puede representarse a través de una lista, dónde cada uno de sus elementos tiene tres componentes: el contenedor, la localización asignada y su tiempo F_i, tal como representa 4.6.

$$\{\overbrace{(A, L_A, F_A)}^{1}, \overbrace{(B, L_B, F_B)}^{2}, \ldots, \overbrace{C, L_C, F_C}^{C}\} \tag{4.6}$$

El último tiempo F_C que ofrezca la lista corresponderá al tiempo total que se tarda en realizar todas las operaciones del problema de localización.

4.2 Simplificación matemática al problema de asignación

En esta sección se muestra una simplificación inicial del problema de localización de contenedores dentro de un único bloque descrito en la sección anterior 4.1, y se expone una solución de esta formulación simplificada a partir del ejemplo de estudio planteado en el apartado 4.1.3.

4.2.1. Modelo simplificado

El problema de localización, aunque a priori pueda parecer relativamente sencillo, atiende a un gran número de características y variables para proporcionar una solución cercana a la realidad del problema.

Aunque puede tratar de plantearse de forma concreta algún método que contemple todas las características del problema, solo sería adecuado para una situación con pocos contenedores y localizaciones. Estos métodos exactos se encargan de buscar exhaustivamente una solución óptima global del problema, pero son muy dependientes del tamaño que presenta: su aplicación a problemas pequeños es aceptable; pero en los más complejos, es considerado un método poco viable de resolución, según se expone en De Armas et al. [9].

El problema de localización de contenedores dentro de un bloque puede aproximarse, de forma general, a un problema de asignación en el que cada contenedor se le asigna un hueco dentro del bloque. Concretamente, este tipo de problemas puede clasificarse en la categoría de los llamados problemas de asignación asimétrica. La asignación asimétrica, expuesta en el artículo [3], ocurre cuando los conjuntos de datos sobre los que se trabaja no son del mismo tamaño. En la localización, los C contenedores a localizar son menores que las L localizaciones disponibles ($C < L$). Este tipo de asignación indica que, tras haber asignado todos los contenedores, seguirán existiendo localizaciones libres.

Para que nuestro problema pueda resolverse óptimamente como un problema de asignación, debemos realizar las siguientes simplificaciones:

1) Todos los contenedores disponen de un tiempo $r_i = 0$, por lo que todos se encuentran disponibles nada más comenzar.

2) Todos los contenedores que llegan serán del tipo T_1, es decir, solo entrarán contenedores procedentes de la parte marítima. Este hecho supone que la grúa realizará movimientos cíclicos cuando transporte los contenedores, volviendo siempre al mismo lugar inicial.

Definimos la variable decisión X_{ij} tal que tomará el valor 1 cuando el contenedor $i \in C$ se asigne a la localización $j \in L$ y 0 en caso contrario.

Por tanto, la formulación matemática para la simplificación del problema sería la siguiente:

$$Min \sum_{i=1}^{C} \sum_{j=1}^{L} t_{ij} \cdot X_{ij} \tag{4.7}$$

Sujeto a las restricciones:

$$\sum_{j=1}^{L} X_{ij} = 1, \quad i = 1, 2, \ldots, C \tag{4.8}$$

$$\sum_{i=1}^{C} X_{ij} \leq 1, \quad j = 1, 2, \ldots, L \tag{4.9}$$

$$X_{ij} \in \{0, 1\}, \quad \forall i \in C, \forall j \in L \tag{4.10}$$

La función objetivo 4.7 minimiza el coste temporal de asignar y ubicar cada contenedor i en una localización j. Corresponde al sumatorio del producto del tiempo t_{ij}, calculado siguiendo la fórmula 4.5, por la variable decisión X_{ij}, que indicará si la localización está o no disponible.

Gracias a la restricción 4.8, se asegura que cada uno de los contenedores de la planificación debe estar siempre asignado a exactamente una única localización de las inicialmente disponibles cuando finalice el problema.

Con la restricción 4.9 sobre la localización se indica que en cada ubicación inicialmente disponible puede haber como mucho un contenedor asignado. El hecho de que la suma de sea igual o inferior a 1 busca que se pueda satisfacer que una localización no tenga ningún contenedor asignado, o bien que entre todos los contenedores de la planificación pendientes de asignar, solo haya sido asignado uno.

Por último, la restricción 4.10 asegura que la variable decisión X_{ij} pueda tomar los valores 0 o 1.

4.2.2. Resolución de la formulación simplificada

Una de las técnicas más comunes para la resolución de esta clase de problemas de programación lineal es el método *Simplex*, un procedimiento capaz de resolver problemas de maximización y minimización de PL en los que mejora la solución en cada iteración que realiza. Para la resolución, se empleará la herramienta Solver en Excel, mediante la cual se puede obtener el valor mínimo de la función objetivo 4.7 sujeta a las restricciones 4.9, 4.8 y 4.10.

Utilizaremos el ejemplo de estudio expuesto en el apartado 4.1.3 y se adaptará para que refleje las asunciones de la simplificación. Para ello, estableceremos, por ejemplo, que el r_i de todos los contenedores es igual a 0 y el tipo de todos ellos corresponde a sea-to-yard.

A partir de la fórmula del tiempo 4.5 se calculan los t_{ij}. En este caso se ha establecido que los contenedores únicamente pueden llegar de mar, por lo que este t_{ij} representará el tiempo que tarda la grúa en realizar el ciclo de localizar el contenedor y volver al punto inicial.

La función objetivo de este problema corresponderá a la siguiente ecuación:

$$Min\ 26 \cdot X_{Al1} + 28 \cdot X_{Al2} + 40 \cdot X_{Al3} + \cdots + 36 \cdot X_{El10}$$

En la tabla 4.3 se representa la matriz de las variables decisión, donde cada casilla representa el valor de la variable X_{ij} correspondiente. La suma de las filas indica la restricción en la que hay una única localización asignada a cada contenedor, mientras que la

Localizaciones j	l1	l2	l3	l4	l5	l6	l7	l8	l9	l10
t_{ij}	26	28	40	40	42	62	36	30	32	36

Tabla 4.5: Tiempo t_{ij} de localizar un contenedor en j en el problema de asignación.

Tabla de elaboración propia.

		LOCALIZACIONES DISPONIBLES (j)												
		l1	l2	l3	l4	l5	l6	l7	l8	l9	l10			
CONTENEDORES (i)	A	1	0	0	0	0	0	0	0	0	0	1	=	1
	B	0	1	0	0	0	0	0	0	0	0	1	=	1
	C	0	0	0	0	0	0	0	1	0	0	1	=	1
	D	0	0	0	0	0	0	0	0	1	0	1	=	1
	E	0	0	0	0	0	0	0	0	0	1	1	=	1
		1	1	0	0	0	0	0	1	1	1			
		<=	<=	<=	<=	<=	<=	<=	<=	<=	<=			
		1	1	1	1	1	1	1	1	1	1			
F.O.	152 u.t													

Figura 4.3: Matriz de las variables decisión X_{ij} de la solución del problema de asignación.
Figura de elaboración propia.

suma de las columnas representa la restricción de asignar, como mucho, un contenedor en cada localización.

Esta resolución calculada con Solver indica que la función objetivo corresponde a la ecuación 4.11 y tiene un coste de 152 u.t. Se alcanzará asignando el contenedor A a la localización $l1$, el B a $l2$, C a $l8$, D a $l9$ y el E a $l10$. Puede comprobarse que las restricciones se han cumplido, ya que cada contenedor está asignado a una única localización. Además, las localizaciones solamente tienen como mucho un contenedor asignado y se cumple que todas las variables aleatorias son binarias.

$$26 \cdot X_{Al1} + 28 \cdot X_{Bl2} + 30 \cdot X_{Cl8} + 32 \cdot X_{Dl9} + 36 \cdot X_{El10} = 152\, u.t. \tag{4.11}$$

Para finalizar, anotamos que el tiempo de cómputo de esta solución mediante Solver es de 0,062 segundos.

4.3 Algoritmos propuestos

Basándose en lo establecido en las secciones 4.1 y 4.2, se realiza una introducción explicando qué es una heurística y por qué es relevante a la hora de estudiar posibles soluciones. Después de esto se presentarán los algoritmos que se han planteado en este trabajo para resolver el problema. Por último, se analizará cuáles de ellos son capaces de ofrecer una solución más adecuada a este problema.

4.3.1. Introducción a la heurística

Algunos problemas de optimización tienen un proceso de resolución relativamente sencillo y rápido. Problemas de programación lineal en los que, mediante métodos como la búsqueda del punto interior o el método *simplex*, pueden resolverse de formas simples y con un coste temporal no muy elevado. Sin embargo, la mayor parte de los problemas

de optimización no son tan fáciles de resolver y el hecho de establecer la solución óptima puede requerir un tiempo elevado, incluso puede llegar a no ser posible calcularla.

Generalmente, los problemas de localización en tiempo real teniendo en cuenta restricciones físicas y temporales se clasifican como *NP-Hard*, según el artículo de Caserta et al. [6]. Esta clase de problemas presenta un nivel de complejidad elevado que provoca que no sea posible garantizar una solución óptima del problema en un tiempo polinomial. El clasificarse como un problema *NP-Hard* conlleva que sea necesario realizar un mayor esfuerzo para resolverlo y se necesite utilizar métodos de resolución alternativos. Estas aproximaciones, son capaces de ofrecer no la solución óptima, sino una solución aproximada y rápida.

Gracias a los conocidos como métodos heurísticos se buscan soluciones aceptables para los problemas, cuyo cálculo pueda realizarse en un período breve de tiempo. La heurística o métodos heurísticos son todos aquellos algoritmos, estrategias o procedimientos que, mediante la aproximación intuitiva a un problema, son capaces de obtener una buena solución en un tiempo corto, según se expone en el artículo de Martí[22]. Por tanto, los métodos heurísticos se encargarán de proporcionar una solución buena, en tiempos de cómputo aceptables, que no puede garantizarse que sea la solución óptima.

La simplificación propuesta en la sección 4.2 ofrece una solución capaz de garantizar unos resultados óptimos, aunque corresponde únicamente a este modelo simplificado que no refleja gran parte de las condiciones del problema. Son estos r_i y el hecho de que vengan contenedores tanto por la parte marítima como terrestre, lo que justificará el uso de heurísticas en la resolución de nuestro problema.

La utilización de métodos heurísticos permite resolver esta clase de problemas complejos, evitando incurrir en los costes de cómputo elevados que llevan asociados los métodos exactos. Además, permite explorar y representar las distintas características del problema de localización basándose en los datos y la información que se proporcionan.

