Universidade de Aveiro
2022

Daniel Francisco Domingos Gonçalves

Sistema de informação inteligente para otimização dos processos logísticos: caso de estudo PRIMUS

Universidade de Aveiro
2022

Daniel Francisco Domingos Gonçalves

Sistema de informação inteligente para otimização dos processos logísticos: caso de estudo PRIMUS

Projecto apresentado à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestrado em Engenharia Mecânica, realizada sob orientação científica de José Paulo Oliveira Santos, Professor Auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro, e de Ana Maria Pinto de Moura, Professora Auxiliar do Departamento de Economia, Gestão, Engenharia Industrial e Turismo (DEGEIT) da Universidade de Aveiro.

Este projeto teve o apoio dos projetos UIDB/00481/2020 e UIDP/00481/2020 - Fundação para a Ciência e a Tecnologia; e CENTRO-01-0145 FEDER-022083 - Programa Operacional Regional do Centro (Centro2020), através do Portugal 2020 e do Fundo Europeu de Desenvolvimento Regional.

O júri / The jury

Presidente / President

Prof. Doutor Jorge Augusto Fernandes Ferreira
Professor Associado da Universidade de Aveiro

Vogais / Committee

Prof. Doutor Telmo Miguel Pires Pinto
Professor Auxiliar do Departamento de Engenharia Mecânica da Universidade de Coimbra

Prof. Doutor Ana Maria Pinto de Moura
Professora Auxiliar do Departamento de Economia, Gestão, Engenharia Industrial e Turismo (DEGEIT) da Universidade de Aveiro (co-orientador)

Agradecimentos / Acknowledgements

Em primeiro lugar quero agradecer à minha avó, Casimira, por sempre ter olhado por mim, pelo apoio e afeto que me proporciona desde os tempos de infância. Os momentos são passageiros mas a memória é para sempre. Agradeço especialmente à minha mãe, Maria, e ao meu pai, Diogo, por terem acreditado em mim, e por terem apoiado toda a minha formação académica, apesar de todos os sacrifícios que foram por vezes necessários, e para além disso, por me terem transmitido carinho e também a experiência de vida. Agradeço à Rita pela amizade, apoio, carinho e incentivo. Dou um inestimável valor por teres estado presente tanto nos momentos bons, como nos menos bons. Todo esse apoio ensinou-me a nunca desistir das coisas que mais gosto. Agradeço ao António, que foi um importante pilar na minha formação académica e pessoal, desde o companheirismo até ao incentivo para nunca desistir, obrigado. Agradeço ao Pedro, que sempre me ensinou a ver o lado positivo de todos os cenários e sempre me motivou a nível profissional. Um agradecimento especial a todos os meus amigos de Santo Estevão, que desde do dia um, até aos dias hoje, me deram apoio, alegria e motivação. Sem dúvida uma segunda família que sempre acompanhou o meu percurso pessoal. Para além disso quero agradecer a todas as pessoas que fizeram parte do meu trajeto académico, incluindo professores e amigos de curso. E agradecer a todos os amigos e família que de alguma maneira contribuiram para a pessoa que sou hoje.

Palavras-chave

Toma de Decisão; Logística; Otimização; Python; Algoritmo; Base de Dados.

Resumo

A gestão logística é a parte da Cadeia de Abastecimento que planeia, implementa e controla de forma eficiente e eficaz a movimentação dos fluxos. As atividades logísticas podem fazer parte duma logística externa que contempla os transportes de mercadorias (transportes de entrada e saída), a gestão da frota, desenho da rede logística, entre outros. Por outro lado existem atividades de logística interna tais como a gestão de armazém, a gestão de materiais, a gestão da resposta a encomendas, planeamento do abastecimento e da procura e gestão dos prestadores de serviços logísticos. Nos dias de hoje, os administradores de empresas são desafiados a lidar com problemas tais como a criação de novos produtos estimulados pela rápida mudança dos requisitos de clientes específicos, pelo que é importante ter em sua posse informação relevante para que seja possível traçar metas de vendas de acordo com a produção, gestão da matéria-prima, monitorização do chão de fábrica, monitorização e gestão de stock, gestão de lucros de cada setor específico da empresa, entre muitos outros. Devido ao crescente volume de dados e informação, existe uma quase obrigatoriedade de se utilizarem Sistemas Inteligentes para gerir e transformar os dados em informação útil para a empresa. A competitividade empresarial faz com que as empresas necessitem de informação para sobreviver. Na indústria da cerâmica como em qualquer indústria uma gestão logística ineficiente pode-se refletir em prejuízos relevantes para a organização. Desta forma no âmbito da Indústria 4.0, o trabalho realizado teve como objetivo o desenvolvimento de um Sistema Inteligente constituído por um conjunto de algoritmos baseados em modelos matemáticos e heurísticas, desenvolvido em *Python*, com o propósito de otimizar a atividade de *picking* para a preparação de encomendas, no armazém da Primus Vitória. Numa fase inicial, o *picking* de produtos era realizado pelos operadores que seguem uma Ordem de Trabalho (OT) emitida manualmente pela responsável logística. Nessa OT constam as informações que o operador necessita para realizar a atividade, no entanto, a ordem pela qual os itens são recolhidos são critério do operador. Este problema foi o alvo do presente caso de estudo, sendo que o grande desafio, foi criar um algoritmo que permita recolher da base de dados da empresa informação relativa à zona onde os produtos estão armazenados e dessa forma definir a sequência que resulta no caminho mais curto para a preparação de cada encomenda.

Keywords Decision Making; Logistics; Optimization; Python; Algorithm; Data Base.

Abstract

Logistics management is the part of the Supply Chain that plans, implements and efficiently and effectively controls the movement of flows. Logistical activities can be part of external logistics which includes the transport of goods (inbound and outbound transport), fleet management, design of the logistics network, among others. On the other hand there are internal logistics activities such as warehouse management, materials management, order fulfillment management, supply and demand planning, and management of logistics service providers. Today, business managers are challenged to deal with problems such as the creation of new products stimulated by the dynamic requirements of specific customers, so it is important to have relevant information in their possession in order to be able to set sales targets according to production, raw material management, shop floor monitoring, stock monitoring and management, profit management of each specific sector of the company, and many others. Due to the growing volume of data and information, there is an almost compulsory need to use Intelligent Systems to manage and transform data into useful information for the company. Companies need information to survive due to the Business competitiveness. In the ceramic industry, as in any other industry, inefficient logistics management can result in significant losses for the organization. This way, in the scope of Industry 4.0, the work done had as an objective the development of an Intelligent System constituted by a set of algorithms based on mathematical models and heuristics, developed in *Python*, with the purpose of optimizing the *picking* activity for the preparation of orders, in the Primus Vitória's warehouse. Initially, the picking of products was carried out by operators following a Work Order manually issued by the logistics manager. This Work Order contains the information that the operator needs to perform the activity, however, the order in which the items are picked is at the operator's discretion. This problem was the target of this case study, and the great challenge was to create an algorithm that could collect from the company's database information regarding the area where the products are stored and thus define the sequence that results in the shortest path for the preparation of each order.

Índice

Lista de Tabelas

Lista de Figuras

Lista de Abreviaturas, Siglas e Acrónimos

SAD - Sistema de Apoio à Decisão

TSP - Travelling Salesman Problem

STSP - Steiner Travelling Salesman Problem

ILP - Intiger Linear Programming

MILP - Mixed Intiger Linear Programming

BD - Base de Dados

OT - Ordem de Trabalho

SKU - Stock keeping unit

CA - Cluster Analysis

WMS - Warehouse Management System

TI - Taboeira Interior

TE - Taboeira Exterior

IDE - Integrated Development Environment

API - Application Programming Interface

SQL - Structured Query Language

Capítulo 1

Introdução

É importante destacar que a otimização do processo produtivo e da rede logística está interligada com a evolução tecnológica do mundo em que vivemos. A competitividade empresarial é o motor que estimula o desenvolvimento e implementação de novos tipos de tecnologias, tais como os Sistemas de Apoio à Decisão, nos vários setores das empresas. A eficiência e a produtividade geral de qualquer organização reflete-se num aumento de lucros e redução de desperdícios de tempo. No final, o maior problema pelo qual os administradores destas entidades passam, é conseguir, num determinado processo de decisão, escolher a alternativa que mais benefícios trará.

1.1 Contexto e Apresentação da *Primus Vitória*

A Primus Vitoria é uma empresa especializada em revestimento cerâmico que, desde 1969, data em que iniciou a sua produção, até hoje, sofreu grandes mudanças face ao avanço tecnológico dos últimos anos. A empresa produz azulejos, desde o mais tradicional ao mais moderno, produz uma vasta gama de cores e séries contemporâneas. A empresa conta com uma produção anual na ordem dos 5 milhões de m2 de azulejo. Da sua produção, 75% tem como destino o mercado internacional, tendo como clientes de destaque na Europa, os Países Nórdicos, França e Inglaterra. Para além da Europa a empresa exporta para o mercado internacional americano, evidenciando-se encomendas de grande volume de azulejo branco para os Estados Unidos e Canadá. É reconhecida como uma das mais importantes unidades de produção industrial de azulejo tradicional português.

O processo de produção de azulejos da Primus Vitoria é uma produção em série que funciona com duas linhas de produção paralelas. Pode-se dizer, em grosso modo, que a linha de produção é totalmente automatizada, isto é, sem necessidade de intervenção humana no processo produtivo. As únicas interações dos operários com a linha de produção são no que diz respeito a ajustes em parâmetros e funcionamento das máquinas, reabastecimento e trocas de tintas e ainda atuam também no controlo de qualidade, para a verificação de possíveis defeitos que não sejam identificados de forma automática.

O presente trabalho assenta no desenvolvimento de um sistema inteligente de informação. O tema do trabalho surgiu pela necessidade de otimizar o processo logístico do armazém da empresa, pois revela ser um setor com custos e desperdícios significativos. O ponto de partida para a realização do projeto é o armazém exterior de uma das duas

unidades produtivas da empresa em Aveiro, mais concretamente, da unidade de produção de Taboeira.

1.2 Motivação e Relevância do Tema

É bastante interessante a maneira como os Sistemas de Apoio à Decisão podem interferir na eficiência e eficácia de processos logísticos. Este tipo de sistemas necessitam de dados para serem uma mais valia, dados esses presentes no *Data Warehouse* da empresa ou provenientes duma fonte externa. Assim sendo, a transformação de dados em informação útil, promove o desenvolvimento de soluções numa ótica de *Business Intelligence* que leva a um melhor desempenho da empresa no geral. No âmbito do presente Caso de Estudo, o problema sobre o qual se vai incidir é a ineficência do processo de *picking*, que logo à primeira vista apresenta um potencial de otimização relevante. Na atualidade não existe qualquer critério definido para a realização desta atividade. A pessoa responsável pela atividade apenas recebe uma lista com a informação dos produtos que são necessários recolher para preparar a respetiva encomenda, e as rotas que vai fazer são definidas empiricamente, o que faz com que a tarefa esteja normalmente relacionada com a experiência de cada trabalhador. Posto isto, a otimização desta tarefa em concreto em conjunto com uma arrumação eficiente dos produtos em armazém, pode levar a uma redução das distâncias percorridas e de tempo, tempo esse que pode ser utilizado para outras tarefas, revelando um impacto relevante em todo o fluxo logístico. A longo prazo, uma otimização deste género, pode se refletir num melhor cumprimento de prazos e aumentar a satisfação do cliente, que são pontos importantíssimos a ter em conta na maioria dos negócios.

1.3 Objetivos

O objetivo deste trabalho é a criação de um sistema inteligente de apoio à decisão direcionado para a atividade de *picking* em armazém, que é um processo de logística interna fulcral. Inicialmente, efetuaram-se algumas visitas às instalações da Primus Vitória para conhecer todo o processo produtivo e atividades logísticas existentes desde que um produto sai da linha, até ao seu armazenamento ou expedição. Após conhecer um pouco melhor o contexto real da unidade de produção de Taboeira, foi possível identificar o problema que diz respeito ao *order picking* no respetivo armazém. Pretende-se que o sistema a desenvolver apresente como solução as rotas de *picking* otimizadas para a preparação de cada encomenda nesse armazém, e pretende-se também comparar as rotas obtidas pelo sistema desenvolvido com as rotas obtidas por um modelo matemático, que nos dá a solução ótima.

Esta otimização será realizada através do desenvolvimento de um algoritmo específico, que tem por base regras heurísticas para o cálculo de distâncias e definição do caminho mais curto. Esta otimização irá permitir a redução de desperdícios de tempo, tempo esse que pode ser gasto em outras atividades, e também a redução de eventuais custos associados a atrasos na preparação de encomendas para entrega. Na literatura, existem casos de estudo atuais com problemas semelhantes, porém é difícil encontrar uma solução aplicada a este contexto logístico que se enquadre neste problema em particular.

1.4 Organização do documento

A organização deste documento permite ao leitor entender as várias fases de desenvolvimento deste projeto, percorrendo várias definições e conceitos importantes para uma fácil compreensão do que foi realizado. Nos subcapítulos anteriores ao presente foi feita uma introdução ao problema, contextualização e apresentação da empresa com a qual foi feita uma colaboração.

No capítulo 2 é realizada uma revisão da literatura atual onde se apresentam conceitos importantes para uma melhor compreensão do trabalho realizado e também propostas apresentadas por outros autores para casos de estudo inseridos na mesma temática.

No capítulo 3 é apresentado o problema em detalhe, assim como o funcionamento atual de todas as atividades logísticas associadas, nomeadamente, o funcionamento do armazém e a atividade de *picking*. São explicadas todas as particularidades subjacentes a essas atividades, que foram alvo de otimização através do sistema desenvolvido.

No capítulo 4 são apresentadas as soluções propostas. Foram desenvolvidas soluções para duas abordagens ao problema, uma abordagem ao problema real da empresa que contempla o *picking* de paletes completas para a preparação de encomendas e uma outra, que pode ser considerada uma abordagem conceptual onde se realiza o *picking* de produto até preencher uma palete. Esta segunda abordagem ainda que seja conceptual pode-se revelar útil no futuro e para outros trabalhos, isto porque, na maioria dos casos de empresas com uma grande variedade de produto para distribuição, o comum é que o *order picking* seja feito dessa forma. Ainda neste capítulo é explicado o algoritmo desenvolvido e é feita uma análise crítica dos resultados que foram obtidos através do mesmo.

Por fim, no capítulo 5, apresentam-se as conclusões que se podem retirar do trabalho que foi desenvolvido e também as prespetivas de trabalhos a serem realizados no futuro.

Capítulo 2

Revisão do Estado da Arte

A logística é responsável pela gestão de recursos e informações num contexto empresarial. Fornece uma visão organizacional dos vários setores que representam a produção de uma empresa e tem um elevado impacto na produtividade geral.

Os Sistemas de apoio à decisão (SAD) são sistemas desenvolvidos com o objetivo de, face a um problema, apresentar uma ou um conjunto de soluções alternativas. Um SAD quando utilizado num contexto de gestão logística de uma empresa pode revelar-se bastante útil para o gestor responsável pela toma de decisões, apresentando soluções ou diretrizes e em alguns casos o impacto da escolha de cada uma das diferentes alternativas.

2.1 Sistemas de Apoio à Decisão

2.1.1 Características e Vantagens

É vasta a bibliografia existente sobre SAD. Genericamente, um SAD é constituído por um processo de decisão que passa por quatro etapas: um *input* que é o problema, uma análise, a formulação de soluções e o *output* que são os resultados. Há características importantes que estes sistemas devem ter, tais como, uma avaliação correta da situação, uma boa fusão de informação e a criação de alternativas. Para que um SAD possa ser considerado bem sucedido é necessário que seja simples, fácil de controlar e dentro da sua simplicidade deve ser completo na informação relativa a problemas importantes.

Seguem-se algumas vantagens de um SAD [1]:

- Consegue lidar com grandes quantidades de dados;
- Consegue adquirir e processar dados provenientes de diferentes fontes, sejam essas internas ou externas;
- Apresenta flexibilidade para encaixar nas necessidades do gestor responsável pela toma de decisões;
- Realiza análises complexas recorrendo a software;
- Apoia a otimização;
- Realiza análises do tipo *what-if* e *goal-seeking*.

2.1.2 Estrutura, Taxonomia e Processo de Toma de Decisão

Estes sistemas podem ser constituídos pelos subsistemas de Gestão de Dados, Gestão do Modelo, Gestão baseada no Conhecimento e de Interface do Utilizador, e cada um destes subsistemas tem a sua função.

O primeiro é constituído pela base de dados que contém a informação necessária do problema e é gerido por software. O Subsistema de Gestão do Modelo é constituída por uma base de modelos que proporciona uma variedade de modelos ao responsável pela toma de decisão e pode incluir um software específico que coordena o uso dos modelos num SAD. O Subsistema de Gestão baseada no Conhecimento, é o subsistema que suporta todos os outros mas pode também ser independente. É onde se inclui a inteligência artificial para se gerarem as diferentes alternativas ao problema. Por fim, o Subsistema da Interface do Utilizador, permite que este interaja com o sistema para obter a informação que o ajuda a tomar uma decisão face a um determinado problema, tanto no sentido sistema-utilizador como no sentido utilizador-sistema [1].