Emplear heurísticas para la resolución de problemas se caracteriza por presentar unas estrategias generales a problemas comunes muy diversos. Gracias a la información específica disponible en cada problema, es posible adaptar estas aproximaciones de carácter general de unos problemas base, como la asignación de recursos o los de selección de rutas, y ofrecer una solución especifica y personalizada para cada problema particular. Además, también aporta cierta simplicidad en la representación si lo comparamos con formulaciones matemáticas exactas.

Gracias a la flexibilidad que ofrecen los métodos heurísticos, es posible plantear condiciones cuya representación matemática supone un grado de complejidad elevado. El incorporar las características de nuestro problema resulta mucho más sencillo introduciendo heurísticas en vez de modelos matemáticos.

Otra de las razones por las que los procedimientos heurísticos resultan interesantes es la capacidad de proporcionar una solución inicial sencilla de entender. Sobre esta, se pueden aplicarse iterativamente criterios con los que se especificará una solución capaz de representar mejor las características del problema.

Gran parte de los artículos académicos relacionados con esta problemática realizan aproximaciones aplicando estos métodos heurísticos y metaheurísticos[1], por lo que será relevante tener presentes los métodos heurísticos para las aproximaciones de este trabajo.

[1]Los métodos metaheurísticos, al igual que los heurísticos, no garantizan encontrar una solución óptica y se emplean para resolver aquellos problemas más complejos que no son capaces de cubrir los métodos heurísticos. Normalmente, son métodos de carácter general que no son dependientes del problema.

4.3.2. Algoritmo de localización aleatoria

El primer método heurístico que se plantea en este trabajo es un algoritmo de localización aleatoria de contenedores. Este primer algoritmo se centra en el concepto de aleatoriedad a la hora de asignar la localización dentro del bloque. Esta clase de algoritmos aleatorios, también llamados algoritmos *random*, suelen ser tomados como punto de referencia a la hora de comparar distintos algoritmos. El método aleatorio utilizado ofrece una solución aceptable del problema en el que las localizaciones se asignan con el único criterio de ser seleccionadas de manera aleatoria.

El diseño de métodos aleatorios suele presentar un planteamiento sencillo, por lo que estas aproximaciones son un buen punto de partida sobre el que plantear otros algoritmos más complejos. Además, las soluciones que ofrecen pueden llegar a encontrarse en unos rangos similares a las ofrecidas por aproximaciones más complejas. Teniendo en cuenta que, generalmente, esta clase de algoritmos no consume muchos recursos en su cálculo, resulta interesante comenzar a plantear los métodos heurísticos de este trabajo partiendo del método de localización aleatoria de contenedores.

La asignación aleatoria puede realizarse asignando contenedores elegidos de forma aleatoria, estableciendo localizaciones aleatorias a cada contenedor o bien siguiendo una combinación de ambas. Sin embargo, elegir contenedores aleatoriamente no es una práctica que suela realizarse en los puertos, ya que si en dicha elección aleatoria de contenedores se elige algún contenedor que tiene un tiempo de llegada r_i superior a algún otro de la lista de peticiones, la grúa se quedará parada a la espera de que llegue dicho contenedor. De esta manera, el posible tiempo ocioso que puede surgir al asignar los contenedores en un orden aleatorio solo consigue que las operaciones no se ejecuten cuando deberían. Este es el principal motivo que lleva a centrar el planteamiento de este algoritmo aleatorio sobre el hecho de asignar localizaciones aleatorias en lugar de partir de un orden de contenedores aleatorio.

En el algoritmo 1 se plantea el pseudocódigo correspondiente a este método heurístico aleatorio.

El algoritmo de asignación de localizaciones aleatorias requiere unos datos de entrada previos para poder aplicar los distintos pasos. Es necesario disponer previamente de la lista de peticiones de contenedores a localizar, así como también conocer aquellas localizaciones disponibles antes de comenzar.

El primer paso consiste en inicializar la lista de la solución listSol (siguiendo la estructura propuesta en 4.6) y el tiempoActual, y después, ordenar la lista de contenedores según su fecha de llegada r_i, tal como se indica en los puntos 2-4 del algoritmo 1. A continuación, se selecciona uno a uno todos los contenedores según su orden establecido siguiendo la política FIFO, indicado en el bucle del punto 5 del algoritmo.

Este problema utiliza el tiempoActual para indicar cuánto tiempo ha transcurrido hasta ese momento, teniendo en cuenta tanto los tiempos de operación t_{ij} como los tiempos r_i de llegada a E/S. Es necesario que hayan transcurrido las r_i unidades temporales de un contenedor para que se encuentre disponible y la grúa pueda trabajar con él en caso de que lo necesite. Con el fin de controlar si el contenedor ya está disponible, se realiza una comparación del tiempoActual y del r_i asociado al contenedor sobre el que se van a realizar las operaciones de localización: si el tiempo transcurrido es inferior al de llegada a E/S, la grúa se quedará esperando hasta dicho tiempo. Este aspecto se recoge en los puntos 6 y 7 del algoritmo 1.

Después de que se asegure que la grúa ya puede acceder al contenedor i, se asignará una localización aleatoria j de entre todas las disponibles. Antes de localizarlo, hay que comprobar que la ubicación aleatoria j está libre. En caso de que j esté ocupada, no sé

Algoritmo 1: Algoritmo de asignación de localizaciones aleatorias

Require: Conjunto de peticiones de localización de contenedores, localizaciones disponibles
Ensure : listSol

```
1  procedure RandomAlgorithm
2  listSol = ∅;
3  tiempoActual = 0;
4  Ordenar de menor a mayor r_i los contenedores;
5  for cada contenedor i hasta C do
6      if tiempoActual <r_i then
7          tiempoActual = r_i;
8      Seleccionar aleatoriamente localización j;
9      while la localización seleccionada esté ocupada do
10         Asignar nueva localización aleatoria;
11     end
12     Marcar localización como ocupada;
13     Calcular t_ij → tiempoActual = tiempoActual + t_ij;
14     Actualizar tiempo final de i → F_i = tiempoActual;
15     Añadir a listSol el contenedor i, localización asignada j y F_i;
16 end
17 return listSol;
18 end procedure
```

podrá llevar a cabo la operación de localización sobre ese lugar, por lo que será necesario asignar una nueva localización aleatoria. En los pasos 8 a 11 del algoritmo puede observarse este proceso.

Una vez se haya proporcionado de forma aleatoria una localización que esté libre, la grúa tomará el contenedor y lo llevará hasta la posición asignada. Será conveniente que una vez colocado en j, se indique que j está ocupada. De esta forma se asegura que no se intente localizar otro contenedor en una localización ya ocupada. Después de que la grúa haya completado todos los pasos de la operación de localización, se calcula el tiempo t_{ij} correspondiente y se acumula al tiempoActual. Por último, se añade en la listSol el contenedor, el lugar en el que ha sido localizado, y el instante en el que finaliza el proceso de localización del contenedor. Todo este proceso corresponde a los pasos 12 hasta el 17 del algoritmo aleatorio 1.

Finalmente, una vez localizados adecuadamente todos los C contenedores del problema, este algoritmo devolverá una solución factible que será distinta cada vez que se ejecute el algoritmo diseñado, debido al método aleatorio que se le aplica. Esta solución corresponde a la listSol, que muestra el orden, los contenedores, su localización y su tiempo tiempoActual.

Tras describir el algoritmo aleatorio, se formula un ejemplo numérico para facilitar la comprensión de este método. Se parte del ejemplo de estudio propuesto en la sección 4.1.3. Este ejemplo deberá resolver el problema de asignar los 5 contenedores de la tabla 4.3 entre alguna de las 10 localizaciones de la tabla 4.4.

El primer paso que se debe seguir es el de ordenar de menor a mayor r_i, correspondiente a seguir una política FIFO. De los cinco contenedores, el C es el que presenta un r_i, correspondiente a 1 u.t., el siguiente será el contenedor B con una fecha de llegada a E/S de 6 u.t.. Tras comprobar todos los contenedores, se ordenan los contenedores siguiendo este criterio, tal como representa la lista de contenedores ordenados 4.6.

Tabla 4.6: Lista FIFO de contenedores del algoritmo aleatorio.

Contenedor	T_γ	r_i
C	1	1
B	2	6
D	2	8
E	1	18
A	1	29

Tabla de elaboración propia.

Siguiendo el orden de la tabla 4.6 se escoge el contenedor C. Al realizar el paso 6 del algoritmo 1, se comprueba que el F_c corresponde a 0 u.t., y ya que es menor que el r_i, se esperará hasta el instante de 1 u.t. en el que está disponible. A este primer contenedor se le asigna la localización aleatoria $l10$.

Ya que es el primer contenedor, ninguna de las localizaciones estará ocupada, por lo que C puede situarse en la $l10$. Al aplicar la fórmula para el cálculo del tiempo t_{ij} (expresión 4.12), se obtiene que el coste de localizar el contenedor en esta localización es de 26 u.t., y actualizando el tiempo se obtiene que el tiempoActual tras localizar C, equivale a la 1 u.t. que ha estado esperando la grúa más las 26 u.t. de t_{ij}. Se marcará como ocupada la localización $l10$ y se añade el elemento {C, l10, 27} a la listSol.

$$\begin{aligned} t_{ij} = Max\{|5-5|,|0-0|\} + |5-1| + |5-1| + \\ Max\{|5-5|,|0-10|\} + |5-1| + |5-1| = 26\,u.t. \end{aligned} \tag{4.12}$$

El siguiente contenedor es B. Al realizar la comprobación del tiempoActual transcurrido hasta ese momento con el tiempo de llegada de este contenedor se comprueba que el contenedor ha llegado al punto E/S antes de que se completase la primera operación de localización, lo que indica que este contenedor es accesible desde el momento en que la grúa haya terminado de localizar el contenedor C en el bloque. Se le asigna la posición aleatoria $l4$, que se comprueba que no se encuentra ocupada. Se actualiza el tiempoActual sumándole el coste temporal t_{ij} del contenedor B, por lo que tiempoActual = 75 + 27 = 102 u.t., y se asigna esta lista de localización a la listSol con el tiempoActual correspondiente.

Este proceso se repetirá con los tres contenedores restantes, siguiendo la misma estrategia. Se obtiene que el contenedor D se localiza en su primera localización aleatoria asignada, $l7$. Lo mismo ocurre para el E, que se localiza en $l5$. Sin embargo, al contenedor A se le asigna inicialmente la localización $l10$, que al comprobar que ya está ocupada por C, le asigna otra localización aleatoria correspondiente a $l2$ que sí está disponible, por lo que es localizado en esa posición.

Finalmente, todos los contenedores han terminado localizados el algoritmo devuelve la solución factible correspondiente a la listSol {[C, l10, 27], [B, l4, 102], [D, l7, 179], [E, l5, 217], [A, l2, 253]}. El tiempo total que se ha tardado en localizar a los cinco contenedores es de 253 u.t..