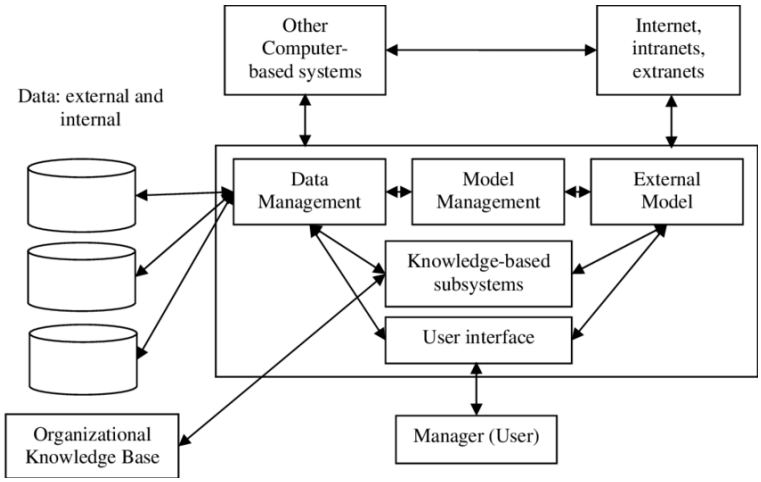


Figura 2.1: Estrutura esquemática comum de um Sistema de Apoio à Decisão (Fonte: Tawfik,2017 [1]).

De acordo com estudos realizados sobre a estrutura destes sistemas, estes podem ser divididos em três grandes constituintes: Bases de Dados e parâmetros de *input*, Ferramentas Analíticas e Mecanismo de Apresentação [9].

O primeiro é formado pela informação base necessária pelo sistema para o processo de análise e apresentação de soluções. Pode ser um conjunto de dados específicos direcionados para um problema específico, um armazenamento de dados acumulados que contêm informações do histórico de atividades de uma empresa ou bases de dados distribuídas cujo o acesso pode ser feito através de um servidor. A análise dos dados exige ferramentas analíticas e conhecimento associado ao problema, e permite que o utilizador afine certos parâmetros. As ferramentas analíticas normalmente aplicadas são algoritmos baseados em inteligência artificial, ferramentas de simulação e cálculo de custo, análise de fluxo e outros procedimentos lógicos. Este constituinte da estrutura pode se considerar o mais complexo pois existem variadíssimos *solvers* em desenvolvimento e pouco conhecidos que podem lidar melhor ou pior com os problemas que, nos dias de hoje, as empresas enfrentam. O Mecanismo de apresentação é a interface final com a qual o utilizador vai

interagir para obter a solução ou alternativa para o problema que pretende otimizar.

Em conformidade com diferentes estudos existem várias taxonomias para estes sistemas, que os classificam com base em diferentes critérios. Estes sistemas podem ser divididos em [9]:

- Passivo - Apoia o processo de toma de decisão mas não consegue produzir uma sugestão de decisão concreta.
- Ativo - Produz propostas de soluções concretas.
- Cooperativo - Permite que o responsável pela toma de decisão interaja com o sistema.

Apresenta-se agora outra forma mais recente de classificação destes sistemas [10]:

- *Data-Driven* - Como o próprio nome indica, este tipo de sistemas têm como requisitos chave o acesso a uma grande quantidade de dados e, simultaneamente, informação de qualidade que possa ser útil.
- *Model-Driven* - Na base deste tipo de sistemas está o acesso a uma vasta quantidade de modelos que permitem, a quem vai tomar a decisão, fazer uma análise das várias situações possíveis. É um tipo de sistema de uso comum em problemas logísticos [11]. Ao contrário do anterior, não necessitam geralmente de um conjunto muito elevado de dados.
- *Document-Driven* - Estes sistemas dão ênfase à integração e análise de documentações tais como políticas e procedimentos, especificações de produto, documentos históricos da instituição, entre outros. Proporcionam espaço para armazenamento e tecnologias de processamento.
- *Communications-Driven* - São geralmente usados em situações de cooperação entre empresas por exemplo, onde existem vários responsáveis pela toma de decisão a colaborar entre si.
- *Knowledge-Driven* - Apresentam-se normalmente implementados em situações que exigem um uso superior de inteligência artifical para resolver o problema. Tal como o tipo de sistemas anterior, têm normalmente como objetivo a toma de decisão estratégica e tática.

A toma de decisão é o processo cognitivo que exige uma posse de informação pertinente e bem organizada, que permita ao responsável escolher, de entre as várias alternativas, aquela que mais vantagens tiver. Para além disso, para que a toma decisão seja eficiente, é necessário perícia, experiência na área respetiva e *feedback* do processo. No mundo empresarial escolher a melhor alternativa é o problema, e uma decisão correta pode representar uma grande redução de custos.

2.2 Gestão de Armazém

Nos dias de hoje sabemos que qualquer organização que faz a produção de um ou vários produtos tem um armazém, e sem dúvida que este faz parte do sistema logístico da maioria das empresas. Os armazéns exigem normalmente grandes investimentos e custos de operação, uma vez que necessitam de espaço e instalações adequadas ao armazenamento e transporte de produto. Se de facto exige um grande investimento e existem bastantes organizações com armazém, é porque estes contribuem para um bom desempenho e organização, que por consequência diminuem custos. Segundo o autor Lambert[12] um armazém ajuda as empresas a estarem um passo mais perto dos seus objetivos pois permite:

- Economizar em custos de transporte e deslocações, por exemplo, colocando produtos com maior volume de encomendas mais acessíveis;
- Poupar na produção uma vez que podem ser adotados, por exemplo, planos de *make-to-stock*;
- Tirar proveito de compras de produto em preço baixo, para o caso das companhias que comprem o produto que vendem;
- A organização de stock e prazos de entrega, fazendo desta forma uma ponte entre a produção e o cliente.

Assim sendo, ter um armazém aparenta ser vantajoso, mas é necessário que o mesmo esteja organizado e sincronizado com os planos de produção de cada firma em específico. Uma arrumação correta dos itens em armazém otimiza também não só o próprio armazém como também outras tarefas logísticas associadas como é o caso do Processo de *Picking*.

2.3 Processo de *Picking*

A tarefa de *Order Picking* engloba o processo de recolher, agrupar e agendar as encomendas do cliente, de maneira a atribuir estrategicamente à encomenda o stock disponível em armazém [4]. Uma encomenda de um cliente consiste num conjunto de várias linhas, cada uma representando um respetivo produto ou *stock keeping unit (SKU)*, numa determinada quantidade. Há diferentes tipos de sistemas de *picking* dependendo do Planeamento Operacional da empresas. O típico é ser contratado um operador para realizar esta tarefa, mas é cada vez mais comum sistemas de *picking* autónomo tratarem do assunto. Entre os vários sistemas de picking estão ***Picker-to-parts***, ***Parts-to-picker***, ***Put system*** e ***Picking* Autónomo** [13].

Na estratégia *Picker-to-parts*, o operador desloca-se ao longo dos corredores (utilizando um veículo específico ou a pé) para recolher os produtos que constituem o pedido, e neste caso, o tempo que o operador demora a procurar os produtos e a distância que percorre pelo armazém são as variáveis críticas. Dentro deste sistema existem dois níveis: *picking* de alto nível, quando o operador responsável pela tarefa vai a bordo de um equipamento que para de forma automática nas localizações indicadas para recolher os itens, e o picking de baixo nível quando o operador percorre os corredores para recolher os itens nas quantidades necessárias [3]. Pode também ser realizado nomeadamente, o *picking*

de paletes completas para situações em que o cliente encomenda um grande volume de produto em cada pedido, e o *picking* de produtos para casos de menores dimensões e maior variedade de produto, onde o operador recolhe itens solicitados na encomenda até completar a palete. Num sistema *Parts-to-picker*, em vez de um operador, são os produtos que se movem, normalmente através de sistemas autónomos de movimentação. A pessoa responsável recolhe a quantidade de cada artigo que é necessária em conformidade com o pedido do cliente, e depois, a carga é novamente armazenada. Este sistema reduz custos de recolha relacionados com o espaço utilizado e tempo de operação, no entanto, existe um aumento do risco de criação de pontos de estrangulamento na alimentação dos postos de recolha [14]. O *Put system* baseia-se numa estratégia diferente das anteriores, neste sistema existe uma primeira etapa de recolha onde se recolhem produtos de vários pedidos, e uma segunda etapa de preparação e finalização das encomendas. O *Picking* Autónomo utiliza *robots* ou distribuidores automáticos para fazer a recolha de produtos e preparação da encomenda. Este apresenta-se mais eficaz quando existe um volume elevado de produtos para recolha. A vantagem destes sistemas são as capacidades que os mesmos têm que vão para além dos limites do ser humano, como a produtividade, velocidade com que realizam a operação e precisão, por outro lado exigem elevados custos de implementação e manutenção por ser uma tecnologia avançada.

O objetivo mais comum dos sistemas de *picking* é de maximizar o nível de serviço. Esse nível de serviço é afetado por vários fatores como a média de tempo de entrega de encomendas, a sua integridade e precisão. A ligação entre o processo de *picking* e o nível de serviço é evidente uma vez que quanto mais rápido uma encomenda for preparada mais cedo estará disponível para envio [15]. Se uma encomenda não for enviada devido a um atraso na sua preparação, pode ter que esperar até ao próximo período de transporte para o cliente, o que naturalmente, diminui o nível e qualidade de serviço. Para além disso, segundo Bartholdi [16], tempo em viagem é desperdício, utiliza horas de trabalho e não adiciona valor ao produto final. É, deste modo, um dos principais processos que se pode e deve melhorar. Na figura 2.2, é possível observar como é utilizado o tempo na tarefa de *picking* num típico sistema *Picker-to-parts*. A informação que se pode retirar é que, de entre os vários passos envolvidos neste processo, aquele que ocupa mais tempo é de facto o tempo em viagem.

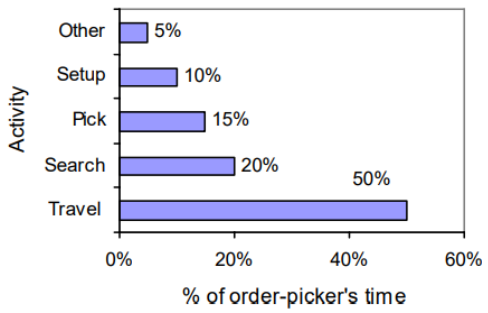


Figura 2.2: Distribuição comum do tempo pelas etapas que constituem o processo de *picking* (Fonte: Riedel, 2011 [2]).

Quando o *picking* é realizado por um operador, o tempo em viagem é diretamente proporcional à distância percorrida, desta forma é comum considerar a distância percorrida o principal foco de otimização num armazém.

Na figura 2.3 apresentam-se de uma forma esquematizada as várias características que o processo de *picking* pode ter.

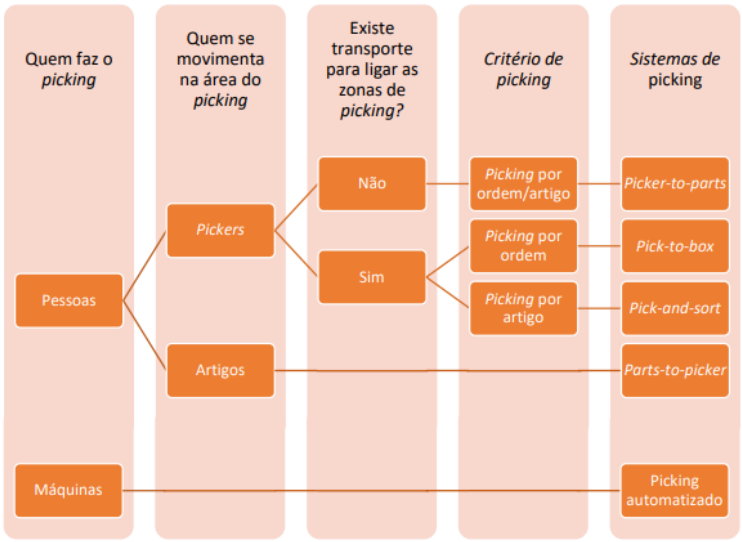


Figura 2.3: Esquema dos vários tipos de sistemas de *picking* (Fonte: Raquel, 2018 [3]).

2.4 Otimização de Problemas Logísticos

A missão da Logística é colocar as mercadorias ou serviços certos no lugar indicado, no instante correto e na condição desejada, ao menor custo possível. De facto, se em alguma circunstância fosse viável todos os bens e serviços serem produzidos no mesmo local onde são consumidos, deixaria de existir a necessidade de gestão logística. Poupar tempo e distância na movimentação de bens ou na entrega de serviços de forma eficaz e eficiente é do principais papéis da Logística. Desta forma, surge a necessidade de se definirem caminhos otimizados para este tipo de operações logísticas, desde o caminho que um operador percorre para fazer o *picking* no interior de um armazém até ao caminho realizado por um ou vários meios de transporte que circulam por vários países até conseguirem finalizar a entrega de uma encomenda ao respetivo cliente.

2.4.1 Problemas de planeamento de rotas

A natureza competitiva do mercado tem vindo a crescer ao longo dos anos, e é por esse motivo que cada vez mais empresas se preocupam em tentar eliminar, dentro dos possíveis, todas as fontes de desperdício. Foi através desta prespetiva que foram surgindo vários problemas de índole logística tais como os problemas de rotas de recolha de encomendas para distribuição.

Um dos problemas usados neste contexto é o **Problema do Caixeiro Viajante** ou ***Travelling Salesman Problem (TSP)***. O nome deste problema derivou da situação

em que um vendedor ambulante inicia o seu trajeto a partir da sua cidade, tem de visitar um determinado número de cidades exatamente uma vez e voltar à sua cidade inicial. As distâncias entre cada par de cidades são conhecidas e o objetivo é obter a ordem, pela qual o vendedor tem de percorrer as cidades, que resulte no trajeto mais curto [4]. Claramente o TSP tem bastantes semelhanças relativamente ao processo de *picking* realizado por um operador em armazém pois geralmente, o operador tem que iniciar o seu trajeto por um ponto de partida, usualmente chamado *depot*, e visitar vários pontos de *picking*. Evidentemente que existem extensões do TSP para problemas mais concretos e com certas particularidades, mas a verdade é que o TSP serve de base para a maioria dos problemas deste género.

O problema mais associado ao processo de *order picking* é uma adaptação do TSP, e é conhecido como *Steiner Travelling Salesman Problem (STSP)*. Esta adaptação surge porque na realização do *picking* em armazém, o operador não tem necessariamente de percorrer todos os pontos do armazém e pode passar por cada ponto mais do que uma vez. O STSP apresenta uma maior complexidade e geralmente não é possível resolvê-lo em tempo polinomial [4].

2.4.2 Heurística e Algoritmos

O planeamento de rotas, em contexto fabril, é normalmente resolvido recorrendo a algoritmos ou heurísticas. Estas apresentam-se vantajosas pois um algoritmo que devolva uma solução ótima pode ser difícil de obter, demorado e é normalmente concreto a cada situação em específico. Algumas das heurísticas de planeamento de rotas são[5]: ***S-shape***, ***Largest Gap***, ***Aisle-by-Aisle***, ***Return Method***, ***Midpoint Method***, ***Combined heuristic*** e ***Optimal heuristic***.

S-shape é uma das heurísticas mais simples e utilizadas para problemas de roteamento, basicamente consiste em percorrer inteiramente todos os corredores que tenham pelo menos uma recolha, e voltar ao ponto inicial, não percorrendo corredores onde não se recolhe nenhum item.

Largest Gap é também um método conhecido, e nesta heurística o operador vai até ao local mais longe do ponto de partida, quando estiver a percorrer o corredor lateral (perpendicular aos corredores de *picking*) faz a recolha de produtos que estejam estejam mais próximos. Quando estiver a fazer o caminho de retorno recolhe os produtos que faltam.

A heurística ***Aisle-by-Aisle*** é em parte semelhante ao *S-Shape*. Os corredores são todos percorridos uma e uma só vez, em cada corredor são recolhidos todos os itens exigidos e o operador sai pelo cruzamento que for mais favorável depois de recolher o último item do corredor.

Para uma melhor vizualização das várias heurísticas de roteamento em armazém, apresenta-se na figura 2.4, um exemplo do seu funcionamento.

Em problemas de roteamento, a solução que se deseja obter é um conjunto de pontos (ou nós) que representa o melhor trajeto em armazém, isto é, o trajeto mais curto. Por detrás dessa solução está todo um processamento que pode ser realizado, por exemplo, por um algoritmo. É frequente que, para problemas semelhantes ou derivados do TSP,

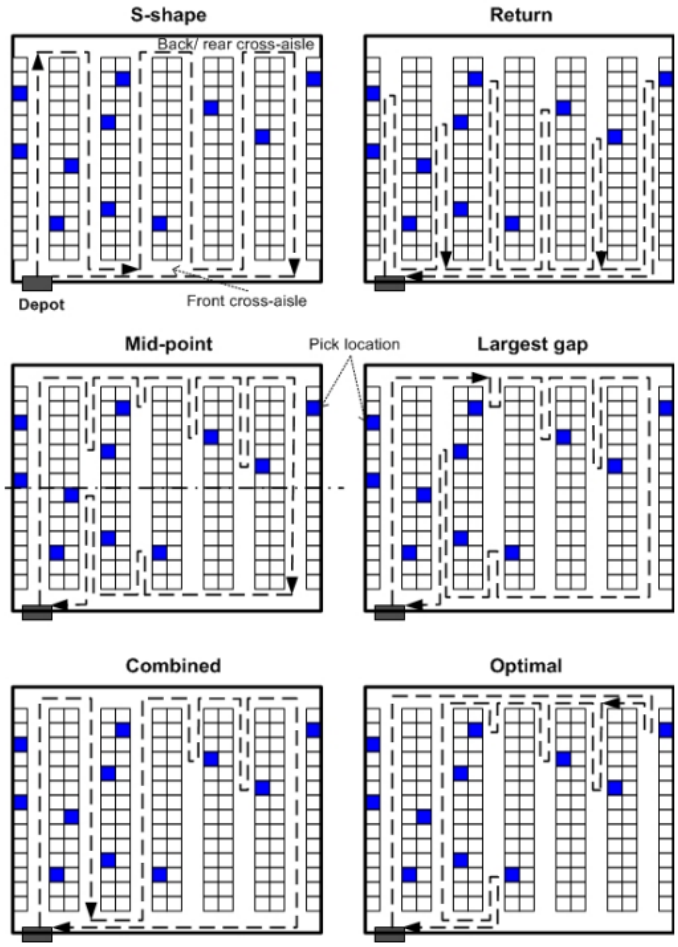


Figura 2.4: Exemplo das principais heurísticas de roteamento em armazém (Fonte:De Koster, 2007 [4]).

se utilizem algoritmos para se tentar chegar à melhor solução. Posto isto, existe um conjunto de algoritmos bem conhecidos nesta temática, que resolvem ou servem de estrutura para resolver bastantes problemas de otimização de rotas. Segundo Beker[17] os mais populares são: ***Algoritmo A****, ***Algoritmo Dijkstra***, ***Floyd-Warshall*** e ***Algoritmo Bellman-Ford***.

O ***Algoritmo Dijkstra*** procura encontrar o caminho mais curto global entre dois nós de um grafo e a função custo do algoritmo é, na sua forma mais simples, baseada nas distâncias entre nós [18]. O ***Algoritmo A**** é outro algoritmo amplamente usado, mas ao invés de realizar uma procura global, utiliza uma heurística. À semelhança de *Dijkstra* este algoritmo calcula o custo dos caminhos percorridos, enquanto a heurística direciona a procura para o objetivo [19]. O peso computacional do $A*$ é menor do que *Dijkstra*, o que o torna mais rápido. No entanto para ambos os algoritmos a solução obtida não é necessáriamente a solução ótima [18]. O Algoritmo de ***Floyd-Warshall*** é ,digamos que, o precedente de Dijkstra. È um algoritmo para encontrar o caminho mais curto entre todos os pares de vértices num grafo com distâncias entre nós. Tem em conta

todas as rotas possíveis para que haja alguma rota a ser apresentada, enquanto verifica cada nó que é percorrido para selecionar a rota mais curta (ponto ótimo local) de modo a que o tempo necessário na procura seja menor. O ***Algoritmo Bellman-Ford*** encontra o caminho mais curto desde um nó inicial até todos os outros nós. Ao contrário do que acontece com os outros algoritmos, este pode ser usado quando há pesos de arestas negativos [5].

2.5 Trabalhos Realizados na área

É importante compreender o panorama geral de desenvolvimento dos SAD atual e compreender também as técnicas e metodologias de gestão logística contemporâneas.

Assim sendo, é apresentada de seguida a revisão duma fração dos trabalhos existentes. Nestes trabalhos são utilizados um SAD flexível para um problema de logística externa, e duas abordagens que incidem sobre a parametrização de critérios na atividade de picking. Nesses trabalhos, estão englobados métodos, modelos e ferramentas utilizadas, e no final da análise de cada trabalho, é realizada uma breve conclusão acerca dos resultados obtidos pelos autores.

2.5.1 Proposta de um Algoritmo para melhorar as Rotas de Picking

Na abordagem a um problema de ineficiência das Rotas de Picking em armazém na Luís Simões Logística Integrada o autor Santos [5] inicia com uma análise do estado inicial do problema, e procura perceber o que realmente acontece no chão de fábrica. A empresa tem serviços de armazenagem de produtos, preparação de encomendas e distribuição. Neste âmbito existem várias exigências impostas pelos clientes que limitam a disposição dos artigos na zona de Picking. O armazém trabalha com PLD(Produtos da categoria drogaria), PLP (Produtos da categoria perfumaria), PLA(Produtos da categoria "arbora") e PLG(Produtos da categoria Gillette) e um dos requisitos é por exemplo, o facto de produtos PLA não poderem estar em contacto com produtos PLD.

Inicialmente a falta de produtividade no contexto industrial levava a atrasos em entregas a clientes. O problema identificado pelo autor remete-nos para o conhecido *Travelling Salesman Problem (TSP)*. A este problema acrescenta-se a escassez de dados históricos desta atividade na empresa. Uma vez que não havia acesso aos tempos de Picking anteriores para poder comparar com os resultados obtidos depois da implementação do novo algoritmo, utilizou-se, como métrica de comparação, a distância percorrida por um operador durante a atividade, procedendo-se a criação em *Excel* de uma matriz de distâncias. Um exemplo duma pequena fração dessa matriz pode ser observado na figura 2.5.

Deste modo, na tentativa de reduzir o tempo que é gasto e não adiciona valor ao produto final, o autor propõe uma nova disposição geográfica dos artigos para *Picking* e um novo Algoritmo para calcular a distância mínima a percorrer num trajeto de *Picking* de um operador.

Os algoritmos depois de desenvolvidos foram programados em *Visual Basic for Applications*, e utilizam diretamente o ficheiro *Excel* da Matriz de Distâncias criado pelo autor. Em primeiro lugar elaborou os algoritmos da função de cálculo das distâncias e o algoritmo que determina a distância mínima entre os locais possíveis a visitar. Seguidamente, desenvolveu através dos anteriores, o algoritmo que gera a rota de picking.

	A	NZ	OA	OB	OC	OD	OE	OF	OG	OH	OI
1		AE093	AE095	AE097	AE099	AE101	AE103	AE105	AE107	AE109	AE111
2	DEPOT	49,17	50,11	51,05	51,99	52,93	53,87	54,81	55,75	56,69	57,63
3	AC031	41,77	42,71	43,65	44,59	45,53	46,47	47,41	48,35	49,29	50,23
4	AC033	40,83	41,77	42,71	43,65	44,59	45,53	46,47	47,41	48,35	49,29
5	AC035	39,89	40,83	41,77	42,71	43,65	44,59	45,53	46,47	47,41	48,35
6	AC037	38,95	39,89	40,83	41,77	42,71	43,65	44,59	45,53	46,47	47,41
7	AC039	38,01	38,95	39,89	40,83	41,77	42,71	43,65	44,59	45,53	46,47
8	AC041	37,07	38,01	38,95	39,89	40,83	41,77	42,71	43,65	44,59	45,53
9	AC043	36,13	37,07	38,01	38,95	39,89	40,83	41,77	42,71	43,65	44,59
10	AC045	35,19	36,13	37,07	38,01	38,95	39,89	40,83	41,77	42,71	43,65
11	AC047	34,25	35,19	36,13	37,07	38,01	38,95	39,89	40,83	41,77	42,71
12	AC049	33,31	34,25	35,19	36,13	37,07	38,01	38,95	39,89	40,83	41,77
13	AC051	32,37	33,31	34,25	35,19	36,13	37,07	38,01	38,95	39,89	40,83
14	AC053	31,43	32,37	33,31	34,25	35,19	36,13	37,07	38,01	38,95	39,89
15	AC055	30,49	31,43	32,37	33,31	34,25	35,19	36,13	37,07	38,01	38,95

Figura 2.5: Matriz parcial de distâncias elaborada pelo autor.(Fonte: Santos, 2015 [5])).

Por fim elaborou o algoritmo da aplicação filtrar encomenda, cujo o objetivo é selecionar a categoria e a classificação da caixa/embalagem exigidas. A título de exemplo, e por os algoritmos serem demasiado extensos, apresenta-se de seguida na figura 2.6 apenas o algoritmo de cálculo do caminho mínimo.

```
Algoritmo CaminhoMínimo ()
ordem (i , 1) ← Ordem estabelecida pelo coeficiente N
ordem (i , 2) ←  correspondente às distâncias percorridas em m trajetos

While N < Nemb   'Número de produtos com a mesma
                          classificação de embalagem

        For i = 1 To Nemb
            If ordem (i , 1) = 0 Then
             distancias (i) = calcdist (poscorrente, localpickcatemb (i))
            End If
        End For

        For i = 1 To Nemb
                If minimodist > distancias (i) Then
                posmin = i
                minimodist = distancias (i)
                End If
        End For
        ordem (posmin, 1) ← N
        ordem (posmin, 2) ← minimodist
        poscorrente ← localpickcatemb (posmin)
End While
```

Figura 2.6: Algoritmo de cálculo do caminho mínimo elaborado pelo autor.(Fonte: Santos, 2015 [5])).

Depois de obtida a aplicação baseada nos algoritmos desenvolvidos, procedeu à sua implementação em armazém para a realização de testes e posterior avaliação da solução proposta.

Após a análise dos resultados obtidos da implementação como mostra a figura 2.7, o autor retira as seguintes conclusões:

-Com a aplicação apenas da nova disposição geográfica dos itens, a redução média da distância percorrida era de **49,37%**;

-Com a aplicação da nova disposição e do novo algoritmo, a redução média era de **66,15%**;

-Mesmo com a aplicação de uma margem de erro generosa, no caso mais pessimista, obteve-se uma redução média de **59,66%** já com a aplicação do novo algoritmo e nova disposição geográfica.

		Novo algoritmo e nova localização (sem margem de erro)		Novo algoritmo e nova localização (com margem de erro)			
	Teste do histórico (real)	Algorit. e nova localiz.	Redução em % (algorit. e nova local.)	Com margem de erro (visão pessimista)	Redução em % com margem de erro (pessimista)	Com margem de erro (visão otimista)	Redução em % com margem de erro (otimista)
2º Quinzena dezembro	540.525	170.010	68,55%	204.338	62,20%	135.682	74,90%
1º Quinzena janeiro	634.398	210.016	66,90%	250.275	60,55%	169.758	73,24%
2º Quinzena janeiro	600.220	204.515	65,93%	243.460	59,44%	165.570	72,42%
1º Quinzena fevereiro	592.906	190.525	67,87%	230.050	61,20%	151.000	74,53%
2º Quinzena fevereiro	539.200	186.434	65,42%	217.712	59,62%	155.156	71,22%
1ºQuinzena março	553.947	209.136	62,25%	249.561	54,95%	168.710	69,54%
		Médias	**66,15%**	-	**59,66%**	-	**72,64%**

Figura 2.7: Resultados obtidos pelo autor após a implementação da sua proposta (Fonte: Santos, 2015 [5])).

De facto pode se considerar que o trabalho implementado pelo o autor foi eficaz, uma vez que, o objetivo da empresa seria uma redução dos tempos de encomenda na ordem dos **10%**. Na realidade os valores de redução obtidos com as alterações efetuadas são bastante superiores por isso pode considerar-se que o sistema contemplado foi uma boa solução.

2.5.2 Um Sistema de Apoio à Decisão para o *Design* e Gestão da rede Logística

No trabalho realizado por R.Manzini [6] é desenvolvida uma abordagem *top-down* para o controlo, gestão e *design* eficiente de uma rede logística de produção-distribuição multi-nível. Esta abordagem foi direcionada sobretudo à logística externa e foi posteriormente aplicada a um SAD utilizando o software *LD-LogOptimizer* e o seu *layout* pode ser observado na figura 2.8 com o objetivo de ajudar no processo de toma de decisão nos diferentes patamares do problema. No final o autor apresenta os resultados obtidos por uma análise

do tipo *what-if* em diferentes cenários e configurações.

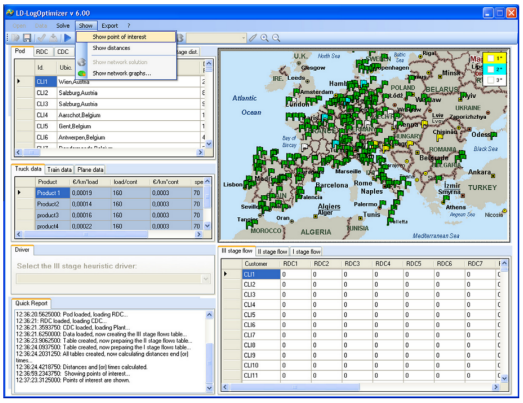


Figura 2.8: *Layout* do *software* LD-LogOptimizer (Fonte: Manzini, 2012 [6])).

Nos problemas mais importantes da Gestão da Cadeia de Abastecimento incluem-se a definição da melhor configuração de rede possível e a identificação dos melhores métodos de gestão e procedimentos operacionais. Neste trabalho é apresentada uma rede de distribuição constituída por três patamares e quatro níveis. Os três patamares são o **Planeamento Estratégico** que lida com decisões de longo prazo como a configuração da rede, o **Planeamento Tático** que lida com decisões a médio prazo e o **Planeamento Operacional** que lida com situações de curto prazo dinâmicas, isto é, dependentes da variável tempo, uma vez que a procura vai variando e consequentemente as atividades associadas podem ser otimizadas. Os quatro níveis são a Unidade de Produção, Centros de Distribuição Centrais, Centros de Distribuição Regionais e o Cliente.

Na literatura é geralmente abordado apenas um problema específico, como por exemplo, a localização das instalações, o *location allocation problem* (LAP) ou o problema da escolha da melhor rota para os veículos de distribuição. A particularidade deste estudo é que, é feita uma análise mais complexa onde se têm em conta vários problemas de índole logística, apesar de não ser possível chegar a uma solução ótima mas sim uma aproximação.

Neste trabalho, para obter soluções eficientes em cada passo da estrutura de planeamento do problema apresentada, são aplicados modelos como por exemplo *Mixed integer linear programming* (MILP) e *Cluster Analysis* (CA) e métodos tais como *solvers* lineares e heurísticas.

Foram desenvolvidas soluções para as três fases de Planeamento. De seguida apresenta-se o desenvolvimento dessas soluções para cada fase em específico.

É importante gerar o **Planeamento Estratégico** como *preset*, antes de se entrar no Planeamento Tático. O Planeamento Estratégico define a configuração da rede (número de instalações ativas e a distribuição da procura nos vários níveis da cadeia). Esta metodologia pode reduzir significativamente o peso computacional da otimização no Planeamento Tático. No âmbito do Planeamento Estratégico o autor propõe o uso de um modelo MILP, e utiliza uma complexa função objetiva da otimização proposta, com inúmeras variáveis associadas aos vários níveis da rede. Este modelo é gerado e resolvido com a ajuda de *A Modeling Language for Mathematical Programming* (AMPL) e de um

solver linear no software *IBM ILOG CPLEX Optimizer*.

O *software LD-LogOptimizer* é flexível e é usado por Gestores Logísticos a operar em empresas de vários escalões e multinível. Utiliza a linguagem C# e pode ser usada em ambientes *Windows*. Podem ser executados separadamente os vários módulos de planeamento ou de acordo com uma abordagem *top-down*. As distâncias no problema de transporte, são normalmente calculadas pela distância euclidiana, mas o software do SAD utilizado dá-nos a possibilidade de calcular essa distância considerando a distância de estrada entre dois pontos.

No caso de o problema se tornar demasiado abrangente e complexo, o utilizador tem a possibilidade de atribuir cada ponto de procura a um centro de distribuição regional disponível, operando assim no terceiro nível da rede logística. Para isso podem ser utilizadas heurísticas como:

- *Max critical custumer convenience (MCCC)*, seguindo uma atribuição baseada no custo;
- *Max critical custumer convenience (MCCC)*, seguindo uma atribuição baseada na distância;
- *Mean facilities through average convenience (MFAC)*, que é uma saturação baseada no custo;
- *Min facilities through average convenience (MFAC)*, que é uma saturação baseada na distância.

Desta forma obtém-se um modelo reduzido constituído por apenas dois níveis, o que ajuda a reduzir bastante a complexidade e peso computacional do problema.

O foco do **Planeamento Tático** é a gestão do cumprimento das entregas entre duas instalações numa rede logística e neste contexto o planeamento é efetuado para períodos de alguns meses onde se usa como unidade temporal o dia. O modelo MILP proposto pelo autor para este tipo de planeamento é dinâmico (dependente do tempo) e portanto complexo, com várias variáveis contínuas e Boleanas.

Dada a complexidade do modelo ótimo, pode ser necessário recorrer a soluções aproximadas e não ótimas (regras heurísticas, por exemplo). Desta forma, recorrendo novamente ao *LD-LogOptimizer* o autor implementa duas possibilidades de resolução para o Planeamento Tático, a ótima e a aproximada (perto do ótimo). Na solução apróximada é realizado um *presetting modelling* do modelo MILP original. Com isto, o peso computacional reduz-se e a solução obtida apresenta-se admissível e muito próxima da solução ótima do modelo dinâmico original.

Assim sendo, para cada ponto no tempo (t) dentro do período de planeamento, para cada produto e para cada patamar da rede logística é possível definir entregas ideais com a especificação das quantidades de produtos, localização do fornecedor genérico e a localização do ponto de procura genérico. Esta é uma calendarização de entregas para o período de planeamento, de acordo com a disponibilidade de diferentes modos e capacidades de transporte, capacidades de produção e armazenamento, em cada ponto no tempo (t).

O **Planeamento Operacional** opera a curto prazo e lida com decisões logísticas relacionadas com as necessidades diárias dos clientes. Neste patamar são usadas estratégias como por exemplo a estratégia de grupo. Esta estratégia consiste em definir os grupos de procura que podem ser visitados por um meio de transporte numa única viagem. Com adoção desta metodologia advém um problema semelhante ao conhecido TSP, onde uma frota fixa de veículos deve responder às necessidades do cliente, a partir de um armazém em comum, com o menor custo de viagem possível.

O autor opta por uma abordagem eficiente de dois passos baseada em regras heurísticas, onde em primeiro lugar se agrupa e depois se traçam as rotas. Na fase de implementação desta abordagem no software *LD-LogOptimizer*, o primeiro passo fundamenta-se em *clustering analysis(CA)* e na adoção de algoritmos heurísticos de agrupamento hierárquico. O agrupamento é feito com base em índices de similaridade que têm uma aplicação alargada em diversas áreas da ciência. A primeira tarefa passa por determinar a matriz de similaridade. O segundo passo é, tendo já os grupos formados, atribuir a rota de menor percurso possível a cada um deles, recorredo novamente a técnicas heurísticas.