4.3.3. Algoritmo de Escalada Simple Iterativo

El método planteado en esta sección se basa en la idea del funcionamiento de un algoritmo de Escalada Simple sobre el que, posteriormente, se realiza una serie de iteraciones

sobre una solución inicial. En cada una de estas iteraciones se compara si el resultado obtenido en la iteración es capaz de minimizar los tiempos más que la mejor solución obtenida hasta ese momento.

Los métodos de escalada o de ascenso de colina, también conocidos por su designación inglesa *Hill Climbing*, son técnicas heurísticas de optimización matemática. Es posible diferenciar dos fases principales que caracterizan el funcionamiento de esta clase de algoritmos. A priori, su coste computacional es superior al del algoritmo de localización aleatoria: tras calcular una solución inicial, tiene una parte iterativa. Este algoritmo se orienta siempre hacia una solución igual o mejor que la inicial.

Este proceso de escalada hace referencia al hecho de, partiendo de un punto de referencia inicial, la solución del problema va *escalando* hacia el objetivo de minimización o maximización, según expone Aparicio et al. [2]. En cada iteración se trata de asignar un valor que se aproxime más que el anterior hacia la minimización/maximización que plantee el problema.

La primera fase corresponde a la primera iteración, con la que se obtiene una listSol inicial a través del método aleatorio. Mientras que la segunda fase consistirá en el resto de iteraciones, que buscan obtener un resultado mejor que la listSolMejor con la que comparan.

Este algoritmo de Escalada Simple iterativa puede considerarse un perfeccionamiento del algoritmo aleatorio 1. Gracias a las repeticiones que realiza, se espera que obtenga unos resultados mejores que este. Al igual que el algoritmo aleatorio, aunque no se pueda que sea óptima, es una solución aceptable.

El pseudocódigo del algoritmo 2 representa la aproximación basada en métodos de Escalada Simple iterativos. Tras inicializar las lista listSol y listMejorSol, el tiempoActual y la iteración, calculamos la lista ordenada según r_i.

En el paso 7 del algoritmo, se indica que realizará el bucle unas MAX_IT veces. En el paso 8 se vuelve a inicializar la listSol para que su valor se reinicie cada iteración. A continuación, se realiza el método aleatorio para obtener una listSol entre los pasos 9 y 20. Si es la primera iteración, guardaremos el valor de listSol en una listMejorSol que servirá de punto de partida para realizar las comparaciones, como indican los pasos 21 y 22. Tras aumentar en 1 el número de iteraciones realizadas, se vuelve a realizar la búsqueda de una nueva listSol.

Para cada iteración, el algoritmo realiza una *escalada* de su solución si obtiene una mejor solución en esta. Comparamos si la nueva solución listSol es capaz de mejorar la listMejorSol en los pasos 23 a 24, y en caso de que la mejore, actualizaremos el valor de listMejorSol a esta nueva solución calculada.

Una vez realizadas todas las iteraciones, la listMejorSol solución almacenará el mejor resultado y será la solución que proporciones este método.

Para realizar una resolución numérica, se parte del ejemplo de estudio de la sección 4.1.3 y se toma un número máximo de iteraciones de 3. La primera fase del algoritmo de Escalada Simple consiste en calcular una solución inicial aplicando el algoritmo de localizaciones aleatorias. Como solución inicial, tomaremos la que calculaba el método aleatorio anterior: {[C, l10, 27], [B, l4, 102], [D, l7, 179], [E, l5, 217], [A, l2, 253]}. Por tanto, la listMejorSol se actualizará a la solución de esta primera iteración .

A continuación, se obtendrá para cada una de las iteraciones que quedan, una listSol que irá comparándose con la listMejorSol almacenada. Según se puede observar en la tabla 4.7, el valor del tiempoActual final obtenido en la iteración 2 equivale a 270 u.t.. Comparándolo con las 253 u.t. de la solución inicial, se aprecia que esta listSol no será

Algoritmo 2: Algoritmo de Escalada Simple Iterativa

Require: Conjunto de peticiones de localización de contenedores, localizaciones disponibles, número de iteraciones máximas MAX_IT
Ensure : listMejorSol

```
1  procedure EscaladaSimpleIterativa
2  listSol = ∅;
3  listMejorSol = ∅;
4  tiempoActual = 0;
5  iteracion = 0;
6  Ordenar de menor a mayor r_i los contenedores;
7  while iteracion < MAX_IT do
8      listSol = ∅;
9      for cada contenedor i hasta C do
10         if tiempoActual <r_i then
11             tiempoActual = r_i;
12         Seleccionar aleatoriamente localización j;
13         while la localización seleccionada esté ocupada do
14             Asignar nueva localización aleatoria;
15         end
16         Marcar localización como ocupada;
17         Calcular t_ij → tiempoActual = tiempoActual + t_ij;
18         Actualizar tiempo final de i → F_i = tiempoActual;
19         Añadir a listSol el contenedor i, localización asignada j y F_i;
20     end
21     if es la primera iteración then
22         listMejorSol = listSol;
23     if listSol < listMejorSol then
24         Actualizar listMejorSol → listMejorSol = listSol;
25     iteracion = iteracion + 1;
26 end
27 return listMejorSol
28 end procedure
```

mejor que la listMejorSol. Como vemos en la tabla 4.7, la iteración 3 ofrecerá la mejor solución, con un tiempoActual de 211 u.t..

1ª iter.	2ª iter.	3ª iter.	Solución final
253	270	211	211

Tabla 4.7: Solución de cada iteración del algoritmo Escalada Simple.

Tabla de elaboración propia.

Tras realizar las cuatro iteraciones y actualizar la lista de la mejor solución, se tiene una listMejorSol con un coste temporal de 211 u.t.. El algoritmo de Escalada Simple ha logrado en el ejemplo numérico mejorar el tiempo total que se ha tardado en realizar todas las operaciones, ya que se ha visto reducido de 253 u.t. iniciales hasta las 211 u.t..

A modo de conclusión, se espera que este algoritmo basado en los procedimientos de escalada sea capaz de encontrar buenas soluciones, que a pesar de que no se garantice que sean óptimas, son eficientes. Sin embargo, el hecho de que realice varias iteraciones

del mismo algoritmo, supone que tenga un coste temporal mayor y que dependerá del número de iteraciones.

4.3.4. Algoritmo VV-Min

Dada la relevancia del concepto de minimización del tiempo en este problema, parece conveniente diseñar un método que asigne los contenedores siguiendo criterios temporales. Para ello, nos alejamos de los métodos aleatorios planteados anteriormente en 4.3.2 y 4.3.3 y buscaremos una heurística en la que el tiempo no solo sea el concepto a minimizar, sino que sirva como criterio previo para asignar localizaciones.

Este método ha sido diseñado partiendo de otras aproximaciones como los algoritmos de vecinos más próximos y los voraces. La denominación de este algoritmo hace referencia a estas **V**(ecinos), **V**(oraz) y el **min**imizar los tiempos.

Respecto a los algoritmos de vecinos más próximos, también llamados K-NN, ha servido de inspiración el concepto de proximidad. Los K-NN son unos algoritmos complejos, basado en el aprendizaje supervisado y en la clasificación de los elementos según un criterio de proximidad. Aunque este problema es complejo y no ha sido implementado directamente en nuestro algoritmo, sí que ha ayudado en su desarrollo. En nuestro problema, tomamos el concepto de *proximidad* temporal: cuanto menor sea el tiempo de localización, más próximo estará al tiempo total mínimo.

Los algoritmos voraces, también conocidos como *greedy*, se utilizan para problemas de optimización. Aunque no garantizan la solución óptima, eligen en cada paso la mejor solución a nivel local. En nuestro algoritmo, calcularemos la lista con todos los costes temporales t_{ij} de las localizaciones disponibles, y se asignará aquel con menor t_{ij} de esa lista.

El algoritmo 3 muestra el pseudocódigo de la heurística VV-Min. Primero, iniciamos las dos listas y el tiempoActual. En el paso 2 se ordenan los contenedores siguiendo el criterio FIFO. Tras esto, se calculará la listaTiempos con los t_{ij} de todas las localizaciones. A partir de la listaTiempo, se asignan las localizaciones disponibles que minimizan el coste temporal de cada contenedor. Por último, se marca la localización como ocupada, y actualizamos la listSol.

Mediante el uso del ejemplo de estudio 4.1.3, vamos a realizar una comprobación numérica del algoritmo VV-Min.

Tras ordenar la lista de contenedores según su r_i, tomamos primero el contenedor C. La grúa esperará hasta el instante 1 en el que llegue dicho contenedor. Se calcula la listaTiempos, que incluirá los t_{ij} de todas las localizaciones disponibles. La listaTiempos para C corresponde a la indicada en la tabla 4.8, y sobre esta se elegirá la localización que minimice el t_{ij}. En este caso, se asignará al contenedor C la $l1$, cuyo coste temporal de 21 u.t. es el menor de toda la lista. A continuación se marca como ocupada $l1$. Se actualiza el tiempoActual a 22 u.t.. Este mismo proceso se realizará sobre todos los contenedores restantes.

La solución mediante el método VV-Min calcula la siguiente listSol: {[C, l1, 22], [B, l6, 94], [D, l5, 155], [E, l2, 191], [A, l8, 219]}. La solución proporcionada por el algoritmo VV-Min presenta un coste de 219 u.t.. Esta solución es mejor que la proporcionada por los otros dos métodos.

Algoritmo 3: Algoritmo VV-Min

```
Require: Planificación de contenedores, localizaciones disponibles
Ensure : listSol
procedure AlgoritmoVV-MIN
listSol = ∅;
listaTiempos = ∅;
tiempoActual = 0;
Ordenar de menor a mayor r_i los contenedores;
for todo contenedor i do
    for toda localización j do
        Calcular t_ij correspondiente;
        Añadir t_ij a listaTiempos;
    end
end
for contenedor i hasta C do
    if tiempoActual <r_i then
        tiempoActual = r_i;
    Asignar a i la localización de mínimo t_ij de listaTiempos;
    Marcar la localización como Ocupada;
    Actualizar tiempoActual → tiempoActual = tiempoActual + t_ij;
    Actualizar tiempo final de i → F_i = tiempoActual;
    Añadir a listSol el contenedor i, localización asignada j y F_i;
end
return listSol;
end procedure
```

Localizaciones	l1	l2	l3	l4	l5	l6	l7	l8	l9	l10
Contenedor C	21	22	28	27	28	39	26	22	23	26

Tabla 4.8: Lista de los tiempos t_{ij} correspondiente al primer contenedor del ejemplo de estudio.

Tabla de elaboración propia.