O objetivo do caso de estudo apresentado pelo autor é o *design* e configuração duma rede de abastecimento Europeia duma empresa, de três patamares e quatro níveis, constituída por 318 pontos de procura, 15 Centros de Distribuição Regional, 8 Centros de Distribuição Centrais e 5 Centros de Produção. A empresa é responsável pela venda de seis famílias de produtos diferentes, vendidas em 25 países maioritariamente na Europa.

O sistema que foi desenvolvido apresenta várias alternativas baseadas nas diferentes metodologias apresentadas em cima. Os resultados, por exemplo, para o Planeamento Estratégico podem ser observados na figura 2.9, e estes contemplam detelhadamente os custos de cada alternativa. Várias organizações por todo o mundo têm de lidar em simultâneo com problemas relevantes tais como: determinação da melhor localização e melhor número de centros de produção/distribuição, determinação da melhor atribuição de clientes a centros de distribuição, determinação da melhor rota de transporte, entre outros. Um Sistema de Apoio à Decisão deste género apresenta ao responsável pelas decisões logísticas imensa informação relativamente à implementação de cada uma das opções e permite que o mesmo, depois de uma análise cuidada, tome uma decisão informada e acertada. Refletindo sobre a base do sistema e sobre os resultados apresentados, pode considerar-se que a solução proposta foi uma boa solução.

2.5.3 Otimização do Processo de *Picking*

Apresenta-se de seguida, o trabalho realizado por Fernandes,P. [7], cujo o objetivo foi a análise e optimização da atividade de *picking* numa empresa distribuidora de material elétrico. A empresa em questão dedica-se à venda e distribuição desse material no formato de Retalho. O espaço do seu armazém central é de 6000 m^2 e divide-se em quatro zonas: A,B,C e D. O *picking* é feito por 9 operadores que se distribuem pelas várias zonas do armazém conforme se pode ver na figura 2.10.

Neste caso de estudo, tal como na maioria dos outros casos, o *picking* inicia-se com um pedido de encomenda do cliente. Posteriormente, o Sistema de Gestão de Armazém, ou *Warehouse Management System (WMS)*, converte a encomenda numa lista de *picking* apropriada à tarefa a realizar em armazém, tendo em conta os locais onde estão armazenados os produtos. Essa lista contém informações como: Observações; Cliente;

Strategic planning modelling	3S	1S & 2S	MCCC (cost based) & 2S	MCCC (distance based) & 2S	MFAC (cost based) & 2S	MFAC (distance based) & 2S
Full optimization (FULL) vs multi-step (M_Step)	FULL	M_Step	M_Step	M_Step	M_Step	M_Step
Rule 3rd stage	–	MILP	MCCC (cost based)	MCCC (distance based)	MFAC (cost based)	MFAC (distance based)
Modelling 1st & 2nd stages	–	MILP	MILP	MILP	MILP	MILP
Active RDC	4	4	5	15	6	4
Active CDC	3	3	3	3	3	3
Active Plants	3	3	3	3	3	3
Total cost (€)	52,323,030	52,417,060	52,827,160	64,316,780	53,519,830	53,189,450
RDC cost (€)	10,642,130	10,726,690	10,798,620	21,681,440	11,189,590	11,409,560
CDC cost (€)	16,112,710	16,112,710	16,112,710	16,112,710	16,112,710	16,112,710
Plant cost (€)	23,082,760	23,082,760	23,082,760	23,082,760	23,082,760	23,082,760
Facility cost (%)	*95.25*	*95.24*	*94.64*	*94.65*	*94.14*	*95.14*
3° stage transportation cost (€)	859,050	751,126	1,000,685	1,590,049	1,245,030	1,109,545
2° stage transportation cost (€)	581,558	786,470	875,077	864,321	941,733	471,039
1° stage transportation cost (€)	1,044,831	957,311	957,311	985,510	948,015	1,003,842
Transportation cost (%)	4.75	4.76	5.36	5.35	5.86	4.86
Average n° of points of demand served by a RDC	80.00	79.88	64.07	21.19	53.69	80.17
Average n° of RDCs that serve a point of demand	1.01	1.00	1.01	1.00	1.01	1.01
Average n° of RDCs served by a CDC	1.50	1.56	1.94	5.44	2.11	1.50
Average n° of CDCs that serve a RDC	1.13	1.17	1.17	1.09	1.06	1.13
Average n° of CDCs served by a plant	1.44	1.56	1.50	1.50	1.28	1.44
Average n° of plants that serve a CDC	1.44	1.56	1.50	1.50	1.28	1.44
Average 3° stage distance (km)	621	543	703	1155	880	805
Average 2° stage distance (km)	603	624	712	665	754	394
Average 1° stage distance (km)	773	766	786	807	718	813
Average time to reach a point of demand (h)	68.92	78.35	86.29	91.04	87.28	71.94
Average time to reach a regional distribution centre (h)	60.05	70.59	76.24	74.54	74.71	60.44
Average time to reach a central distribution centre (h)	34.45	38.78	39.69	40.60	36.68	40.42

Figura 2.9: Resultados obtidos pelo autor para o Planeamento Estratégico (Fonte: Manzini, 2012 [6])).

Zona	Área (m^2)	Localização	Nrº de Estantes	Nrº de SKU	Nrº de operadores
A	1400	Piso 0	26	6452	3
B	2637	Piso 1	64	9845	2
C	1480	Piso 1	22	5245	3
E	483	Piso 2	27	2362	1
Total	6000	4	139	23904	9

Figura 2.10: Estrutura do armazém e distribuição dos operadores (Fonte: Fernandes, 2017 [7]).

Realização da entrega; Código do produto; Descrição detalhada do produto; Quantidades pedidas e Localização do produto. O autor apresenta três propostas de melhoria para o armazém da empresa a questão. Apresenta uma proposta de melhoria para a Rastreabilidade das encomendas, para o *Layout* do Armazém e para as Rotas percorridas pelo *picker*. Como o presente documento está inserido no âmbito da otimização de rotas de picking, a proposta do autor à qual se vai dar mais atenção é precisamente a proposta de melhoria de rotas. Para este caso de estudo, considera-se que a velocidade do *picker* é constante, assumindo que o tempo gasto na tarefa é diretamente proporcional à distância percorrida. O autor começa, numa primeira fase, por recolher uma amostragem de listas de *picking* para análise, e depois, cria um grafo representativo do *layout* do armazém, constituído pelos caminhos possíveis, para se poder determinar as distâncias percorridas pelo *picker*, tendo em conta que o mesmo começa num ponto inicial e termina noutro ponto diferente do inicial. Todos os pares de nós têm uma distância entre eles e cada nó está associado a um *Stock Keeping Unit (SKU)*. Na situação inicial o WMS da empresa emite uma lista de *picking* ordenando de forma decrescente o número de corredores. O que foi proposto foi primeiro uma reorganização do armazém e posteriormente uma ordenação da recolha de produtos com base no critério do "vizinho mais próximo". O autor denomina a lista de *picking* da situação inicial por LPi, a lista de *picking* depois de reorganização do armazém por LPi* e a lista de *picking* com a reorganização do armazém e com o novo critério de definição da rota por LPi**. Na figuras 2.11 e 2.12 podemos ver

os resultados da LPi** em relação à situação inicial.

	Zona A				Zona E			
	LPA (m)	LPA** (m)	m	%	LPE (m)	LPE** (m)	m	%
LP1	152,3	83,6	-68,7	-45%	68,3	52,5	-15,8	-23%
LP2	154,1	80,8	-73,3	-48%	69,8	41,9	-27,9	-40%
LP3	43,3	13,5	-29,8	-69%	97,5	51,6	-45,9	-47%
LP4	202,9	90,3	-112,6	-55%	93,5	39,5	-54	-58%
LP5	261	114,1	-146,9	-56%	69,9	56,5	-13,4	-19%
Total	**813,6**	**382,3**	**-431,3**	**-53%**	**399**	**242**	**-157**	**-39%**

Figura 2.11: Resultados das distâncias percorridas da Situação Inicial *vs* Solução Proposta LPi** (Fonte: Fernandes, 2017 [7]).

	Zona A				Zona E			
	LPi	LPi**	Diferença	%	LPi	LPi**	Diferença	%
Distância (m)	893,70	462,40	-431,30	-48%	460,70	303,70	-157	-34%
Tempo (min)	10,43	5,55	-4,53	-43%	5,53	3,38	-2,15	-39%
Eficiência (min)	20,52	11,10	-9,42	-46%	11,06	6,46	-4,6	-42%
Blocos (dia)	23,00	43,00	20,00	87%	43,00	74,00	31	72%

Figura 2.12: Resultados das distâncias, tempo, eficiência e blocos da Situação Inicial *vs* Solução Proposta LPi** (Fonte: Fernandes, 2017 [7]).

Ao olhar para os resultados obtidos pelo autor é possível retirar a informação de que a implementação da proposta de reorganização do armazém e a definição do um novo critério para o *picking* resultaram numa diminuição média de distância percorrida de 53% para a zona A e de 39% para a zona E. Relembrando novamente que o tempo gasto na atividade pode se considerar diretamente proporcional à distância percorrida, e desta forma, é possível observar que o tempo gasto em viagem diminuiu em média 43% para a zona A e 39% para a zona E.

No final de contas, refletindo sobre os resultados obtidos pelo autor, podemos dizer que a proposta se revelou uma boa solução, uma vez que os desperdícios associados a esta tarefa de *picking* diminuiram em percentagens significativas, chegando a alcançar quase os 70% para o caso com uma melhoria mais elevada, e os 19% para o caso de menor sucesso, que por si só, já é um valor bastante significativo, isto porque, por exemplo no trabalho de Santos, D.[5], o objetivo mínimo imposto para a melhoria do processo era de 10%.

Capítulo 3

Apresentação do Problema

Neste capítulo descreve-se em detalhe o problema identificado sobre o qual este trabalho incidiu. Apresenta-se o principal foco da otimização realizada que é o atual funcionamento do processo de *picking* no armazém exterior da empresa na unidade de produção de Taboeira, localizada em Aveiro.

3.1 Armazém e processos associados

Atualmente a referida unidade de produção, tem dois armazéns, um interior e um exterior. O armazém interior é de pequenas dimensões, é constituído apenas por estantes e por uma zona de monos. A metodologia atual que a empresa adota é nessa zona de estantes colocar produtos que têm como destino encomendas pequenas e direcionadas para clientes específicos. A zona de monos não é de grande relevância pois raramente é frequentada pelos operadores para recolha de produto. É constituída sobretudo de produtos que vão ficando para trás ao longo dos anos e muitos deles já nem se quer são produzidos. O armazém exterior, é um recente investimento na unidade de produção em questão, e tem à volta de 5500 m^2 de área. Os armazéns interior e exterior distinguem-se pelo prefixo TI ou TE respetivamente, e, em ambas, existe uma diferenciação numérica dos corredores que é identificada através de marcas no chão. A título de exemplo, TE06, refere-se ao corredor 6 da zona de armazenamento exterior da unidade de produção de Taboeira.

O armazém sobre que foi alvo de estudo para este trabalho foi o armazém exterior que se pode observar na figura 3.1, que é atualmente o local com um maior fluxo logístico, e consequentemente, mais propício a oportunidades de melhoria. Este contém, junto ao corredor com ligação ao armazém interior, uma zona para guardar paletes vazias, e contém, na zona lateral de maior largura um espaço dedicado ao carregamento das encomendas nos veículos que tratam da sua distribuição. Todo o restante espaço é dedicado ao armazenamento de produtos que saiem da linha de produção adjacente. O critério que a organização utiliza para a disposição dos itens em armazém é a rotatividade, isto é, a frequência com que determinado produto é solicitado numa encomenda. Desta maneira, os produtos que tiverem uma maior rotatividade são colocados mais perto da zona de expedição.

Atualmente, os azulejos que acabam de ser produzidos, são manulamente inseridos na base de dados da Primus, onde a responsável logística assume a responsabilidade de

Figura 3.1: Armazém exterior das instalações da Primus Vitória em Taboeira.

colocar o produto no sistema, indicando várias informações sobre o mesmo, tais como, o lote, a quantidade em m^2, a zona onde vai ser armazenado, entre outros. No seguimento da inserção no sistema, são emitidas etiquetas contendo algumas dessas informações para colocar nas paletes, como forma de validação e para que os operadores possam ler o seu código de barras quando estiverem a fazer a recolha de produto. Desta forma, após a leitura do código de barras, a base de dados é atualizada e o stock desse produto é subtraído. Um exemplo destas etiquetas pode ser observado na Figura 3.2.

Figura 3.2: Etiqueta de produto acabado.

Após as etiquetas serem colocadas, as paletes vão então para o respetivo local do armazém exterior onde vão ficar armazenadas, até serem recolhidas para alguma encomenda.

3.2 Funcionamento do *Order Picking*

Antes da atividade de *picking* propriamente dita, existem várias tarefas logísticas. Assim que os produtos saem da linha de produção percorrem um sistema automático que

determina a melhor disposição para os colocar na palete, estes vão sendo empilhados até completar uma palete. Depois, através do *robot* designado para o efeito, prodece-se à colocação de filme, que é uma película de proteção para os produtos que fica envolvida em toda a palete, dessa forma miniza-se o efeito sobre a qualidade do produto, de fatores relacionados com o ambiente industrial e com o transporte. Após a filmagem das paletes estas vão para a zona de receção de *picking*, como se vê na figura 3.3, que é a zona onde ficam provisorimente até serem levadas para o armazém. Em algumas situações é realizada a atividade *cross-docking*, que é quando as paletes chegam à zona de receção de *picking* e vão diretamente para a zona de expedição sem ser necessário o seu armazenamento.

Figura 3.3: Zona de receção de *picking*
da unidade de produção.

Quando surge uma encomenda de um cliente, é emitida uma Ordem de Trabalho, como a que é ilustrada na Figura 3.4, onde são apresentadas todas as informações necessárias à atividade de *picking* a realizar por um determinado operador. Muitas das vezes, uma só encomenda exige *pickings* de vários produtos diferentes que se encontram em zonas distintas do armazém. Atualmente quem realiza essa atividade não segue nenhuma metodologia específica. Este pode ser considerado um sistema de *picking* do tipo *Picker-to-parts* de baixo nível, uma vez que os operadores se deslocam em porta-paletes elétricos percorrendo os corredores para fazer a recolha de itens. O operador inicia a tarefa a partir de um ponto inicial usualmente chamado de *depot* que se localiza junto à zona de receção de *picking*, e após recolher o(s) produto(s) necessário(s) finaliza o seu trajeto no ponto de expedição.

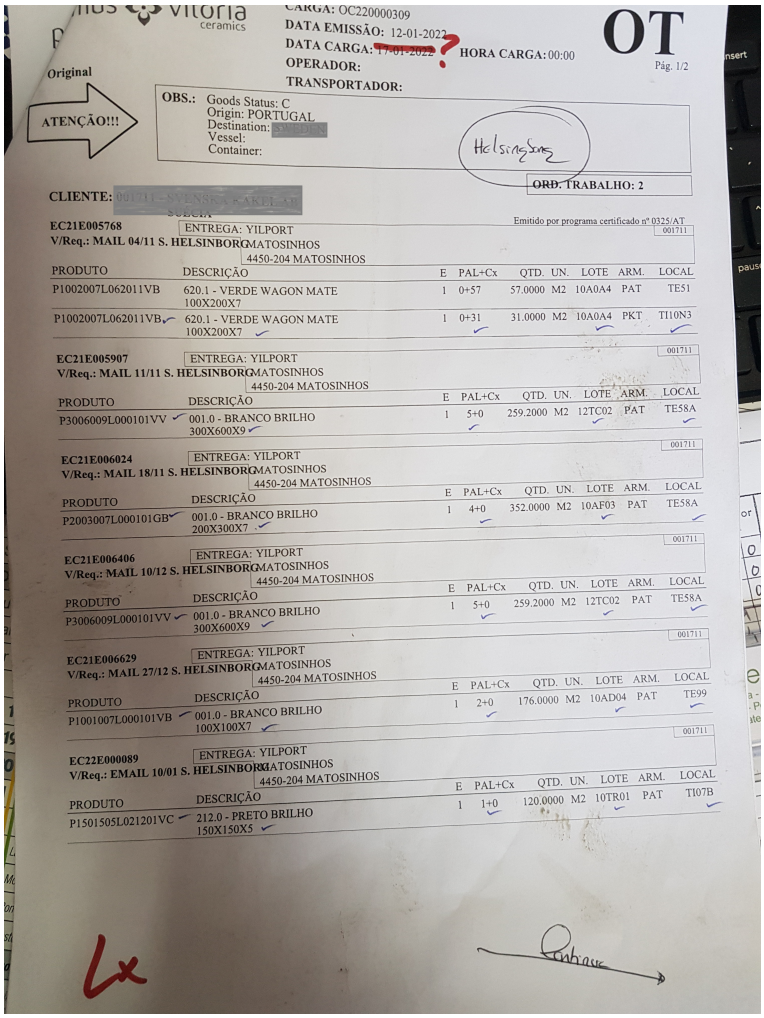


Figura 3.4: Exemplo de Ordem de Trabalho emitida após um pedido de encomenda do cliente.

3.3 Problemas identificados

O operador que recebe a folha com os locais do armazém a percorrer utiliza um critério pessoal para definir a rota que vai realizar. Ainda que na Ordem de Trabalho se apresentem as informações sobre o produto, e a zona onde o mesmo se pode recolher, não é definido qualquer critério que auxilie o operador a decidir qual produto deve ir buscar em primeiro lugar. Assim sendo, pode-se dizer que a eficiência deste processo fica limitada principalmente à experiência do operador que faz o *picking* que, por mais aprimorada que seja, não é suficiente, tendo em conta as dimensões do armazém. Desta maneira identificou-se um dos problemas com potencial de otimização.

Para além disso, na OT, a informação relativa ao local onde estão armazenados os produtos, referencia apenas o corredor e acontece que, num só corredor podem estar armazenados diferentes produtos em ambos os lados, como se pode ver na figura 3.5, e estes corredores chegam a ter mais de 30 metros. Consequentemente pode ser possível por vezes, um operador percorrer todo o corredor para ir buscar um produto que esteja

Figura 3.5: Corredor de *picking* em armazém.

no final do mesmo, enquanto poderia ser mais eficiente recolher o mesmo produto pela entrada oposta do corredor. Este problema identificado podia ser melhorado através de uma reorganização do armazém em conjunto com a atribuição de uma numeração que identificasse pontos de *picking* específicos ao longo dos corredores.

Este cenário e outros semelhantes são ineficiências que somadas podem gerar desperdícios significativos. São essas as situações que se pretendem otimizar através do desenvolvimento de um sistema que auxilie a tomada de decisão para o *picking* de encomenda.

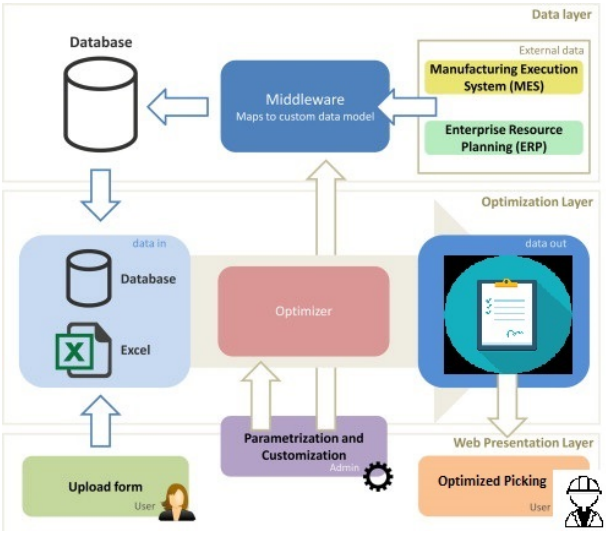


Figura 3.6: Estrutura genérica para um Sistema de Apoio à Decisão direcionado para atividade de *picking* (Adaptado de Figueira, 2015 [8]).

Foi idealizada, como ponto de partida, uma estrutura base semelhante à da figura 3.6, para todo o sistema que se prentedia desenvolver, sendo que é na camada de otimização (*Optimization Layer*) que se insere o algoritmo desenvolvido. O que é apresentado na

figura representa uma estrutura ideal, e uma vez que não foi possível utilizar a infraestrutura de dados real da empresa, no caso do sistema desenvolvido a Camada de Dados (*Data Layer*) é constituída apenas por uma BD no servidor local criada através de registos da atividade da empresa fornecidos pela mesma. O algoritmo para além de devolver uma solução para o trajeto mais curto, contém uma parte de código que manipula essa solução de forma a torná-la *user-friendly* antes de ser apresentada. O local onde se apresenta a solução ao utilizador faz parte duma camada que se pode chamar de Camada de Apresentação (ou *Web Presentation Layer*), que no caso do sistema desenvolvido corresponde a um terminal do *sotfware* IDE (*Integrated Development Environment*) utilizado, mais concretamente, o *Visual Code*, um ambiente de desenvolvimento da *Microsoft*.

Capítulo 4

Solução Proposta

Tendo em conta os problemas identificados no capítulo anterior, procedeu-se com o desenvolvimento de soluções que fossem capazes de dar uma resposta ao problema. O principal objetivo das soluções desenvolvidas é a obtenção de rotas de *picking* otimizadas que de facto sejam uma mais valia para a respetiva tarefa a realizar pelos operadores. Como foi dito, atualmente o responsável pela tarefa não segue nenhum método definido, por isso o que é necessário é a existência de algum critério de suporte para esta tarefa, que é exatamente o que se propõe com a abordagem desenvolvida. De acordo com as várias taxonomias apresentadas na secção 2.1.2 da Análise do Estado da Arte, a solução apresenta certas características que a podem associar a sistemas do tipo *Data-Driven* pois, mesmo que não seja um grande volume, são precisos dados para se obter um resultado. A solução apresenta também características típicas de um sistema *Model-Driven* uma vez que apresenta resultados baseados em modelos que utilizam heurísticas diferentes. No entanto, pode-se afirmar que o tipo de sistema ao qual a solução desenvolvida mais se aproxima é o *Knowledge-Driven* pois para que o algoritmo apresente bons resultados é necessário conhecimento sobre o problema e sobre os métodos e as heurísticas cuja sua utilização se apresenta mais vantajosa. Para além disso é um problema que utiliza inteligência artificial para ser resolvido.

4.1 *Layout* do armazém

Numa fase inicial do desenvolvimento da solução, e uma vez que não foi possível a obtenção de uma planta do armazém, foi realizado presencialmente nas instalações um levantamento da geometria e medidas do armazém. Depois de efetuadas todas as medições foi desenhado, um esquema representativo do armazém exterior. O esquema contém todos os trajetos possíveis a realizar pelo operador durante a tarefa e pode ser observado no anexo A.

Seguidamente, foi realizado um segundo esquema com maior detalhe onde para além dos trajetos possíveis, constam também todas as distâncias (em metros) necessárias ao desenvolvimento dos passos seguintes da solução. Para além disso neste esquema foram adicionados nós ao longo dos trajetos, que numa primeira fase apenas representavam o início e o final de cada corredor. No total, este esquema era constituído por 26 nós. Em conformidade com esta fase inicial do esquema, foi desenvolvida uma matriz de distâncias. Depois, foi feita uma abordagem conceptual onde os corredores têm vários

pontos de *picking* para que fosse possível obter rotas mais eficientes a partir do algoritmo, em casos onde o operador não necessita de percorrer todo o corredor para recolher um determinado produto, e pode voltar para trás. Assim sendo, decidiu-se atribuir a cada corredor 4 pontos de *picking*, todos equidistantes entre si, e no total o esquema ficou com 59 nós. O segundo esquema que foi desenvolvido pode ser vizualizado no anexo B.

Este último esquema desenvolvido serviu de base para o próximo passo do desenvolvimento da solução, a criação de um grafo que representa o armazém e que é um dos *input's* da abordagem.

4.2 Criação do grafo

Para a criação do grafo de suporte do algoritmo desenvolvido foi usado o *Networkx*. O *Networkx* é um *package* de uso livre para a linguagem de programação *Python* utilizado para criar, manipular e estudar as dinâmicas e estrutura de grafos e redes. Para além do *networkx* foram usadas bibliotecas como o *pandas* e *numpy*. A biblioteca do *pandas* foi usada para se poderem ler e extrair valores dum ficheiro excel que contém a matriz de distâncias desenvolvida com base nos esquemas realizados anteriormente. *Numpy* é uma biblioteca *Python* que suporta o processamento de *array's* e matrizes multidimensionais e de grande dimensão, e contém um variado conjunto de funções matemáticas de alto nível para operar sobre essas matrizes. A criação do grafo em questão passou por 2 fases:

- **Criação de nós** - Os nós foram criados através de uma função específica do *networkx*, chamada *add_node*, cujo o *input* é o número de identificação desse nó, assim como o seu posicionamento no espaço. O seu posicionamento no espaço não tem nenhuma relação com as distâncias entre os nós (isto é, com os pesos das arestas) e serve simplesmente para uma melhor vizualização gráfica e para se evitar possíveis erros que pudessem surgir do cruzamento de arestas.

- **Criação de arestas de ligação** - Primeiro foram importadas as distâncias da matriz desenvolvida. Depois recorreu-se à função *add_weighted_edges_from* para criar as arestas com peso associado. Esse peso corresponde ao comprimento das arestas que ligam os vários nós.

No fim obteve-se um grafo que se pode visualizar graficamente na figura 4.1, que é constituído por 59 nós que se encontram com uma identificação numérica, e pelas arestas que os ligam, que têm um peso associado correspondente à distância (em metros) entre nós.

4.3 Abordagens ao problema

No contexto real da empresa, a tarefa de *picking* contempla a recolha de paletes completas, iniciando o trajeto no ponto de partida (de agora em diante chamado *depot*), avançando até à zona onde está a palete e finalizando-o na zona de expedição onde deixa a respetiva palete. Depois disso, o operador repete o processo até ter recolhido todas as paletes necessárias para uma determinada encomenda.

Considerou-se importante que a solução desenvolvida fosse o mais genérica possível para que se podesse enquadrar em vários problemas semelhantes que existem nos dias

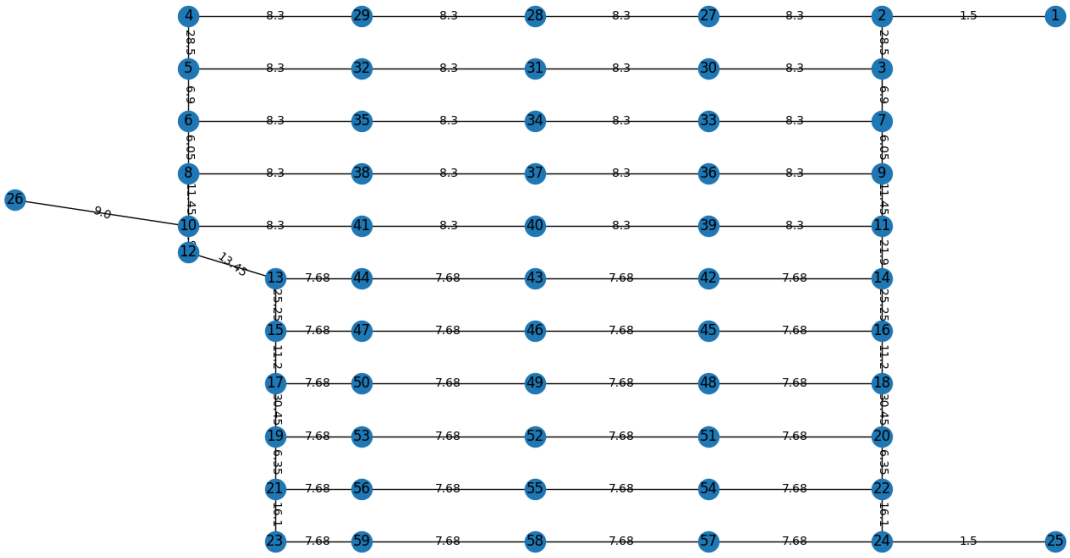


Figura 4.1: Grafo representativo do armazém (não está à escala).

de hoje. Desta forma, decidiu-se que se iriam usar duas abordagens diferentes para o mesmo problema. A verdade é que existem vários sistemas de picking diferentes, mas apesar disso há tipos de *picking* que são usados com mais regularidade que outros. Esse é o caso do *picking* de produto, onde se recolhem ao longo dos armazéns os vários produtos solicitados na encomenda, até completar a encomenda ou até completar a palete.

Então as duas abordagens consideradas para as soluções desenvolvidas foram o *Picking* de palete e o *Picking* de produto. Na primeira abordagem, considera-se que o caminho a efetuar pelo operador é constituído por um conjunto de vários trajetos. Cada trajeto representa a viagem do operador desde o *depot* até um ponto de *picking*, e desse ponto até à zona de expedição. Para a segunda abordagem o caminho a realizar é um trajeto único que se inicia no *depot*, passa por todos os Pontos de *picking* necessários e termina na zona de expedição.

Ainda que a abordagem de *Picking* de Produto seja uma abordagem conceptual para o armazém exterior, esta pode ser utilizada, por exemplo, no armazém interior da empresa, pois ao contrário do que acontece no armazém exterior onde já se encontram armazenadas paletes completas de produtos, no interior os produtos estão colocados em estantes, e desta forma, encomendas de menor dimensão e para clientes específicos, seguem um tipo de *picking* idêntico à abordagem de *Picking* de Produto considerada.

Nas figuras 4.2 e 4.3 encontra-se um esquema simples para ilustrar as duas abordagens. As particularidades de cada abordagem levaram a que se tivessem de desenvolver dois algoritmos diferentes, um para cada abordagem.

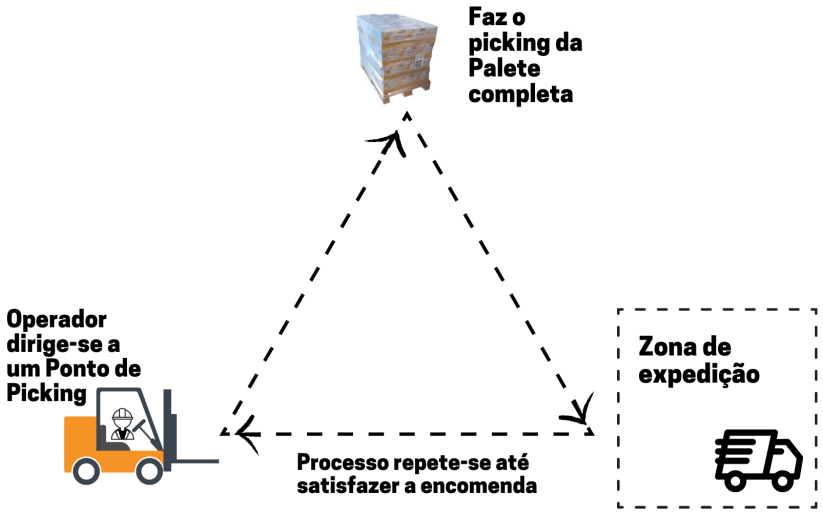


Figura 4.2: Ilustração da abordagem para o *Picking* de paletes completas.

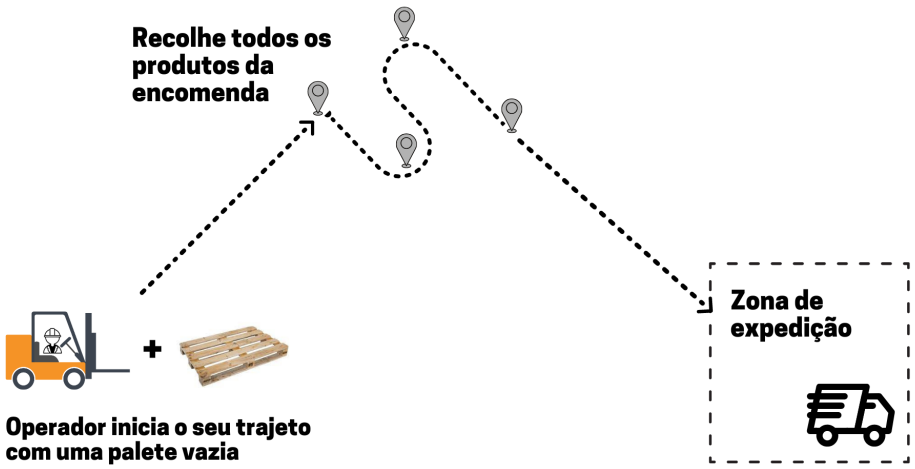


Figura 4.3: Ilustração da abordagem para o *Picking* de produto.

4.4 Abordagem heurística para o planeamento das rotas de *picking*

Mais uma vez, através do *package Networkx* foi possível desenvolver abordagens com as características desejadas para ambos os problemas. Este *package* coloca ao dispor uma panóplia de funções e algoritmos genéricos para vários problemas, inclusive para o Problema do Caixeiro Viajante. A solução desenvolvida teve como base os algoritmos de *Dijkstra* e *A**. No entanto é preciso fazer uma adaptação a estes dois algoritmos, uma vez que ambos calculam o trajeto mais curto entre dois pontos, o inicial e o final, e o pretendido para solucionar o presente problema é, para a abordagem de *Picking* de produto, o cálculo do melhor trajeto entre dois pontos com a particularidade de ser obrigatório passar por determinado número de pontos de picking, e para a abordagem de *Picking* de palete, determinar o melhor conjunto de trajetos, sendo que cada um dos trajetos tem de se iniciar no *depot* e terminar no ponto final passando obrigatoriamente por um ponto de picking.

De forma a obter a solução para os problemas considerados, foi necessário o desenvolvimento de 4 algoritmos. Cada abordagem utiliza 3 dos algoritmos desenvolvidos, pois existem 2 que são comuns a ambas, o algoritmo do Grafo e de ligação à Base de Dados (BD). Quanto a estes dois últimos, um deles é o algoritmo que contém todo o código necessário à construção do grafo, explicada no subcapítulo 4.2. O outro algoritmo contém o código necessário para establecer a ligação e trocar informação com a BD, que vai ser explicado no subcapítulo 4.5. É neste algoritmo que obtemos os pontos de *picking* associados aos nós que são necessários percorrer para recolher todos os produtos que constituem a respetiva encomenda. Os restantes dois, são constituídos pelo código que calcula a melhor rota para cada abordagem.

4.4.1 Abordagem do *picking* de produto

Ao algoritmo que nos devolve como *output* o melhor trajeto e a sua distância total, para esta abordagem, deu-se o nome de *Picking de Produto*. O código do mesmo contém várias operações lógicas que em conjunto nos levam à solução, no entanto, existe uma primeira condição neste algoritmo que o divide em duas partes e cada uma delas utiliza heurísticas diferentes para chegar à solução. Essa é a condição que verifica se o número de pontos de *picking* é menor ou igual a 8, ou se é superior.

Número de pontos de *picking* inferior ou igual a 8

Para o caso do número de pontos de *picking* ser inferior ou igual a 8, são calculadas todas as permutações possíveis entre o ponto 1 (*depot*), todos os pontos de *picking* e o ponto 26 (zona de expedição). Depois de calcular todos os caminhos possíveis, é devolvido aquele que apresenta o menor comprimento, desta forma obtém-se uma solução que se pode considerar ótima. O senão desta lógica é a exigência computacional da rotina, e foi precisamente por causa disso que se decidiu colocar um limite de 8 pontos de *picking* para que esta seja usada, pois para números dentro deste limite o peso computacional é razoável e o programa demora em média apenas 3 segundos a devolver uma solução. Caso se utilizem mais de 8 pontos de *picking* para esta abordagem, o tempo de computação aumentaria exponencialmente. Para calcular o trajeto entre dois pontos e o seu

comprimento, ao longo de todas as abordagens consideradas, foram usadas funções do *package Networkx* que se baseiam no algoritmo de *Dijkstra* e no algoritmo *A**. Foi realizado um fluxograma, ilustrado na figura 4.4, que descreve de uma forma simplificada a lógica utilizada no algoritmo construído. Para uma análise mais detalhada, o ficheiro de código desenvolvido encontra-se no anexo C.