4.4 Experimentos computacionales

Tras exponer los distintos métodos diseñados para la localización de contenedores en un bloque en la sección 4.3, se realizarán pruebas sobre el algoritmo de localización aleatoria, el algoritmo de Escalada Simple Iterativo y el algoritmo VV-Minimización. Explicaremos cómo se han generado las instancias con las que realizar las pruebas y analizaremos los resultados obtenidos a partir de estas.

4.4.1. Generación de instancias

Para realizar las pruebas, se han generado aleatoriamente un total de 25 instancias. Estas, corresponden a 5 réplicas distintas generadas para cada tamaño correspondiente a 5, 10, 30, 40 o 50 contenedores. Han sido nombradas como Ins_C1_NCA_B, indicando con A el número de contenedores de la instancia, y con B a cuál de las 5 réplicas corresponde. La instancia Ins_C1_NC30_4 corresponderá a la cuarta réplica generada para una petición de 30 contenedores.

Los tipos T_γ del contenedor se asignarán aleatoriamente entre tipo 1 y tipo 2. El tiempo de llegada al punto E/S (r_i) corresponderá a un valor aleatorio comprendido en el intervalo [0, (nº de contenedores · nº bahías) · 0,1].

El número total de localizaciones se calcula multiplicando el número total de contenedores por un parámetro que varía aleatoriamente entre 2 y 3. Para indicar las coordenadas de cada localización, se genera aleatoriamente dentro del rango establecido de filas [1, 10], bahías [1 a 42] y alturas [1 a 4].

4.4.2. Resultados

Para la realización de los experimentos computacionales, se han implementado los tres algoritmos utilizando Python 3.8.3 en un ordenador con sistema operativo Windows 10, con RAM de 8GB y un procesador Intel © Core ™ i7-3770 de 3,40 GHz.

En este apartado se medirá el desempeño o bondad de los tres algoritmos propuestos mediante el análisis de la desviación porcentual relativa, también conocida como RPD (*Relative Percentage Deviation*), con la que calcularemos la desviación de cada uno de los tres algoritmos respecto a la solución óptimo obtenida entre todos ellos. La fórmula del RPD corresponde a:

$$RPD = \frac{s - s^*}{s^*} \cdot 100 \tag{4.13}$$

En esta ecuación, s^* representa la mejor solución obtenida en la instancia de entre todos los métodos planteados, es decir, s^* es el menor tiempo total de realizar todas las operaciones de localización obtenido entre los tres algoritmos. La s corresponde a la solución obtenida para la instancia con un algoritmo en concreto. Calculamos esta solución mediante los tres algoritmos diseñados en este trabajo.

Podemos agrupar las instancias según el tamaño de contenedores. Por una parte, habrá un grupo de instancias pequeñas correspondiente a las instancias de 5 y 10 contenedores. Las correspondientes a 30, 40 y 50 contenedores las consideraremos instancias grandes.

En la tabla 4.9 se muestran los resultados del grupo de instancias pequeñas y la tabla 4.10 representa los resultados del grupo de instancias grandes. La primera columna de las tablas indica la instancia de datos seleccionada. La segunda columna recoge el valor s^* de los tres algoritmos. Las siguientes columnas corresponden al RPD calculado según la fórmula 4.13, a la solución s obtenida y al tiempo de cómputo t medido en milisegundos la instancia de cada uno de los tres algoritmos: Aleatorio, Escalada Simple con 100 iteraciones y VV-Min.

El algoritmo aleatorio y el VV-Min se ejecutan una vez. El algoritmo de Escala Simple se ejecutará 100 veces, tantas como iteraciones.

Debido a que no poseemos soluciones factibles además de las generadas por nuestros métodos, puede resultar difícil comparar la idoneidad de los resultados obtenidos. Observando las dos tablas, se obtiene que el algoritmo VV-Min obtiene la mejor solución s en todas las instancias del problema. Por lo tanto, partiremos de este algoritmo para realizar las comparaciones tanto del aleatorio como del Escalada Simple Iterativo.

Primero, analizaremos la tabla correspondiente a las instancias pequeñas 4.9. Se comprueba cómo la s del VV-Min es equivalente a la columna s^*. A priori, parece que la heurística detrás de este algoritmo puede ser un punto de partida interesante. El algoritmo aleatorio es, por término medio, un 25,07 % peor que el VV-Min. Este porcentaje disminuye a 21,89 % en el caso del algoritmo de Escalada Simple.

		ALEATORIO			ESCALADA SIMPLE			VV-MIN		
Instancia	s*	RPD	s	t	RPD	s	t	RPD	s	t
Ins_C1_NC5_1	330	47,88	488	0,32	10,61	365	35,02	0,00 %	330	0,40
Ins_C1_NC5_2	212	17,92	250	0,32	40,57	298	33,98	0,00	212	0,44
Ins_C1_NC5_3	477	43,61	685	0,48	3,77	495	32,98	0,00	477	0,45
Ins_C1_NC5_4	347	4,61	363	0,42	13,83	395	34,04	0,00	347	0,38
Ins_C1_NC5_5	477	2,10	487	0,44	2,52	489	33,98	0,00	477	0,44
Ins_C1_NC10_1	386	50,78	582	0,55	51,30	584	39,97	0,00	386	0,66
Ins_C1_NC10_2	596	26,85	756	0,47	19,46	712	40,97	0,00	596	0,92
Ins_C1_NC10_3	593	32,38	785	0,43	46,88	871	40,97	0,00	593	0,97
Ins_C1_NC10_4	719	13,49	816	0,47	7,51	773	39,97	0,00	719	0,94
Ins_C1_NC10_5	747	11,11	830	0,43	22,49	915	41,97	0,00	747	0,67
Promedio		25,07	604,20	0,43	21,89	589,70	37,39	0,00	488,40	0,63

Tabla 4.9: RPD y tiempos de computación de los grupos de instancias pequeñas

Respecto al tiempo de cómputo t de la tabla 4.9 se observa que, además de que el VV-Min es el algoritmo que mejores resultados proporciona, también aporta un buen tiempo de cómputo. El tiempo promedio que tarda en calcular la solución es de 0,626 milisegundos. Cabe destacar que el algoritmos aleatorio, a pesar de no ser capaz de ofrecer unos resultados mejores que VV-Min, es el algoritmo que más rápido se ejecuta.

Es sencillo observar el impacto que tiene el uso de métodos iterativos en relación con el tiempo de cómputo. Las 100 iteraciones que debe realizar el Escalada Simple, provocan un t promedio de 37,39 ms, considerablemente superior que los otros dos. La diferencia temporal entre este método y el resto es notable. Sin embargo, debe indicarse que estos tiempos de cómputo son insustanciales, especialmente en instancias de estos tamaños. Los tiempos no tienen mucho peso, ya que con un número de contenedores pequeño, la solución se obtiene en un período de tiempo prácticamente instantáneo. Ya que el tiempo de cómputo no es muy relevante en las instancias pequeñas, se concluye que lo mejor será utilizar el algoritmo VV-Min para problemas con un número de contenedores similar.

Posteriormente, se analizan los datos obtenidos en la tabla de las instancias grandes 4.10. Las diferencias entre la eficiencia de los distintos métodos son mucho más apreciables en este grupo.

La mejor solución la sigue ofreciendo el método VV-Min, con un tiempo total de localización promedio de 2370,4 ms. Sin embargo, la diferencia entre los métodos aleatorio y Escalada Simple es notablemente mayor. El algoritmo aleatorio aporta soluciones que son un 121 % peores, por término medio, que las ofrecidas por el VV-Min. Por otra parte, las soluciones del método Escalada Simple son un 32 % peores que las del VV-Min. La gran diferencia entre estas dos RPD parece indicar que para instancias de mayor tamaño, el aleatorio no es tan eficaz como en la tabla 4.9. El s de los aleatorios se encuentra en un promedio de 5253 u.t.. Gracias a las iteraciones de la Escalada Simple, se logran soluciones promedio de 3118 u.t.. Y por último, el algoritmo VV-Min, puede obtener una solución promedio de 2370,4 u.t. gracias a los criterios de minimización.

Desde el punto de vista del cómputo, hay que destacar que el t del VV-Min ha experimentado un aumento considerable, llegando a tener un tiempo promedio de 12,6 ms. Sin embargo, parece que el aleatorio no se ha visto afectado por el aumento de contenedores, ya que ofrece un tiempo del mismo orden que con pocos contenedores. La heurística que hay detrás del algoritmo VV-Min parece ser la causante de este aumento de tiempo: mientras el aleatorio no tiene que realizar ningún cálculo para asignar localizaciones, el VV-Min debe realizar las tablas de t_{ij} para poder establecer los tiempos mínimos.

Mientras que los t promedio del aleatorio y de Escalada simple han crecido a un ritmo similar, los algoritmos VV-Min presentan tiempos de cómputo promedio mucho más elevados, pasando de 0,63 a 12,6 ms en las instancias grandes. Esto ocurre debido a los cálculos del t_{ij} para cada contenedor y localización que realiza el método VV-Min, con los que poder seguir los criterios de minimización. Por otra parte, los otros dos métodos se mantienen con unos tiempos inferiores ya que el único criterio que siguen es la aleatoriedad, cuyo coste de cómputo es muy inferior al de los algoritmos VV-Min.

Si se busca alcanzar la mejor solución, independientemente de su tiempo de cómputo, la mejor alternativa seguirá siendo el método VV-Min. La siguiente alternativa, sería la escalada simple iterativa. Aunque el aleatorio sea el que más rápidamente se ejecuta, no logra compensar unas soluciones un 121 % peores de promedio.

		ALEATORIO			**ESCALADA SIMPLE**			**VV-Min**		
Instancias	**s***	**RPD**	**s**	**t**	**RPD**	**s**	**t**	**RPD**	**s**	**t**
Ins_C1_NC30_1	2032	115,80	4385	0,93	29,92	2640	76,95	0	2032	3,89
Ins_C1_NC30_2	1900	92,89	3665	0,59	21,68	2312	71,99	0	1900	3,88
Ins_C1_NC30_3	1943	28,20	2491	0,54	26,81	2464	68,96	0	1943	7,17
Ins_C1_NC30_4	1733	99,71	3461	0,55	30,29	2258	74,95	0	1733	3,95
Ins_C1_NC30_5	1795	89,25	3397	0,63	28,75	2311	71,96	0	1795	3,85
Ins_C1_NC40_1	2130	183,90	6047	0,80	41,78	3020	85,23	0	2130	13,37
Ins_C1_NC40_2	2310	37,45	3175	0,74	30,26	3009	83,95	0	2310	15,35
Ins_C1_NC40_3	2195	178,59	6115	0,69	33,30	2926	90,94	0	2195	7,02
Ins_C1_NC40_4	2388	130,32	5500	0,59	32,75	3170	90,94	0	2388	7,62
Ins_C1_NC40_5	2127	211,28	6621	0,74	45,79	3101	87,95	0	2127	13,69
Ins_C1_NC50_1	2771	111,95	5873	0,80	37,28	3804	107,62	0	2771	24,55
Ins_C1_NC50_2	3387	36,32	4617	0,84	21,49	4115	112,93	0	3387	24,37
Ins_C1_NC50_3	2936	29,87	3813	0,86	36,65	4012	104,94	0	2936	24,04
Ins_C1_NC50_4	2898	264,84	10573	0,78	33,61	3872	105,94	0	2898	11,86
Ins_C1_NC50_5	3012	200,93	9064	0,87	24,90	3762	107,93	0	3012	24,34
Promedio		121	5253	0,73	32	3118	89,55	0,00	2370,4	12,60

Tabla 4.10: RPD y tiempos de computación de los grupos de instancias grandes

En la figura 4.4 puede apreciarse cómo evolucionan las soluciones s de las tablas anteriores. Puede identificarse como el algoritmo aleatorio dista mucho del resto cuando las instancias son grandes.

Gracias al gráfico de dispersión de la figura 4.5, puede observarse cómo se relaciona el número de contenedores de las instancias con los resultados obtenidos. Para las instancias de 5 y 10 contenedores, no puede afirmarse cuál de los dos algoritmos es mejor o peor. Sin embargo, a partir de las instancias de 30 contenedores, el algoritmo aleatorio obtiene unos resultados considerablemente peores. La Escalada Simple aporta, por tanto, mejores resultados con respecto a la aleatoria para un número elevado de contenedores.

Ya que en los problemas reales de localización no se trabaja con un número tan reducido de contenedores, puede afirmarse que utilizar un algoritmo aleatorio o de Escalada simple no resultaría conveniente a la hora de gestionar las localizaciones del puerto. La alternativa de VV-Min es la que mejores resultados proporciona.

A partir de estos resultado, se puede extraer la idea de que dotar a los métodos de cierta inteligencia, ayuda a obtener unos mejores resultados. Mientras que los aleatorios se basan en la fuerza bruta, asignando localizaciones con el único criterio de que sean aleatorias, los VV-Min asignan de forma inteligente las localizaciones y proporcionan estos mejores resultados.

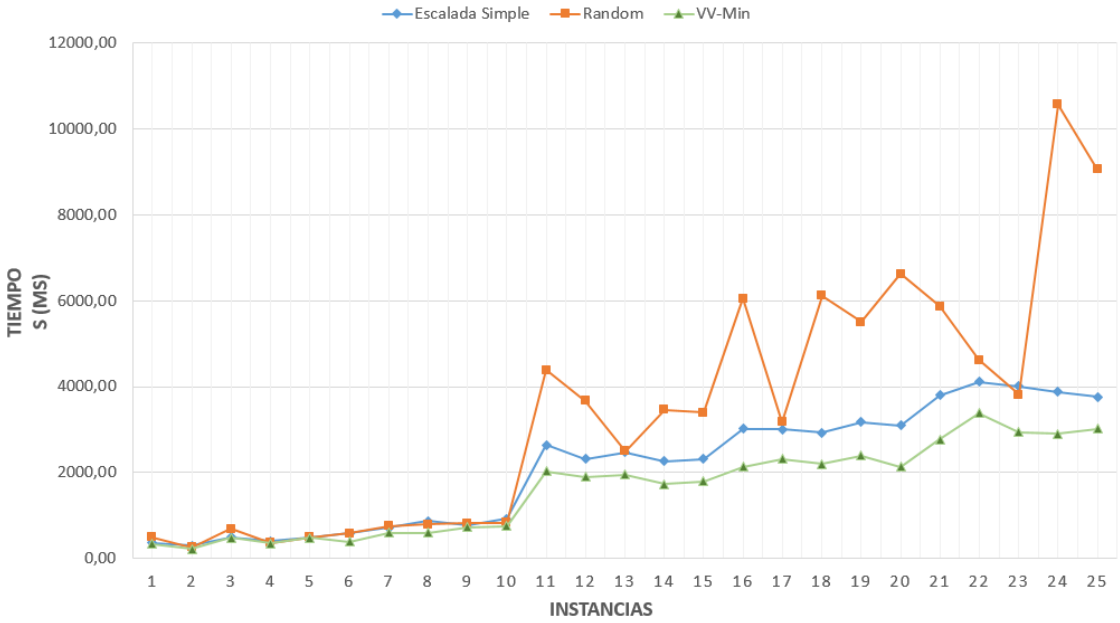


Figura 4.4: Evolución de los tiempos totales s de las instancias.
Figura de elaboración propia.

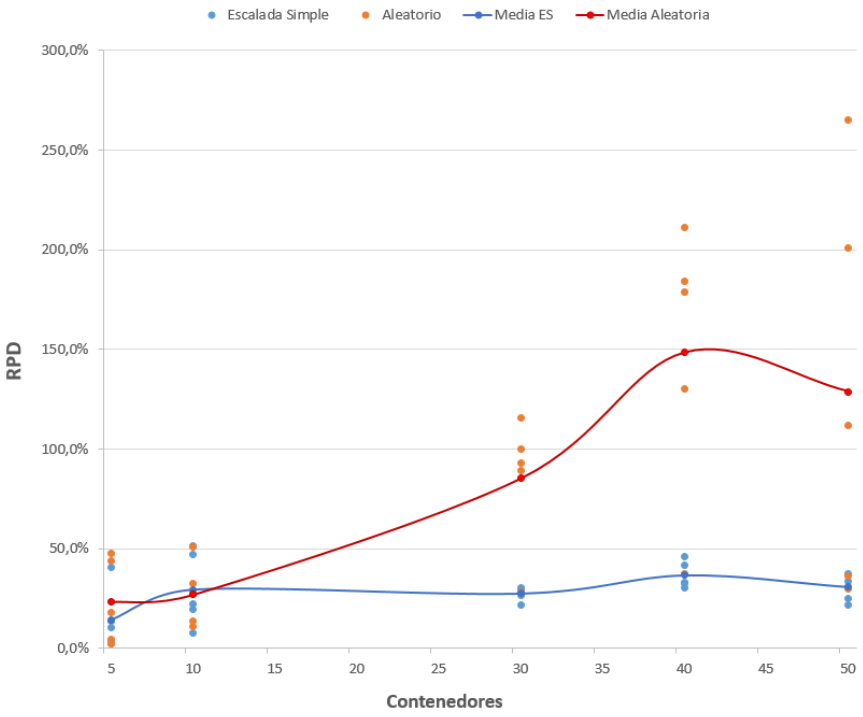


Figura 4.5: Gráfico de dispersión del RPD de los algoritmos aleatorio y Escalada Simple.
Figura de elaboración propia.

A partir de la figura 4.6 se puede observar el tiempo de cómputo promedio de las instancias con el mismo número de contenedores. Aunque el objetivo de los métodos de este trabajo no es el de minimizar el tiempo de cómputo, puede resultar relevante conocer cuánto tarda en ejecutarse una solución del algoritmo. En la gráfica puede observarse que la diferencia entre los t de los métodos aleatorio y los VV-Min es pequeña, aunque van diferenciándose más conforme hay más contenedores. El método de Escalada Simple es

el más costoso para todas las instancias del problema, y su crecimiento parece estar ligado al tamaño de las instancias.

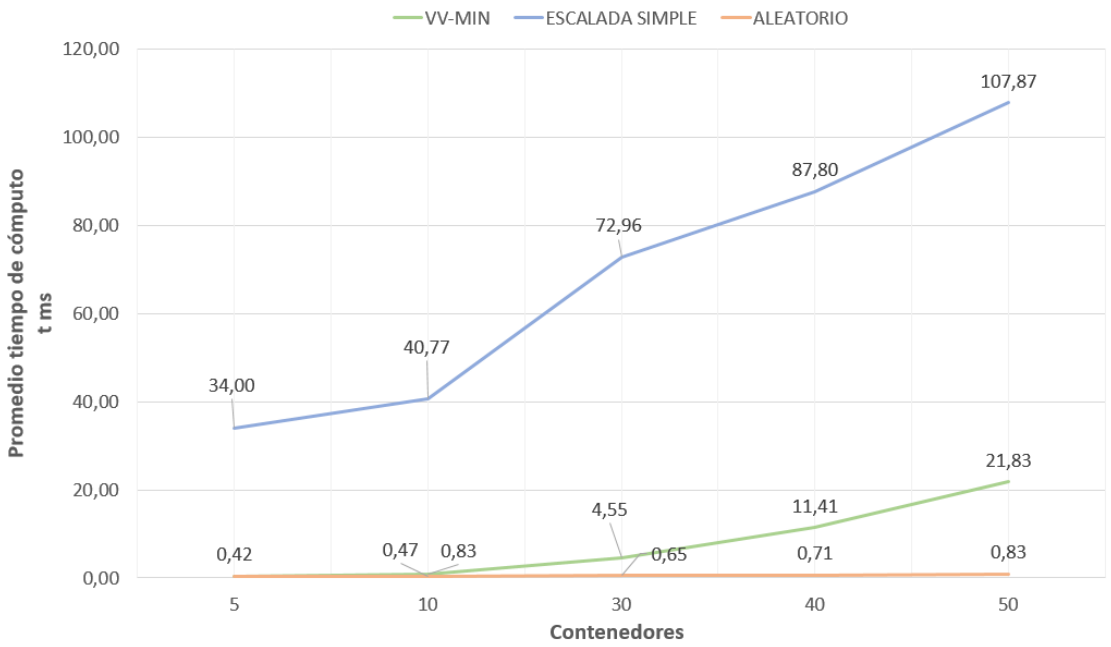


Figura 4.6: Tiempos de cómputo promedio de las instancias de cada tamaño de contenedores. Figura de elaboración propia.

A través de los resultados obtenidos mediante estos métodos, es destacable recordar la importancia que tienen las heurísticas a la hora de optimizar problemas. Tras este análisis de las soluciones obtenidas, se ha podido observar como el dotar a los algoritmos de unas buenas técnicas heurísticas, ayuda a alcanzar mejores soluciones. La heurística implementada en el algoritmo VV-Min ha permitido conseguir unos mejores resultados que los métodos aleatorios y de Escalada Simple.

CAPÍTULO 5
Revisión bibliográfica

En este capítulo se presenta cuáles han sido las principales fuentes bibliográficas en las que se ha apoyado este Trabajo de Fin de Grado, relacionadas especialmente con el problema de localización y asignación de espacio de los contenedores marítimos y los distintos enfoques que plantea.

El contexto histórico del transporte marítimo y el concepto de contenedor se exponen ampliamente en el libro de M. Levinson [21], donde se detallan diversos factores de la relevancia de las terminales de contenedores en la actualidad.