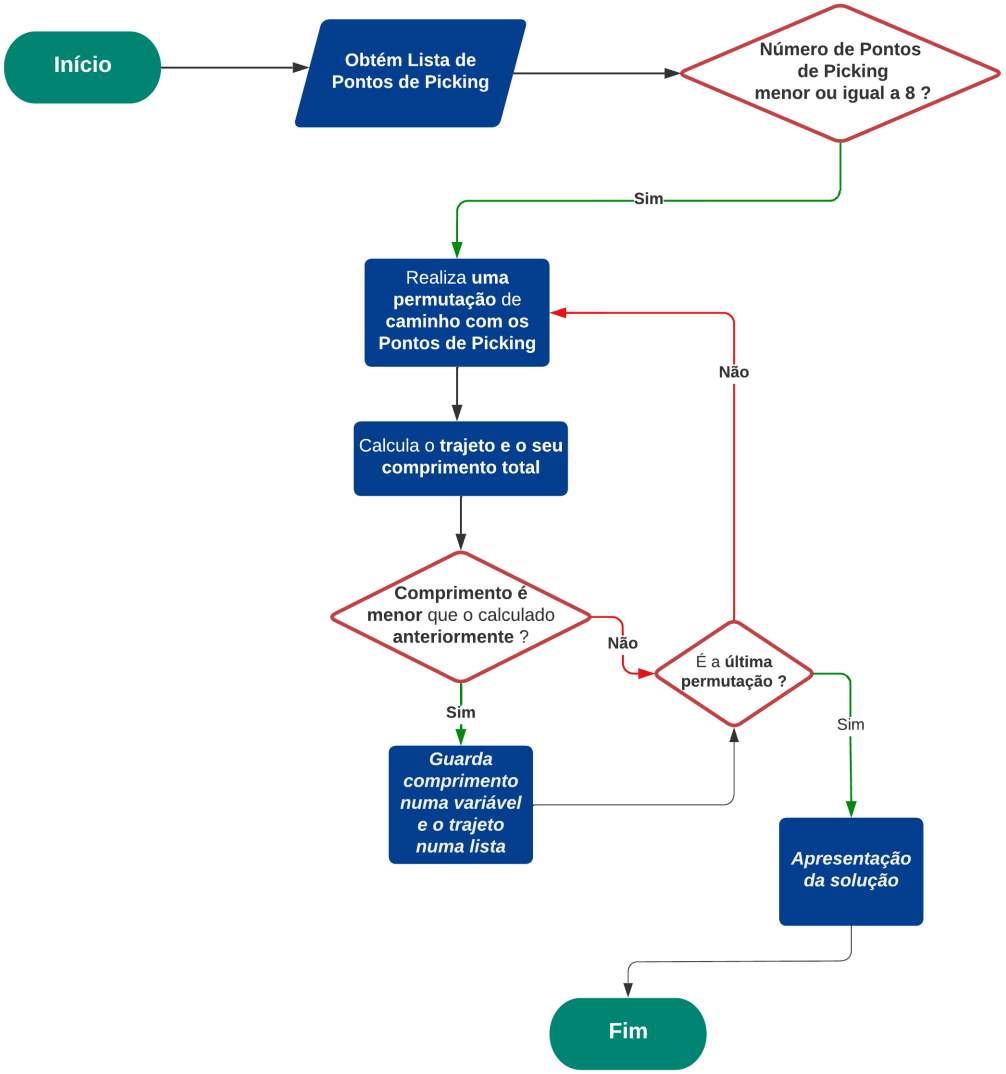


Figura 4.4: Fluxograma desenvolvido para o *Picking* de produto até 8 pontos de *picking*, inclusive.

Número de pontos de *picking* superior a 8

No caso em que o número de pontos de *picking* é maior do que 8, uma vez que o peso computacional se torna uma variável crítica, utiliza-se para resolver o problema, a heurística do vizinho mais próximo. O programa calcula qual é o ponto de *picking* que

se encontra mais próximo do *depot*, avança para esse ponto, e depois calcula o ponto de *picking* seguinte mais próximo, move-se para esse ponto, e assim sucessivamente até chegar ao destino, que é o ponto 26 (zona de expedição). Para calcular cada trajeto individual entre dois pontos, recorreu-se como já foi referido a funções específicas do *Networkx*, que devolvem o caminho mais curto entre dois pontos, e o trajeto final é obtido somando a lista de todos os trajetos individuais. Pode se observar na figura 4.5 um fluxograma que descreve este processo, e encontra-se também no anexo D o excerto de código desenvolvido para esta abordagem.

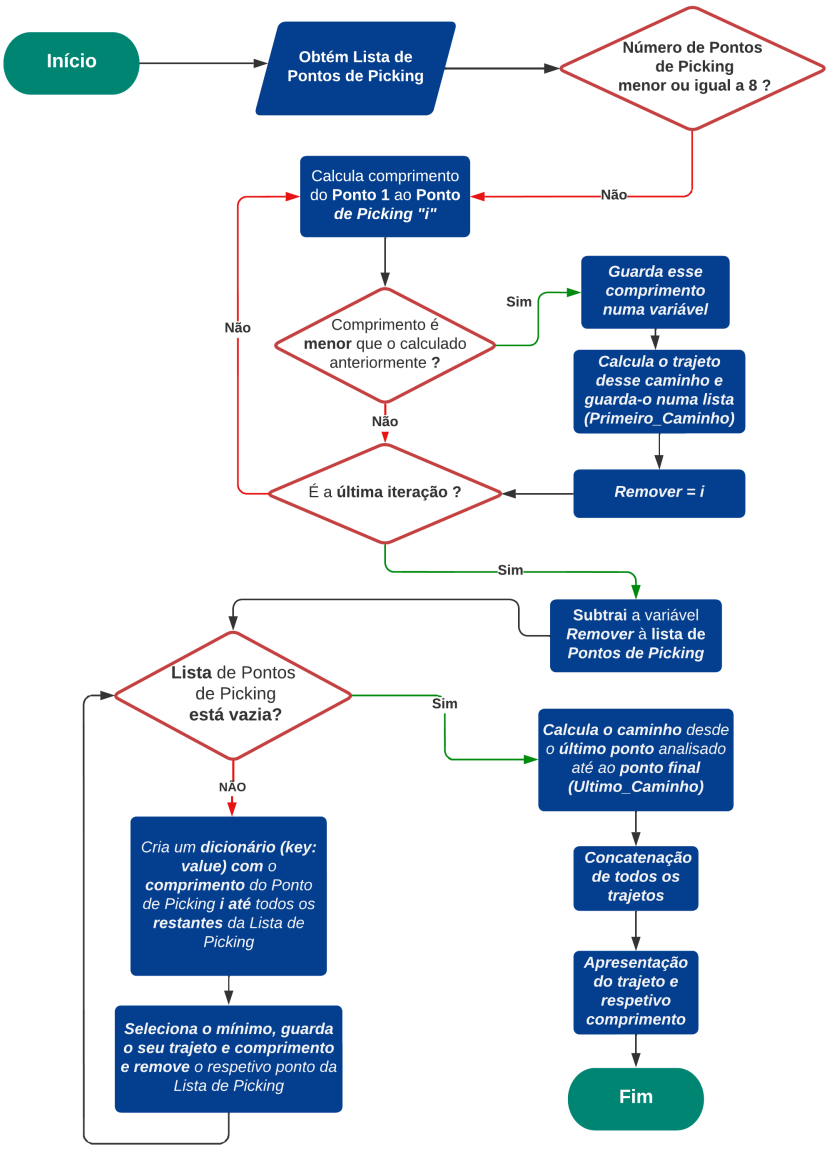


Figura 4.5: Fluxograma desenvolvido para o *Picking* de produto com mais de 8 pontos de *picking*.

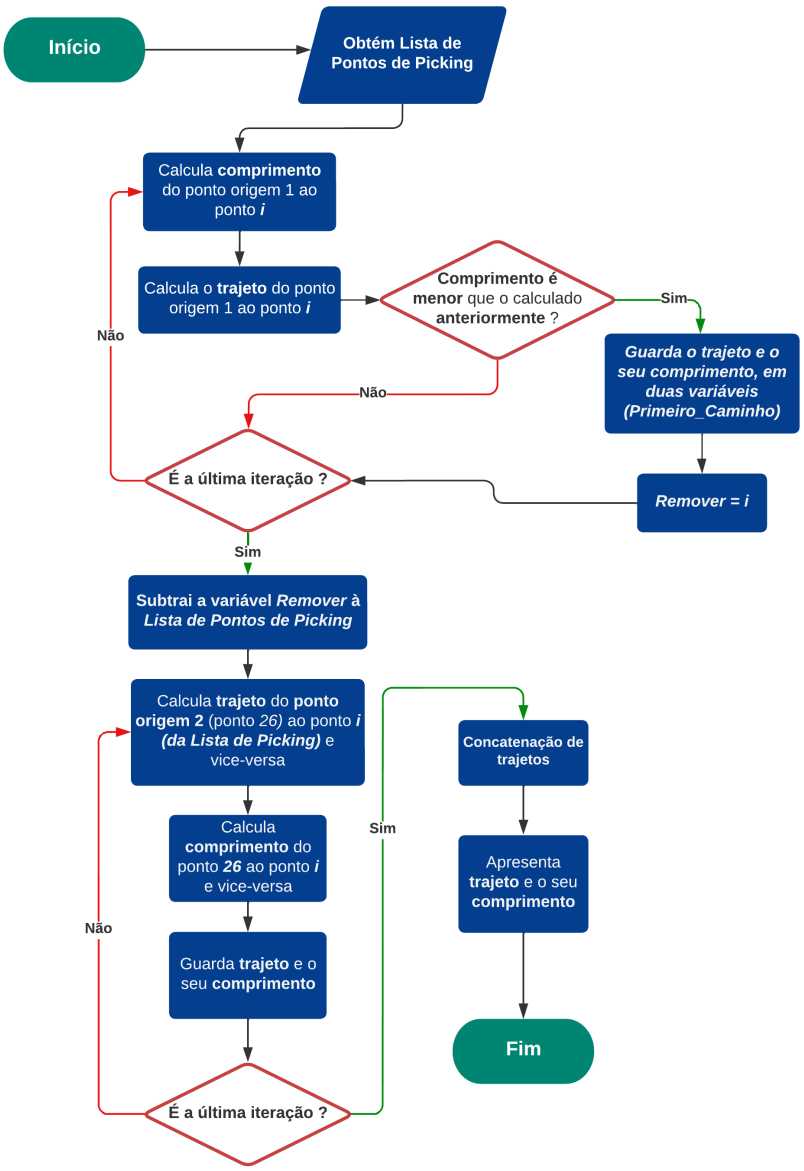


Figura 4.6: Fluxograma do algoritmo de *Picking de Palete*.

4.4.2 Abordagem do *picking* de palete completa

Neste caso em que cada ponto de *picking* representa uma palete completa a recolher, o caminho de todos os pontos de *picking* até ao ponto 26 (zona de expedição), tem que ser feito, e por isso esse mesmo caminho é independente da ordem pela qual a recolha nos pontos de picking é efetuada. Desta forma o caminho que vai influenciar o comprimento final do trajeto é o caminho do *depot*, ao primeiro ponto de *picking*, uma vez que os restantes pontos serão sempre rotas diretas entre o ponto 26 e o ponto de *picking*. Portanto temos que escolher como primeiro ponto de *picking* a percorrer aquele que estiver mais próximo do *depot*. O código do algoritmo desenvolvido para esta abordagem assenta precisamente nesta lógica. Resumidamente, o algoritmo calcula qual o melhor ponto de *picking* para ir em primeiro lugar, definindo o seu trajeto pelos pontos do grafo, e calcula os melhores trajetos de todos os outros pontos de *picking* até ao ponto 26. Agregando todos esses trajetos, o algoritmo devolve o trajeto total assim como o seu comprimento. A lógica utilizada para o desenvolvimento da solução desta abordagem, é ilustrada de uma forma simples na figura 4.6. Para uma análise mais cuidada, é apresentado também no anexo E, o código do *script* desenvolvido, responsável pela obtenção de um trajeto otimizado para esta abordagem.

4.5 Ligação à base de dados

Os pontos de *picking* a usar como *input* no algoritmo desenvolvido, foram extraídos de uma BD. O *software MySQL*, pertence à Oracle Corporation, e é um sistema de gestão de bases de dados que utiliza a linguagem SQL. Dentro deste programa podem-se criar e importar bases de dados, e as mesmas podem ser manipuladas através da execução de *queries*. Para o presente caso de estudo, foi criada uma BD para a realização de testes, no servidor local (*local host*). A empresa com a qual houve cooperação, forneceu um ficheiro *Excel* com registos de um ano de encomendas, contendo informações como: Tipo, Documento, Linha, Produto, Lote, Quantidade, Quantidade da Palete e Data. Uma vez que na atualidade a empresa apenas identifica numericamente os corredores, foi adicionada à tabela uma coluna com o nome Pontos de *Picking*, para se poderem efetuar testes de funcionamento do algoritmo. O número de *picking* atribuído a cada linha é um número aleatório entre 1 e 59, excluindo o número 1 e 26, que são os pontos inicial e final respetivamente. Um excerto do ficheiro *Excel* já com a coluna de Pontos de *Picking* pode ser observado na figura 4.7.

	A	B	C	D	E	F	G	H	I	J
1	Tipo	Documento	Linha	Produto	Lote	Quantidade	QtdPalete	Data	PontosPicking	
2	Carga	OC220002590	5	P1004000H0PINK1VN	10AA01	21,6	60,48	31/03/2022 10:48	19	
3	Carga	OC220002590	2	P1002007B00040EBB	60A0SL	7	96	31/03/2022 10:48	29	
4	Carga	OC220002590	8	P1004000H0BRIG1VN	10AB01	20,16	60,48	31/03/2022 10:48	40	
5	Carga	OC220002590	1	P1002007B000401G5	10AZ04	18	88	31/03/2022 10:48	29	
6	Carga	OC220002590	7	P1004000H0MARI1VN	10AA01	12,24	60,48	31/03/2022 10:48	47	
7	Carga	OC220002590	6	P1002007B063001VB	10AC01	2	88	31/03/2022 10:48	56	
8	Preparacao	OP22R000527	4	P1501505L062501VC	10TC01	1,5	120	31/03/2022 09:17	52	
9	Preparacao	OP22R000527	1	P1501505L045001VC	10TA01	1,5	120	31/03/2022 09:17	33	
10	Preparacao	OP22R000527	3	P1501505L063001VC	10TB01	1,5	120	31/03/2022 09:17	25	
11	Preparacao	OP22R000527	5	P1501505L005101VC	10TB01	1,5	120	31/03/2022 09:17	40	
12	Preparacao	OP22R000527	2	P1501505L064501VC	10TA01	1,5	120	31/03/2022 09:17	14	
13	Carga	OC220002593	2	P2002007L001111VD	10TA01	52,8	102,4	31/03/2022 09:00	39	
14	Carga	OC220002568	3	P200030XL030211GA	00A0SL	30	6720	31/03/2022 09:00	36	
15	Carga	OC220002593	1	P2002007L040211VD	10AA01	52,8	102,4	31/03/2022 09:00	30	
16	Carga	OC220002568	1	P3006009L000101VV	11TC02	6,48	51,84	31/03/2022 09:00	40	
17	Carga	OC220002568	1	P3006009L000101VV	11TC02	6,48	51,84	31/03/2022 09:00	55	
18	Carga	OC220002568	2	P200030VL030211GA	00A0SL	200	6720	31/03/2022 09:00	23	
19	Preparacao	OP22R000526	1	P1501505L019201VC	10TB01	3	120	31/03/2022 08:49	20	
20	Preparacao	OP22R000526	2	P1501505L019201VC	10TB01	13,5	120	31/03/2022 08:49	12	
21	Carga	OC220002547	5	P1501505L061001VC	10TA01	6	120	31/03/2022 08:48	48	
22	Preparacao	OP22R000525	1	P1500757B062001VI	10AH01	1,3635	86,355	31/03/2022 08:48	19	
23	Carga	OC220002547	6	P1501505L062501VC	10TC01	6	120	31/03/2022 08:48	19	
24	Carga	OC220002547	2	P2002007L067501VD	10TB01	3,2	102,4	31/03/2022 08:48	45	

Folha1

Figura 4.7: Registos com informações relativas a encomendas.

De seguida, importou-se este ficheiro *Excel* para o *MySQL Workbench*, criando uma BD contendo toda a informação do ficheiro, como se pode ver na figura 4.8.