Los datos económicos relacionados con previsiones y el desarrollo del comercio marítimo se apoyan fundamentalmente en los estudios del Parlamento Europeo [4] y en los informes sobre la evolución de los puertos y del transporte marítimo que elabora anualmente la UNCTAD [27, 28]. Estos informes también aportan una visión de la situación general del volumen de los intercambios, de las mercancías y rutas comerciales o de la flota. Además, el libro de D. Song [24] explica cómo es el funcionamiento de la logística portuaria y sus expectativas de crecimiento en los próximos años.

Los autores de los artículos [12, 31, 1, 8, 30, 23, 20] presentan el escenario actual de los *smart ports* y cuál es su influencia en los puertos, al mismo tiempo que relacionan el desarrollo sostenible y el impacto medioambiental con el económico y exponen diversas características de los *green ports*.

Dirk Steenken, Robert Stahlbock y Stefan Voß [26] exponen cómo es una terminal portuaria de contenedores y determinan las actividades fundamentales que se desarrollan dentro de estos puerto, cubriendo un gran número de referencias que engloban distintos enfoques y problemáticas que van desde la estructura del puerto hasta problemas del muelle o de las grúas. También se hace referencia a investigaciones relacionadas con la planificación y optimización de operaciones portuarias.

En el artículo de Stahlbock et al. [25], considerado por los propios autores como una extensión y actualización del artículo [26], se cubren distintos trabajos relacionados con la metodología y operaciones que se aplican en los puertos. En este artículo también se introduce estudios sobre estrategias de decisión relacionadas con los problemas de minimización de retardos en la carga y descarga de contenedores.

Carlo et al. [5] se realiza una revisión de las configuraciones europeas y asiáticas del diseño del patio de contenedores y las principales operaciones portuarias como la asignación de bloques, la localización o el establecer las rutas de las grúas de patio. Para cada actividad del puerto, el trabajo expone diversos estudios influyentes realizados sobre estas problemáticas y apuntan posibles líneas de investigación futuras.

La problemática de la localización dentro de un bloque ha sido tratada por diversos autores en los últimos años. Kim et al. [18] propone un modelo de programación diná-

mica en el que se pre-asigna localizaciones para los contenedores de exportación. En el trabajo de Dekker et al. [10] se estudian distintas políticas de apilamiento dentro de los bloque durante la localización, teniendo en cuenta criterios como el tamaño, el peso o la altura máxima que se puede alcanzar en el bloque. En el artículo de Chen et al. [7] divide al problema de localización en dos partes: una primera fase relacionada con la asignación del contenedor a un bloque, con el objetivo de reducir los tiempos de espera de los vehículos de patio; y una segunda fase encargada de localizar la posición exacta en la que se ubica cada contenedor, centrándose en reducir el número de *rehandles* en las posteriores operaciones.

A partir del informe de Bauer [3] sobre los problemas de asignación con restricciones, se clasifica el problema de localización de contenedores como una asignación asimétrica y se amplia sobre cómo diseñar e implementar esta clase de algoritmos.

Las bases teóricas de los métodos heurísticos desarrollados en el trabajo se ha desarrollado a partir de la definición de la heurística que se establece en [22] y [29], dónde se expone el concepto de heurística y realiza aproximaciones esquematizadas sobre cómo se pueden establecer algunos métodos para problemas comunes como la asignación.

Otras aportación a considerar son el informe de V. Galle [14] y A. Ferrer [13]. En el primero se tratan modelos de optimización de problemas relacionados con el patio de carga como la relocalización de contenedores o la planificación de la grúa de patio para este problema. En A. Ferrer desarrolla modelos de planificación para movimiento de grúas duales en el patio y diseña una serie de algoritmos genéticos sobre esta cuestión.

CAPÍTULO 6
Conclusiones y recomendaciones para investigaciones futuras

En este trabajo, se ha expuesto el contexto histórico y la relevancia que tiene el comercio marítimo a nivel mundial. Se ha remarcado el concepto del contenedor, y su papel clave al desarrollar aproximaciones a los problemas de las terminales.

También se ha presentado una visión general del funcionamiento de las terminales portuarias y de las distintas operaciones que se realizan en ellas. Concretamente, se ha analizado en profundidad el problema de la localización de contenedores en un bloque, y la relevancia que tiene el ser capaces de optimizar los tiempos de estas operaciones.

Gran parte del trabajo se ha sustentado en lo aprendido en investigación operativa. Además, se ha aplicado la filosofía de los modelos de asignación para un caso particular del problema abordado que ha servido de aproximación.

Se han desarrollado tres heurísticas capaces de resolver un problema real, con unas características muy concretas, en unos tiempos de computación muy rápidos. Tras el análisis del RPD, se concluye que, desde un punto de vista de optimización de operaciones en un puerto, el algoritmo más útil a implementar sería el VV-Min, ya que ofrece buenas soluciones, mejores que el resto de métodos, dentro de unos tiempos aceptables. Este análisis de desempeño, ha destacado cómo el introducir adecuadamente información en las heurísticas, provoca que funcionen mejor.

Gracias a esto, se demuestra que la aplicación de heurísticas a problemas de localización de contenedores es una herramienta muy interesante que puede ayudar a los trabajadores del puerto a la hora de la toma de decisiones de una manera racional. A través de una heurística adecuadamente diseñada, una terminal portuaria será capaz de gestionar de manera inteligente unos recursos escasos y valiosos (tiempo, recursos económicos y trabajadores) de forma que le proporcionan un mayor nivel de competitividad.

A fin de exponer el comportamiento de estos algoritmos de una forma mucho más ilustrativa, se ha utilizado un mismo ejemplo numérico en todos los métodos. A partir de este, se han extraído conclusiones de cada uno de los métodos.

Al tratarse de un trabajo de final de grado, hay ciertos puntos que no se han considerado a la hora de desarrollar estos métodos. Sin embargo, nuevas líneas de investigación podrían extender los métodos presentados en este trabajo.

Respecto al algoritmo aleatorio, resultaría interesante tratar de perfeccionarlo mediante la aplicación de metaheurísticas. De esta forma, no partirían de unos resultados completamente aleatorios, sino que podría establecerse, por ejemplo, distintos pesos según aporten un mejor resultado.

El introducir metaheurísticas como los algoritmos GRASP o los algoritmo genéticos, podrían guiar hacia unas soluciones óptimas del problema de localización. Además, proporcionarían una mayor flexibilidad a la hora de plantear nuevos métodos.

Bibliografía

[1] Despina Anastasopoulou, Chrysostomos Stylios, and Stavros Kolios. How will Greek ports become green ports? Technical report, TEI de Epirus, Universidad de Ioannina, 2011.

[2] Amador Aparicio De La Fuente and Alfredo Javier Gonel Crespo. Estudio de los algoritmos de mejora iterativa, 2005.

[3] Ulrich Bauer. Assignment Problem with Constraints - Zuordnungsproblem mit Nebenbedingungen. Technical report, Technische Universität München, Munich, 2005.

[4] Buck Consultants International, Karel Vanroye, and Bart van Mol. La evolución del papel de los puertos marítimos en el ámbito de la logística marítima mundial. Technical report, Parlamento Europeo, Bruselas, oct 2009.

[5] Héctor J. Carlo, Iris F.A. Vis, and Kees Jan Roodbergen. Storage yard operations in container terminals: Literature overview, trends, and research directions. *European Journal of Operational Research*, 235(2):412–430, jun 2014.

[6] Marco Caserta, Silvia Schwarze, and Stefan Voß. A mathematical formulation and complexity considerations for the blocks relocation problem. *European Journal of Operational Research*, 219(1):96–104, may 2012.

[7] Lu Chen and Zhiqiang Lu. The storage location assignment problem for outbound containers in a maritime terminal. *International Journal of Production Economics*, 135(1):73–80, jan 2012.

[8] Rong-Her Chiu, Le-Hui Lin, and Shih-Chan Ting. Evaluation of Green Port Factors and Performance: A Fuzzy AHP Analysis. Technical report, National Taiwan Ocean University, Taiwan, 2014.

[9] Laidy De Armas, Rafael Bello, and Carlos Morell. Métodos de optimización aplicados al problema de apilamiento de contenedores. *Ciencias Matemáticas*, 31, 2017.

[10] Rommert Dekker, Patrick Voogd, and Eelco Van Asperen. Advanced methods for container stacking. In *Container Terminals and Cargo Systems: Design, Operations Management, and Logistics Control Issues*, pages 131–154. Springer Berlin Heidelberg, 2007.

[11] Ding Ding and Mabel C. Chou. Stowage planning for container ships: A heuristic algorithm to reduce the number of shifts. *European Journal of Operational Research*, 246(1):242–249, oct 2015.

[12] Kaoutar Douaioui, Mouhsene Fri, Charif Mabrouki, and El Alami Semma. Smart port: Design and perspectives. In *Proceedings - GOL 2018: 4th IEEE International Conference on Logistics Operations Management*, pages 1–6. Institute of Electrical and Electronics Engineers Inc., jun 2018.

[13] Avelino Ferrer Ahullana. *Optimización de los movimientos de dos grúas en el patio de contenedores de una terminal portuaria*. PhD thesis, Universidad Politécnica de Valencia, 2017.

[14] Virgile Galle. Optimization Models and Methods for Storage Yard Operations in Maritime Container Terminals. Technical report, Massachusetts Institute of Technology, 2018.

[15] Giovanni Giallombardo, Luigi Moccia, Matteo Salani, and Ilaria Vacca. Modeling and solving the Tactical Berth Allocation Problem. *Transportation Research Part B: Methodological*, 44(2):232–245, feb 2010.

[16] Roberto Guerra-Olivares, Neale R. Smith, Rosa G. González-Ramírez, Ezequiel García-Mendoza, and Leopoldo Eduardo Cárdenas-Barrón. A heuristic procedure for the outbound container space assignment problem for small and midsize maritime terminals. *International Journal of Machine Learning and Cybernetics*, 9(10):1719–1732, oct 2018.

[17] Kap Hwan Kim and Ki Young Kim. Optimal price schedules for storage of inbound containers. *Transportation Research Part B: Methodological*, 41(8):892–905, oct 2007.

[18] Ki Young Kim and Kap Hwan Kim. Heuristic algorithms for routing yard-side equipment for minimizing loading times in container terminals. *Naval Research Logistics*, 50(5):498–514, aug 2003.

[19] Byung Kwon Lee and Kap Hwan Kim. Optimizing the yard layout in container terminals. *OR Spectrum*, 35(2):363–398, jun 2013.

[20] Taehee Lee and Hyunjeong Nam. A Study on Green Shipping in Major Countries: In the View of Shipyards, Shipping Companies, Ports, and Policies. *Asian Journal of Shipping and Logistics*, 33(4):253–262, dec 2017.

[21] Marc Levinson. *The Box: How the Shipping Container Made the World Smaller and the World Economy Bigger* . Princeton University Press, 2006.

[22] Rafael Martí. Procedimientos Metaheurísticos en Optimización Combinatoria. Technical report, Universidad de Valencia, 2003.

[23] Cherdvong Saengsupavanich, Nowarat Coowanitwong, Wenresti G. Gallardo, and Chanachai Lertsuchatavanich. Environmental performance evaluation of an industrial port and estate: ISO14001, port state control-derived indicators. *Journal of Cleaner Production*, 17(2):154–161, jan 2009.

[24] Dong-Wook Song and Photis M Panayides. *Maritime Logistics: A Guide to Contemporary Shipping and Port Management*. Kogan Page Limited, 2015.

[25] Robert Stahlbock and Stefan Voß. Operations research at container terminals: A literature update. *OR Spectrum*, 30(1):1–52, jan 2008.

[26] Dirk Steenken, Stefan Voß, and Robert Stahlbock. Container terminal operation and operations research - A classification and literature review, 2004.

[27] UNCTAD. 50 years of Review of Maritime Transport, 1968–2018: Reflecting on the past, exploring the future. Technical report, Naciones Unidas, 2018.

[28] UNCTAD. Review of Maritime Transport 2019. Technical report, Naciones Unidas, 2019.

[29] Aitana Vidal Esmorís. Algoritmos heurísticos en optimización. Technical report, Universidad de Santiago de Compostela, 2013.

[30] Yi Chih Yang and Wei Min Chang. Impacts of electric rubber-tired gantries on green port performance. *Research in Transportation Business and Management*, 8:67–76, oct 2013.

[31] Yongsheng Yang, Meisu Zhong, Haiqing Yao, Fang Yu, Xiuwen Fu, and Octavian Postolache. Internet of things for smart ports: Technologies and challenges. *IEEE Instrumentation and Measurement Magazine*, 21(1):34–43, feb 2018.

APÉNDICE A
Código de los algoritmos

```
import  csv
import random
from time import time

def elimina_comas(listaTrimeada):
    nueva = []
    for dato in listaTrimeada:
        if dato != ",":
            nueva.append(int(dato))
    return nueva
tINI = time()

for i in range(1):
  numCont = 0
  free_loc_param = 0
  numLocTotales = 0
  posIO_Tierra = []
  posIO_Mar = []
  listaContenedores = []
  listaLocalizaciones = []
  solucionAlgoritmo = []
  tTotal_Localizacion = 0

  with open('C:/Users/Vicente/Desktop/TFG/ADE/programacion/instancia del
    EJEMPLO.txt') as datos:

    for linea in datos:
      if linea.startswith("number of containers"):
        instanciaNumCont = next(datos).strip()
        instanciaNumCont = instanciaNumCont.replace("\t",",")
        numCont = instanciaNumCont
      if linea.startswith("Free Locations Parameter"):
        instanciaFL = next(datos).strip()
        instanciaFL = instanciaFL.replace("\t",",")
        free_loc_param = instanciaFL
      if linea.startswith("Crane"):
        instanciaPosGrua = next(datos).strip()
        instanciaPosGrua = instanciaPosGrua.replace("\t",",")
        instanciaPosGrua = list(instanciaPosGrua)
        instanciaPosGrua = elimina_comas(instanciaPosGrua)
        posGrua = instanciaPosGrua
        pass
      if linea.startswith("I/O Land"):
        instanciaIO_tierra = next(datos)
        instanciaIO_tierra = next(datos).strip()
        instanciaIO_tierra = instanciaIO_tierra.replace("\t",",")
        instanciaIO_tierra = list(map(int,instanciaIO_tierra.split(",")))
```

```
47          posIO_Tierra = instanciaIO_tierra
48          pass
49        if linea.startswith("I/O Sea"):
50          instanciaIO_mar = next(datos)
51          instanciaIO_mar = next(datos).strip()
52          instanciaIO_mar = instanciaIO_mar.replace("\t",",")
53          instanciaIO_mar = list(map(int,instanciaIO_mar.split(",")))
54          posIO_Mar = instanciaIO_mar
55          pass
56        if linea.startswith("#Type"):
57          for veces in range(int(numCont)):
58            instanciaCont = next(datos).strip()
59            instanciaCont = instanciaCont.replace("\t",",")
60            instanciaCont = list(map(int,instanciaCont.split(",")))
61            listaContenedores.append(instanciaCont)
62            pass
63          pass
64        if linea.startswith("Free Locations:"):
65          instanciaLoc = next(datos)
66          numLocTotales=int(numCont)*int(free_loc_param)
67          for veces in range(numLocTotales):
68            instanciaLoc = next(datos).strip()
69            instanciaLoc = instanciaLoc.replace("\t",",")
70            instanciaLoc = list(map(int,instanciaLoc.split(",")))
71            listaLocalizaciones.append(instanciaLoc)
72            listaLocalizaciones[veces].append(0)
73            pass
74          pass
75        pass
76
77    def seleccionarRj(elem):
78      return elem[1]
79
80    def calculoTij(contenedor,locT):
81      fase1=0
82      fase2=0
83      fase3=0
84      fase4=0
85      nuevaPosGrua = 0
86      if contenedor[0] == 1:
87        fase1 = abs(posGrua[2]-posIO_Mar[2]) + abs(posGrua[2]-posIO_Mar[2])
88        pass
89      else:
90        fase1 = abs(posGrua[1]-posIO_Tierra[1]) + abs(posGrua[2]-posIO_Tierra
      [2]) + abs(posGrua[2]-posIO_Tierra[2])
91
92      if contenedor[0] == 1:
93        fase2 = max(abs(posGrua[0]-locT[0]),abs(posGrua[1]-locT[1]))
94        pass
95
96      else:
97        nuevaPosGrua = [5,43,5]
98        fase2 = max(abs(nuevaPosGrua[0]-locT[0]),abs(nuevaPosGrua[1]-locT[1])
      )
99
100     fase3 = abs(posGrua[2]-locT[2]) + abs(posGrua[2]-locT[2])
101
102     fase4 = max(abs(locT[0]-posIO_Mar[0]),abs(locT[1]-posIO_Mar[1]))
103
104     return fase1 + fase2 + fase3 + fase4
105     pass
106
107   tiempo_inicial = time()
108
```

```
  listaContenedores.sort(key=seleccionarRj)
  listaContOrdenada = listaContenedores

  #Para cada contenedor de la lista ordenada FIFO:

  for contenedor in listaContOrdenada:
    if tTotal_Localizacion < contenedor[1]:
      tTotal_Localizacion = contenedor[1]-tTotal_Localizacion
      pass
    locAsignada = 0
    localizacionAleatoria = random.choice(listaLocalizaciones)
    while localizacionAleatoria[3] != 0:

      localizacionAleatoria = random.choice(listaLocalizaciones)
    else:
      tTotal_Localizacion = tTotal_Localizacion + calculoTij(contenedor,
    localizacionAleatoria)
      solucionAlgoritmo.append([contenedor,localizacionAleatoria,
    tTotal_Localizacion])
      localizacionAleatoria[3] = 1
      locAsignada = 1
    pass
  tiempo_final = time()
  tiempo_total = float(tiempo_final) - float(tiempo_inicial)
  pass


tFIN = time()
t_X_iteraciones = tFIN - tINI

archivoEscritura = open("comparacion_tiempos_Pruebas.txt", "a")
archivoEscritura.write(str(tTotal_Localizacion) + "\t" + str(
    t_X_iteraciones))
archivoEscritura.write("\n")

archivoEscritura.close()
```