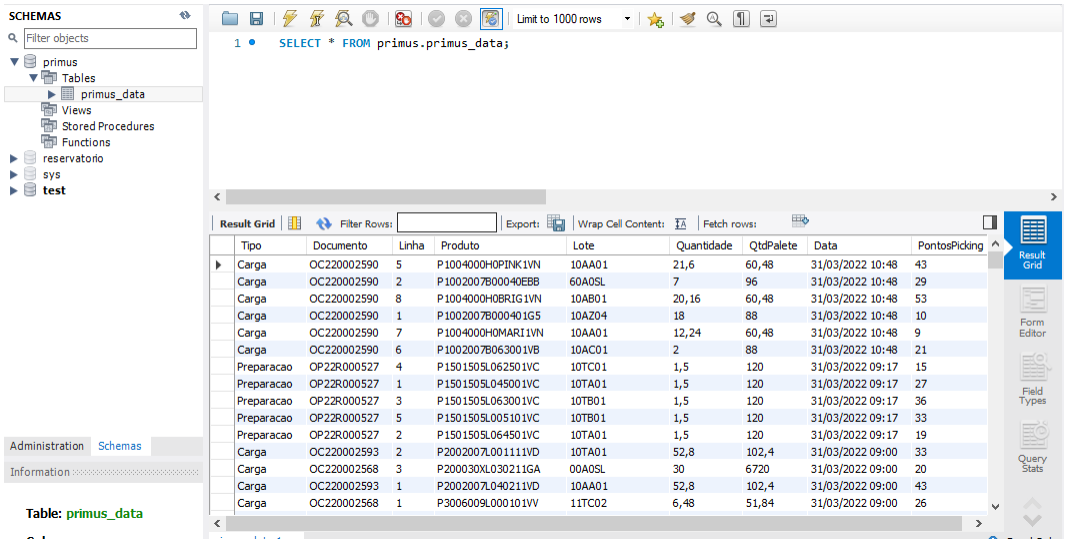


Figura 4.8: Base de Dados constituída pela tabela associada aos resgistos fornecidos.

Para fazer a ligação da BD ao programa em *Python* foi instalada uma biblioteca já existente para o efeito. Esse *package* tem o nome *mysql-connector*, e é tipicamente usado para fazer a ligação *Python* com *MySQL*. O mesmo funciona como uma API, que significa *Application Programming Interface*, e a sua esquemática de funcionamento pode ser observada na figura 4.9. De um modo resumido, esta ligação establece-se primeiramente através de um pedido enviado pelo programa *Python* para a BD. Depois da ligação estar establecida, a biblioteca contém uma função denominada *my cursor* que cria uma variável no programa *Python*, onde se vai introduzir uma *string* que contém *query* SQL que se pretende correr na BD. Ao utilizar no programa a função *Cursor.execute*, o *MySQL* importa essa variável e executa o código contido nela, e posteriormente guarda o resultado

obido na variável *myresult*, e devolve essa variável ao programa *Python*. Esta foi a lógica utilizada para o desenvolvimento do algoritmo de ligação à BD, que se pode vizualizar na figura 4.10.

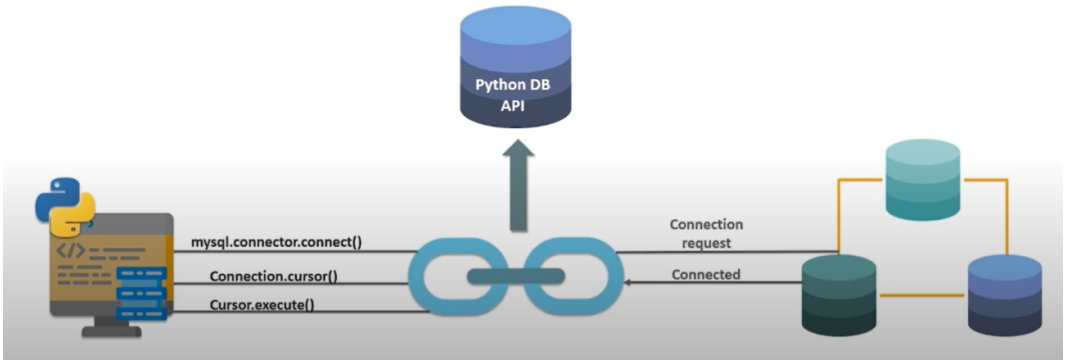


Figura 4.9: Esquemática de funcionamento da biblioteca *python*, *mysql-connector*.

```python
import mysql.connector as mysql
import random

mydb = mysql.connect(host="localhost", user="root", passwd="12345asdfg",database="primus")
print(mydb)

if(mydb):
    print("Connection Successful")
else:
    print("Connection Unsuccessful")

mycursor = mydb.cursor()

mycursor.execute("Select Data from primus_data")
myresult = mycursor.fetchall()
resultado=[]

al = int((random.random()*3998)+1)

for row in myresult: #Row significa linha
    resultado = resultado + list(row)

#print(resultado)
value_to_read = resultado[al]

mycursor.execute("Select PontosPicking from primus.primus_data where Data=\"" + value_to_read + "\"")
myresult = mycursor.fetchall()

Pontos_Picking=[]
for row in myresult:
    Pontos_Picking = Pontos_Picking + list(row)

print(Pontos_Picking)
```

Figura 4.10: Algoritmo de ligação à base de dados desenvolvido.

4.6 Modelo Matemático

Para o problema do presente caso de estudo, além das abordagens desenvolvidas e apresentadas, foi também utilizado um modelo do *Travelling Salesman Problem* (TSP), que é o problema onde um vendedor tem que percorrer várias cidades, iniciando e finalizando

na mesma cidade, realizando o melhor trajeto possível. O TSP foi apresentado com um maior detalhe na secção 2.4.1, da Revisão do Estado da Arte. Existem diferentes formulações para o problema, neste caso a utilizada foi a formulação de *Miller–Tucker–Zemlin* que se encontra descrita nas equações seguintes:

$$\sum_{i \in P} \sum_{j \in P: j \neq i} c_{ij}\ x_{ij} \tag{4.1}$$

s.a. :

$$\sum_{j \in P: j \neq i} x_{ij} = 1, \quad \forall i \in P \tag{4.2}$$

$$\sum_{i \in P: i \neq j} x_{ij} = 1, \quad \forall j \in P \tag{4.3}$$

$$u_i - u_j + P \times x_{ij} \ \leq \ P - 1, \quad \forall\ i, j \in P \quad \text{with} \quad i > 1,\ j > 1,\ i \neq j \tag{4.4}$$

$$x_{ij} \in \{0, 1\}, \quad \forall\ i, j \in P \tag{4.5}$$

$$u_i \geq 0, \quad \forall\ i \in P \tag{4.6}$$

onde os dados de entrada são:

$P = \{1, ..., p\}$	Conjunto de pontos de *picking* e de expedição
$d_{i,j}$	Matriz de distâncias entre todos os pontos

e as variáveis são:

u_i	Variável auxiliar que mantém o registo da ordem pela qual os pontos são visitados
$x_{i,j}$	Variável de decisão binária toma o valor 1 se o percurso for de i para j e toma o valor 0 em caso contrário

O objetivo deste modelo é a minimização da distância total percorrida (Eq. 4.1) para o picking. Este objetivo está sujeito às seguintes restrições: Eq. 4.2 e Eq. 4.3. Estas garantem que em cada ponto de picking só se entra e sai uma única vez. A restrição descrita na Eq. 4.4, permite a prevenção de ciclos, isto é, impõem que haja apenas uma única rota que cubra todos os pontos de picking, e não duas ou mais rotas desarticuladas que cubram apenas coletivamente todos os pontos de picking. As Eq. 4.5 e Eq. 4.6 representam o domínio das variáveis do problema.

Este modelo foi executado recorrendo ao *software IBM ILOG CPLEX Optimization Studio*, com o objetivo de comparar os resultados obtidos a partir do mesmo, com os resultados das abordagens desenvolvidas, como se vai mostrar no capítulo 5. É importante ter em conta que este modelo é do tipo *ILP (Integer Linear Programming)*, onde todas as variáveis são inteiras e todas as condições são lineares e a solução obtida através do mesmo, é uma solução ótima.

Capítulo 5

Apresentação e análise dos resultados

No presente capítulo apresentam-se os resultados obtidos com as soluções desenvolvidas, para as duas abordagens e para o modelo matemático. O objetivo é interpretar uma amostra de 6 testes para o algoritmo de *Picking* de Produto e 3 testes para o algoritmo de *Picking* de Palete. Depois de obter os resultados referidos, realizou-se uma comparação com os resultados obtidos pelo modelo matemático, e uma análise crítica geral dessa comparação. Foram considerados cenários de testes variados para ambas as abordagens para que fosse possível perceber a maneira como os algoritmos se desempenham perante *input's* distintos. Os resultados obtidos foram tratados de forma a que a sua interpretação seja simples. Recorreu-se à utilização de tabelas para a sua apresentação, e foi utilizado o grafo desenvolvido para traçar os trajetos devolvidos pelos algoritmos e pelo modelo.

5.1 Resultados do algoritmo para as duas abordagens

Cada uma das abordagens realizadas tem o seu algoritmo, e cada algoritmo apresenta duas variantes, uma baseada no algoritmo $A*$ e outra baseada no algoritmo de *Dijkstra*. Para o mesmo cenário de testes, essas duas variantes apresentam exatamente os mesmos resultados, pelo que são apresentados apenas os resultados para a variante baseada no algoritmo $A*$.

5.1.1 *Picking* de Produto

Para esta abordagem, foram realizados testes com diferentes conjuntos de pontos de *picking* para que se pudessem obter resultados que pudessem representar os vários cenários possíveis. Assim sendo, nos primeiros 3 testes realizados para este caso, foi considerado um número de pontos de *Picking* inferior ou igual a 8. Os resultados obtidos são os que se encontram na tabela 5.1. É importante ter em conta que para o número máximo de pontos de picking desta condição, o programa demorou cerca de 19 segundos a devolver o resultado. Para os restantes casos o programa demorou menos de 2 segundos a correr pelo que não se considerou um tempo de processamento elevado.

Para além disso, apresenta-se nas figuras 5.1, 5.2 e 5.3, os trajetos sugeridos pelo algoritmo para os três testes. Para cada teste, as setas vermelhas representam o caminho realizado por um operador caso seguisse a solução proposta pelo algoritmo para a realização da tarefa.

Foram realizados mais 3 testes para o caso em que o número de pontos de *Picking* é superior a 8, e os resultados que se obtiveram são apresentados na tabela 5.2. Assim como nos testes anteriores, ilustram-se nas figuras 5.4, 5.5 e 5.6, os trajetos obtidos para os testes 4, 5 e 6 respetivamente.

Tabela 5.1: Resultados obtidos nos 3 testes com algoritmo de *Picking* de Produto, até 8 pontos de *picking*.

Teste nº	Pontos de Picking	Distância total (metros)
1	[25, 29, 47]	371,92
2	[57, 55, 53, 49, 39, 33, 31, 21]	447,2
3	[44, 34, 24, 20, 8, 7, 6]	385,5

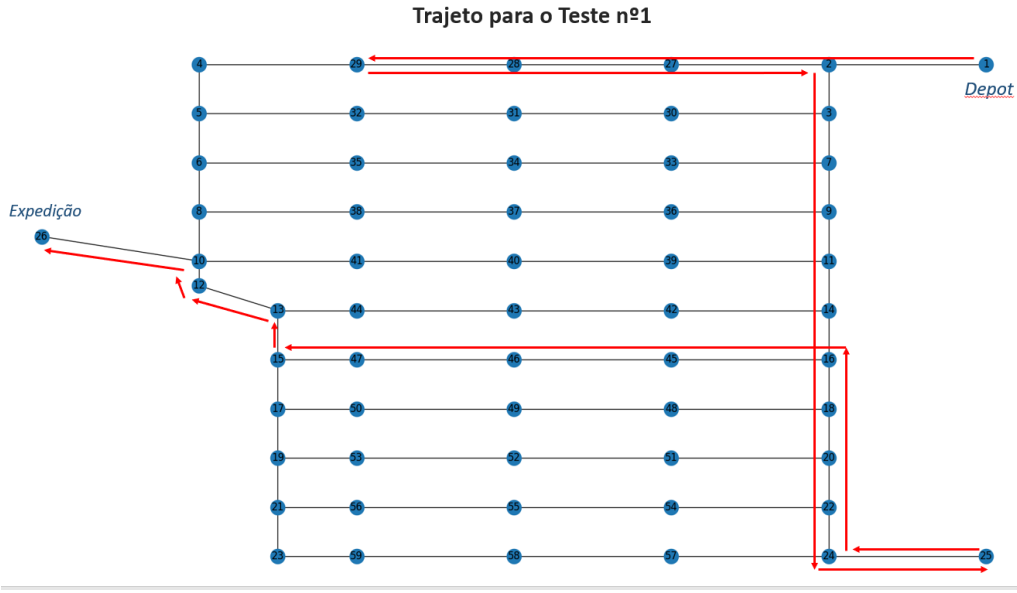


Figura 5.1: Trajeto para o teste número 1.

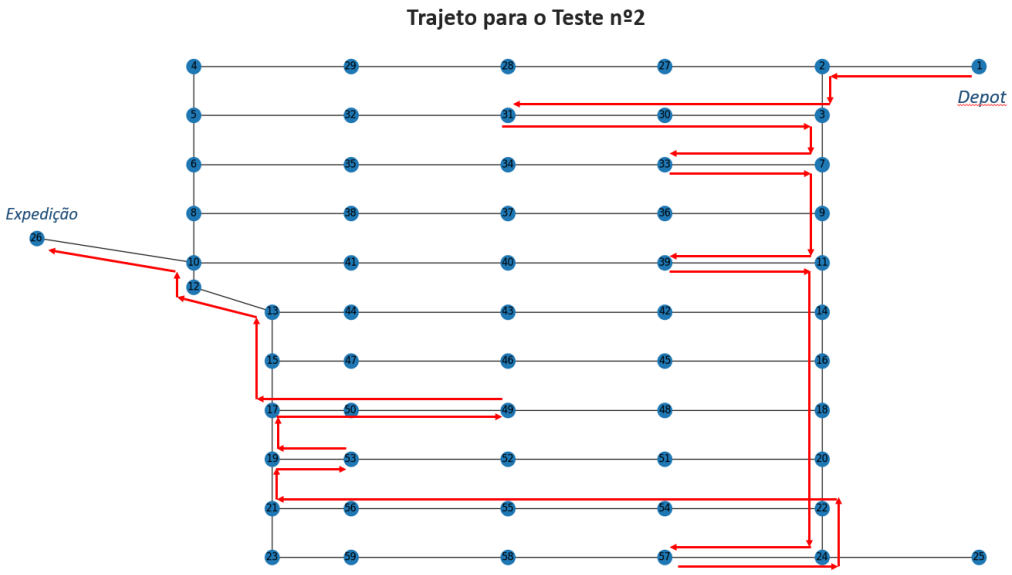


Figura 5.2: Trajeto para o teste número 2.

Tabela 5.2: Resultados obtidos nos 3 testes com algoritmo de *Picking* de Produto, com mais de 8 pontos de *picking*.

Teste nº	Pontos de Picking	Distância total (metros)
4	[3, 13, 17, 19, 31, 37, 38, 39, 40, 43, 44, 48, 51, 56]	465
5	[8, 11, 17, 34, 42, 46, 53, 55, 56]	414,1
6	[3, 9, 13, 17, 19, 20, 23, 27, 31, 37, 38, 39, 40, 43, 44, 48, 51, 56]	560,92

5.1.2 *Picking* de Palete

A natureza desta abordagem leva a que a sequência de pontos da percorrer seja mais extensa, uma vez que um operador depois de percorrer cada ponto de *picking* tem obrigatoriamente de ir até a zona de expedição. Deste modo, abdicou-se da representação gráfica destes resultados pois iria originar num grafo com uma densidade de setas tão elevada que ficaria ilegível, e para além disso a lógica é a mesma utilizada na abordagem anterior. Assim sendo, os resultados são apresentados na tabela 5.3.

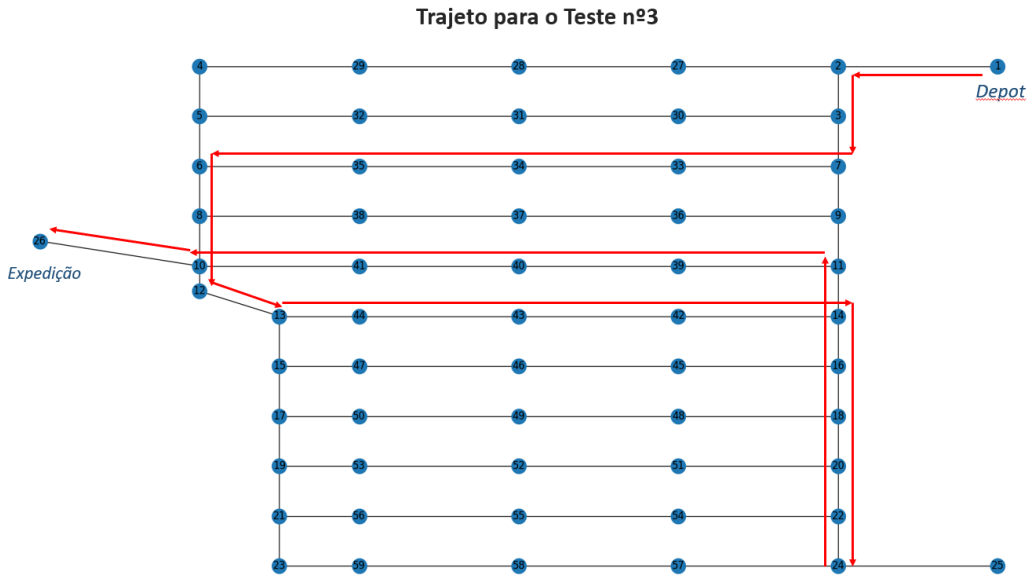


Figura 5.3: Trajeto para o teste número 3.

Tabela 5.3: Resultados obtidos no algoritmo de *Picking* de Palete.