Listing A.1: Algoritmo aleatorio de localización

```
import  csv
import random
from time import time

def elimina_comas(listaTrimeada):
    nueva = []
    for dato in listaTrimeada:
        if dato != ",":
            nueva.append(int(dato))
    return nueva


tINI = time()

for i in range(1):
  numCont = 0
  free_loc_param = 0
  numLocTotales = 0
  posIO_Tierra = []
  posIO_Mar = []
  listaContenedores = []
  listaLocalizaciones = []

  solucionAlgoritmo = []

  tTotal_Localizacion = 0
```

```
27
28  with open('./instancia_ejemplo.txt') as datos:
29
30    for linea in datos:
31
32      if linea.startswith("number of containers"):
33
34        instanciaNumCont = next(datos).strip()
35        instanciaNumCont = instanciaNumCont.replace("\t",",")
36        numCont = instanciaNumCont
37
38      if linea.startswith("Free Locations Parameter"):
39
40        instanciaFL = next(datos).strip()
41        instanciaFL = instanciaFL.replace("\t",",")
42        free_loc_param = instanciaFL
43
44      if linea.startswith("Crane"):
45
46        instanciaPosGrua = next(datos).strip()
47        instanciaPosGrua = instanciaPosGrua.replace("\t",",")
48        instanciaPosGrua = list(instanciaPosGrua)
49        instanciaPosGrua = elimina_comas(instanciaPosGrua)
50        posGrua = instanciaPosGrua
51
52        pass
53
54      if linea.startswith("I/O Land"):
55        instanciaIO_tierra = next(datos)
56        instanciaIO_tierra = next(datos).strip()
57        instanciaIO_tierra = instanciaIO_tierra.replace("\t",",")
58        instanciaIO_tierra = list(map(int,instanciaIO_tierra.split(",")))
59        posIO_Tierra = instanciaIO_tierra
60
61        pass
62
63
64      if linea.startswith("I/O Sea"):
65        instanciaIO_mar = next(datos)
66        instanciaIO_mar = next(datos).strip()
67        instanciaIO_mar = instanciaIO_mar.replace("\t",",")
68        instanciaIO_mar = list(map(int,instanciaIO_mar.split(",")))
69        posIO_Mar = instanciaIO_mar
70
71        pass
72
73      if linea.startswith("#Type"):
74        for veces in range(int(numCont)):
75          instanciaCont = next(datos).strip()
76          instanciaCont = instanciaCont.replace("\t",",")
77          instanciaCont = list(map(int,instanciaCont.split(",")))
78          listaContenedores.append(instanciaCont)
79          pass
80
81        pass
82
83
84      if linea.startswith("Free Locations:"):
85        instanciaLoc = next(datos)
86        numLocTotales=int(numCont)*int(free_loc_param)
87        for veces in range(numLocTotales):
88          instanciaLoc = next(datos).strip()
89          instanciaLoc = instanciaLoc.replace("\t",",")
90          instanciaLoc = list(map(int,instanciaLoc.split(",")))
```

```
91            listaLocalizaciones.append(instanciaLoc)
92            listaLocalizaciones[veces].append(0)
93            pass
94
95          pass
96
97        pass
98
99    Max_IT = 1
100
101   def seleccionarRj(elem):
102     return elem[1]
103
104   def calculoTij(contenedor,locT):
105     fase1=0
106     fase2=0
107     fase3=0
108     fase4=0
109     nuevaPosGrua = 0
110     if contenedor[0] == 1:
111       fase1 = abs(posGrua[2]-posIO_Mar[2]) + abs(posGrua[2]-posIO_Mar[2])
112       pass
113     else:
114       fase1 = abs(posGrua[1]-posIO_Tierra[1]) + abs(posGrua[2]-posIO_Tierra
      [2]) + abs(posGrua[2]-posIO_Tierra[2])
115
116     if contenedor[0] == 1:
117       fase2 = max(abs(posGrua[0]-locT[0]),abs(posGrua[1]-locT[1]))
118       pass
119
120     else:
121       nuevaPosGrua = [5,43,5]
122       fase2 = max(abs(nuevaPosGrua[0]-locT[0]),abs(nuevaPosGrua[1]-locT[1])
      )
123
124     fase3 = abs(posGrua[2]-locT[2]) + abs(posGrua[2]-locT[2])
125
126     fase4 = max(abs(locT[0]-posIO_Mar[0]),abs(locT[1]-posIO_Mar[1]))
127
128     return fase1 + fase2 + fase3 + fase4
129     pass
130
131
132
133   tiempo_inicial = time()
134
135
136   listaContenedores.sort(key=seleccionarRj)
137   listaContOrdenada = listaContenedores
138
139   for contenedor in listaContOrdenada:
140     if tTotal_Localizacion < contenedor[1]:
141       tTotal_Localizacion = contenedor[1]-tTotal_Localizacion
142       pass
143     listaLocalizaciones = listaLocalizaciones
144     locAsignada = 0
145     localizacionAleatoria = random.choice(listaLocalizaciones)
146
147     while localizacionAleatoria[3] != 0:
148
149       localizacionAleatoria = random.choice(listaLocalizaciones)
150     else:
151       tTotal_Localizacion = tTotal_Localizacion + calculoTij(contenedor,
      localizacionAleatoria)
```

```
152       solucionAlgoritmo.append([contenedor,localizacionAleatoria,
    tTotal_Localizacion])
153       localizacionAleatoria[3] = 1
154       locAsignada = 1
155
156     pass
157
158   print(solucionAlgoritmo)
159
160   tLocal=0
161   tOptimo=0
162   for k in range(Max_IT):
163     for contenedor in listaContOrdenada:
164       indice = listaContenedores.index(contenedor)
165
166       tOptimo = solucionAlgoritmo[indice][2]
167       print("Esto es el tOptimo del contenedor ", contenedor, ": ", tOptimo
    )
168       localizacionAleatoria = random.choice(listaLocalizaciones)
169       locAsignada = 0
170
171       while localizacionAleatoria[3] != 0:
172
173         localizacionAleatoria = random.choice(listaLocalizaciones)
174       else:
175         tLocal = calculoTij(contenedor,localizacionAleatoria)
176
177         if tLocal <= tOptimo:
178
179           solucionAlgoritmo[indice] = [contenedor,localizacionAleatoria,
    tLocal]
180           print("Solucion del algoritmo actualizada: ", solucionAlgoritmo)
181           localizacionAleatoria[3] = 1
182           pass
183         locAsignada = 1
184         pass
185       pass
186
187   tiempoTotal_iterativo = 0
188   for lista in solucionAlgoritmo:
189     tiempoTotal_iterativo = tiempoTotal_iterativo + solucionAlgoritmo[
    solucionAlgoritmo.index(lista)][2]
190     pass
191
192
193   #print(posGrua[:])
194   #print(posIO_Tierra)
195   #print(listaContOrdenada)
196   #print("Tiempo de llegada: ", int(listaContenedores[3][1]))
197   #print(calculoTij(listaContOrdenada[0],listaLocalizaciones[0]))
198   #print("\nLocalizacion aleatoria: ", random.choice(listaLocalizaciones),
    "\n")
199   tiempo_final = time()
200   tiempo_total = float(tiempo_final) - float(tiempo_inicial)
201
202   #print("\nEl tiempo de ejecucion del algoritmo ha sido de ",
    tiempo_inicial)
203   #print("\nEl tiempo de ejecucion del algoritmo ha sido de ", tiempo_final
    )
204   #print("\nEl tiempo de ejecucion del algoritmo ha sido de ", tiempo_total
    )
205   pass
206
207 tFIN = time()
```

```
t_X_iteraciones = tFIN - tINI


archivoEscritura = open("comparacion_tiempos_Prueba.txt", "a")

archivoEscritura.write(str(tiempoTotal_iterativo) + "\t" + str(
    t_X_iteraciones))
archivoEscritura.write("\n")
archivoEscritura.close()
```

Listing A.2: Algoritmo de Escalada Simple

```
import  csv
import random
from time import time


tINI = time()

for i in range(1):
  numCont = 0
  free_loc_param = 0
  numLocTotales = 0
  posIO_Tierra = []
  posIO_Mar = []
  listaContenedores = []
  listaLocalizaciones = []

  solucionAlgoritmo = []

  tTotal = 0
  calculo_tiempos = []
  locMIN = 0


  def elimina_comas(listaTrimeada):
      nueva = []
      for dato in listaTrimeada:
          if dato != ",":
              nueva.append(int(dato))
      return nueva


  with open('C:/Users/Vicente/Desktop/TFG/ADE/programacion/definitivos/
    instancia del EJEMPLO.txt') as datos:

    for linea in datos:

      if linea.startswith("number of containers"):

        instanciaNumCont = next(datos).strip()
        instanciaNumCont = instanciaNumCont.replace("\t",",")
        numCont = instanciaNumCont

      if linea.startswith("Free Locations Parameter"):

        instanciaFL = next(datos).strip()
        instanciaFL = instanciaFL.replace("\t",",")
        free_loc_param = instanciaFL

      if linea.startswith("Crane"):

        instanciaPosGrua = next(datos).strip()
        instanciaPosGrua = instanciaPosGrua.replace("\t",",")
        instanciaPosGrua = list(instanciaPosGrua)
```

```
53          instanciaPosGrua = elimina_comas(instanciaPosGrua)
54          posGrua = instanciaPosGrua
55
56          pass
57
58        if linea.startswith("I/O Land"):
59          instanciaIO_tierra = next(datos)
60          instanciaIO_tierra = next(datos).strip()
61          instanciaIO_tierra = instanciaIO_tierra.replace("\t",",")
62          instanciaIO_tierra = list(map(int,instanciaIO_tierra.split(",")))
63          posIO_Tierra = instanciaIO_tierra
64
65          pass
66
67
68        if linea.startswith("I/O Sea"):
69          instanciaIO_mar = next(datos)
70          instanciaIO_mar = next(datos).strip()
71          instanciaIO_mar = instanciaIO_mar.replace("\t",",")
72          instanciaIO_mar = list(map(int,instanciaIO_mar.split(",")))
73          posIO_Mar = instanciaIO_mar
74
75          pass
76
77        if linea.startswith("#Type"):
78          for veces in range(int(numCont)):
79            instanciaCont = next(datos).strip()
80            instanciaCont = instanciaCont.replace("\t",",")
81            instanciaCont = list(map(int,instanciaCont.split(",")))
82            listaContenedores.append(instanciaCont)
83            pass
84
85          pass
86
87
88        if linea.startswith("Free Locations:"):
89          instanciaLoc = next(datos)
90          numLocTotales=int(numCont)*int(free_loc_param)
91          for veces in range(numLocTotales):
92            instanciaLoc = next(datos).strip()
93            instanciaLoc = instanciaLoc.replace("\t",",")
94            instanciaLoc = list(map(int,instanciaLoc.split(",")))
95            listaLocalizaciones.append(instanciaLoc)
96            listaLocalizaciones[veces].append(0)
97            pass
98
99          pass
100
101       pass
102
103   def seleccionarRj(elem):
104     return elem[1]
105
106   def calculoTij(contenedor,locT):
107     fase1=0
108     fase2=0
109     fase3=0
110     fase4=0
111     nuevaPosGrua = 0
112     if contenedor[0] == 1:
113       fase1 = abs(posGrua[2]-posIO_Mar[2]) + abs(posGrua[2]-posIO_Mar[2])
114       pass
115     else:
```

```
116       fase1 = abs(posGrua[1]-posIO_Tierra[1]) + abs(posGrua[2]-posIO_Tierra
      [2]) + abs(posGrua[2]-posIO_Tierra[2])
117
118     if contenedor[0] == 1:
119       fase2 = max(abs(posGrua[0]-locT[0]),abs(posGrua[1]-locT[1]))
120       pass
121
122     else:
123       nuevaPosGrua = [5,43,5]
124       fase2 = max(abs(nuevaPosGrua[0]-locT[0]),abs(nuevaPosGrua[1]-locT[1])
      )
125
126     fase3 = abs(posGrua[2]-locT[2]) + abs(posGrua[2]-locT[2])
127
128     fase4 = max(abs(locT[0]-posIO_Mar[0]),abs(locT[1]-posIO_Mar[1]))
129
130     return fase1 + fase2 + fase3 + fase4
131     pass
132
133   tiempo_inicial = time()
134
135   listaContenedores.sort(key=seleccionarRj)
136   listaContOrdenada = listaContenedores
137
138
139   print(listaContenedores)
140   print(listaContOrdenada)
141   print(listaLocalizaciones)
142
143   for contenedor in listaContOrdenada:
144     if tTotal < contenedor[1]:
145       tTotal = contenedor[1]-tTotal
146       pass
147     lista_AUX = []
148     for loc in listaLocalizaciones:
149       indice = listaLocalizaciones.index(loc)
150       lista_AUX.append(calculoTij(contenedor,loc))
151       pass
152
153     locMIN = listaLocalizaciones.pop(lista_AUX.index(min(lista_AUX)))
154     tTotal = tTotal + min(lista_AUX)
155     solucionAlgoritmo.append([contenedor,locMIN, tTotal])
156     pass
157
158   tiempo_final = time()
159   tiempo_total = float(tiempo_final) - float(tiempo_inicial)
160
161   pass
162 tFIN = time()
163 t_X_iteraciones = tFIN - tINI
164 print("El tiempo de ejecucion de X repeticiones del algoritmo MINIMO es: ",
       t_X_iteraciones, "segundos.")
165 print("\n\nLa solucion de Escalada Simple corresponde a:\n",
      solucionAlgoritmo, )
166 print("\nEl coste temporal del algoritmo MININO es de: ", tTotal)
167
168 archivoEscritura = open("comparacion_tiempos_Pruebas.txt", "a")
169
170 archivoEscritura.write(str(tTotal) + "\t" + str(t_X_iteraciones))
171 archivoEscritura.write("\n")
172 archivoEscritura.close()
```

Listing A.3: Algoritmo VV-Min