Teste nº	Pontos de Picking	Distância total (metros)
7	[8, 12, 14, 15, 16]	540,4
8	[7, 9, 13, 17, 33, 35, 41, 43, 50, 53, 57]	1316,2
9	[14, 42, 53]	404,1

5.2 Resultados obtidos com um modelo matemático

Os resultados da utilização do modelo matemático do TSP, no ambiente *CPLEX*, foram obtidos tendo em conta os mesmos cenários de testes dos resultados do algoritmo de *Picking* de Produto. O propósito da utilização deste modelo para resolução dos problemas foi para haver um ponto de comparação dos resultados obtidos pelas abordagens heurísticas e fazer uma avaliação da sua eficácia e qualidade. Como a solução que se obtém através do modelo é ótima, podemos avaliar a qualidade das soluções obtidas através dos algoritmos comparando a sua proximidade às soluções do modelo. Os resultados obtidos com a utilização do modelo podem ser observados nas tabelas 5.4 e 5.5.

Para além das distâncias dos percursos obtidos pelo modelo, representaram-se também os percursos em si traçando-os novamente no grafo. Para os testes 1,2 e 3, os trajetos que se obtiveram com o Modelo foram exatamente iguais aos que se obtiveram usando o algoritmo, não sendo necessária a sua representação. Quanto aos testes 4,5 e 6, os trajetos que o Modelo sugere já diferem dos obtidos com o algoritmo, e podem ser observados nas figuras 5.7, 5.8 e 5.9.

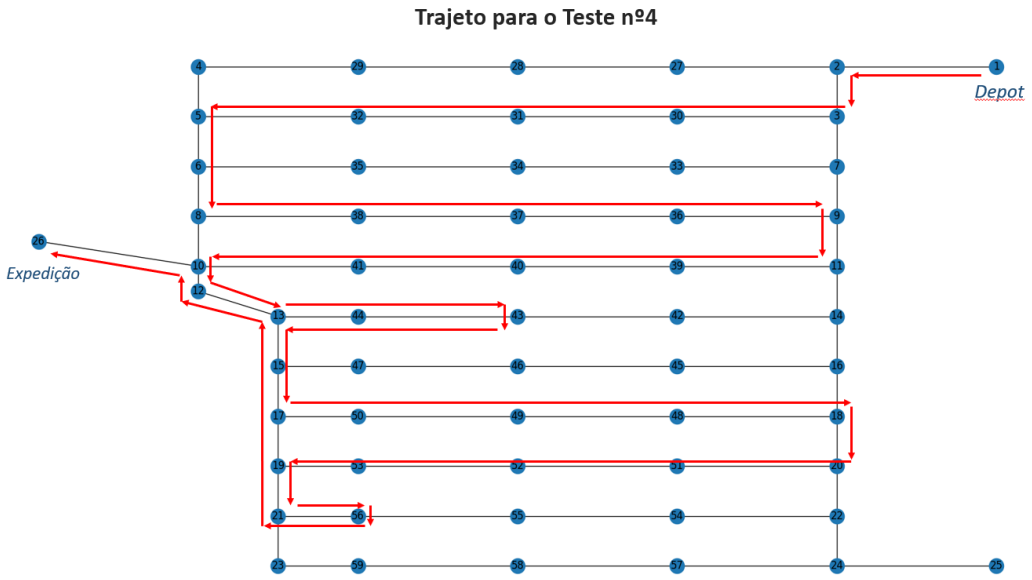


Figura 5.4: Trajeto para o teste número 4.

Tabela 5.4: Resultados obtidos com o modelo, dos 3 primeiros testes para abordagem de *Picking* de Produto.

Teste c/ Modelo	Pontos de Picking	Distância total (metros)	CPU (seg)	GAP (%)
1	[25, 29, 47]	371,92	0,03	0
2	[57, 55, 53, 49, 39, 33, 31, 21]	447,2	0,14	0
3	[44, 34, 24, 20, 8, 7, 6]	385,5	0,03	0

5.3 Análise crítica dos Resultados do algoritmo *vs* Resultados do modelo

Depois de obtidos os resultados das soluções propostas, é necessário proceder à sua interpretação, de forma a perceber a sua eficência e pertinência. Desta forma realizou-se uma tabela (tabela 5.6) onde foram aglomerados todos os resultados obtidos com o algoritmo de *Picking* de Produto e com o Modelo, para que a sua análise seja feita com clareza. Podemos afirmar, numa prespetiva crítica do problema, que nos testes realizados a distância média que um operador iria percorrer numa tarefa de *picking* é cerca de 1/2 quilómetro, ou seja, bastante significativa. Uma otimização que reduza apenas alguns metros, pode resultar em centenas de quilómetros que se evitam percorrer, a longo prazo.

Passando concretamente à análise dos resultados obtidos, para a abordagem de *Picking* de Palete, verificamos que se o número de pontos de *picking* não for muito grande, obtemos trajetos comprimentos razoáveis a rondar os 500 metros. Para o caso do teste

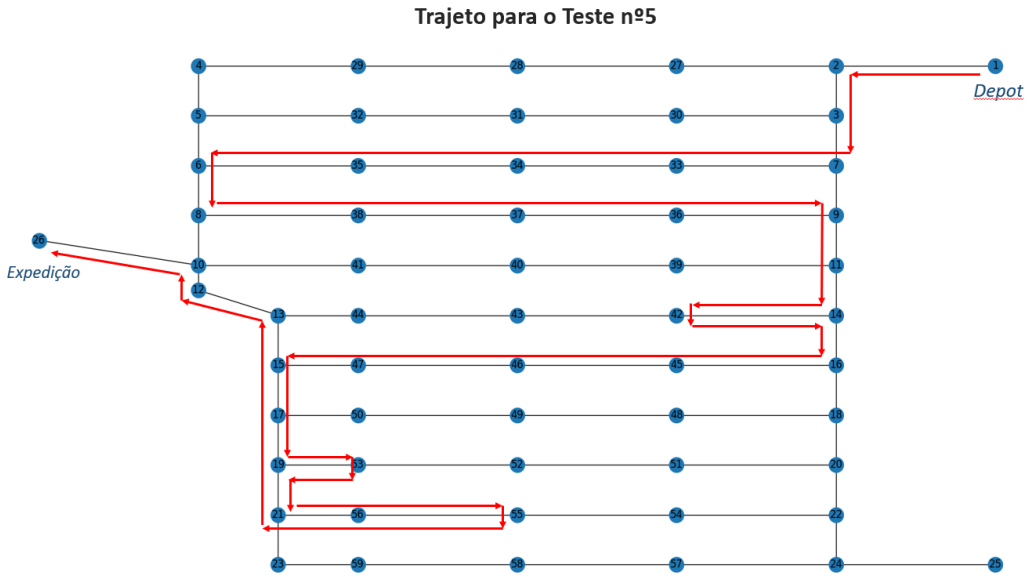


Figura 5.5: Trajeto para o teste número 5.

Tabela 5.5: Resultados obtidos com o modelo, dos 3 últimos testes para abordagem de *Picking* de Produto.

Teste c/ Modelo	Pontos de Picking	Distância total (metros)	CPU (seg)	GAP (%)
4	[3, 13, 17, 19, 31, 37, 38, 39, 40, 43, 44, 48, 51, 56]	445,45	0,19	9,33
5	[8, 11, 17, 34, 42, 46, 53, 55, 56]	345,6	0,06	0
6	[3, 9, 13, 17, 19, 20, 23, 27, 31, 37, 38, 39, 40, 43, 44, 48, 51, 56]	539,62	538	0,29

7, como o número de pontos de *picking* aumentou significativamente, o comprimento do trajeto obtido ultrapassou os 1300 metros. Isto deve-se à natureza do problema, onde um operador ao realizar a tarefa de *picking* tem que fazer várias vezes o trajeto Ponto de *picking*-Ponto Final. Uma vez que no cenário real da empresa não existe nenhum critério para a tarefa, este algoritmo pode ser uma boa base para os operadores.

Para a abordagem de *Picking* de Produto, o comprimento do trajeto que se obteve para o algoritmo foi exatamente igual ao do modelo, então, podemos afirmar que no algoritmo de *Picking* de Produto para o caso do número de pontos de *picking* ser inferior ou igual a 8, ele devolve uma solução ótima, num intervalo de tempo aceitável (no caso limite de 8 pontos de picking cerca de 19 segundos). Para os testes 4, 5 e 6 com o algoritmo de *Picking* de Produto, os resultados já diferem um pouco dos resultados do Modelo, isto porque é adotada uma heurística não exata, para diminuir o peso computacional. Ainda assim, pode-se afirmar que se encontram bastante próximos, uma vez que para os testes 4 e 6, a diferença de comprimento é cerca de 20 metros e para o teste 5 cerca de 70 metros.

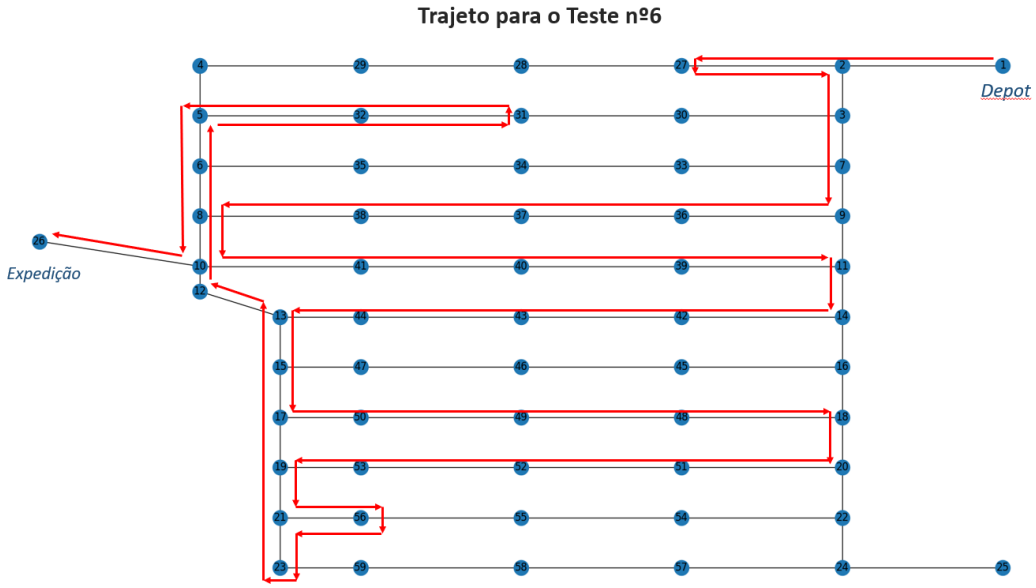


Figura 5.6: Trajeto para o teste número 6.

Este resultado era esperado, uma vez que o Modelo nos devolve a solução ótima. No entanto, para o teste 6 onde são necessários percorrer 18 pontos de *picking*, ele demorou quase 9 minutos para obter a solução, o que nos leva a conlcuir que quando o número de pontos de *picking* é elevado, o tempo de computação do modelo começa a ser uma variável crítica, e a partir desse momento, o algoritmo apresenta uma clara vantagem dado que apresenta uma solução significativamente próxima do ótimo num intervalo de tempo bastante reduzido.

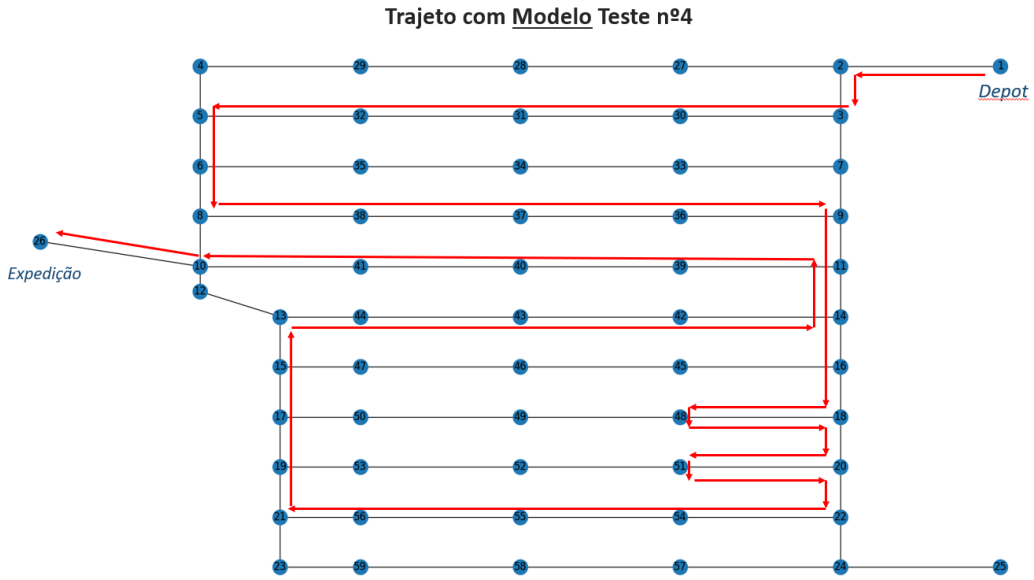


Figura 5.7: Trajeto obtido através do modelo para o teste número 4.

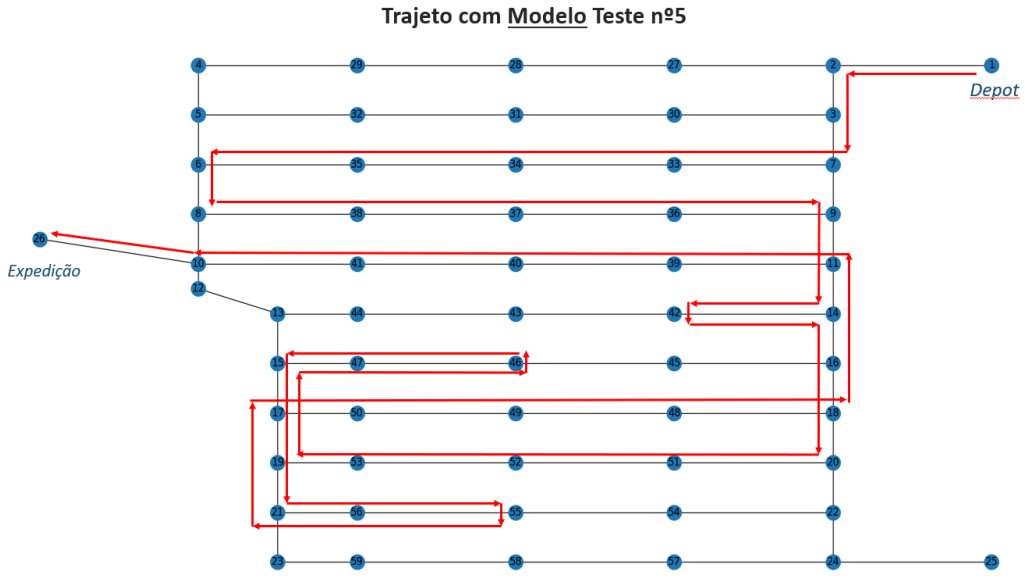


Figura 5.8: Trajeto obtido através do modelo para o teste número 5.

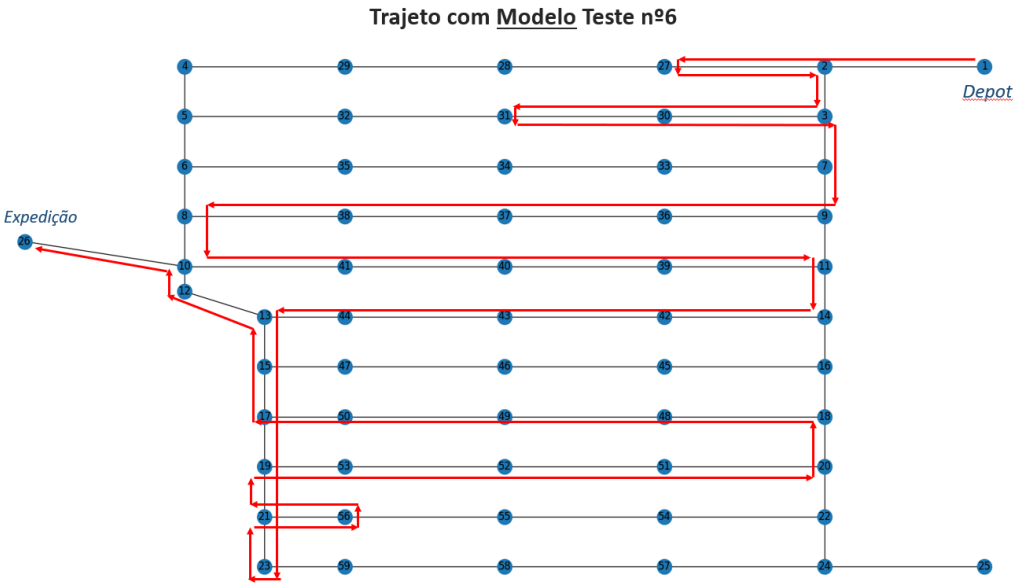


Figura 5.9: Trajeto obtido através do modelo para o teste número 6.

Tabela 5.6: Conjunto de todos resultados dos testes obtidos com o modelo e com o algoritmo de *Picking* de Produto.

Teste nº	Pontos de Picking	Distância c/ Algoritmo (metros)	CPU algoritmo (seg)	Distância com Modelo (metros)	CPU modelo (seg)
1	[25, 29, 47]	371,92	<1	371,92	0,03
2	[57, 55, 53, 49, 39, 33, 31, 21]	447,2	18,78	447,2	0,14
3	[44, 34, 24, 20, 8, 7, 6]	385,5	3	385,5	0,03
4	[3, 13, 17, 19, 31, 37, 38, 39, 40, 43, 44, 48, 51, 56]	465	<1	445,45	0,19
5	[8, 11, 17, 34, 42, 46, 53, 55, 56]	414,1	<1	345,6	0,06
6	[3, 9, 13, 17, 19, 20, 23, 27, 31, 37, 38, 39, 40, 43, 44, 48, 51, 56]	560,92	<1	539,62	538

Capítulo 6

Conclusões e trabalhos futuros

Por fim, o que se pode retirar da análise de resultados é que os algoritmos desenvolvidos são bastante eficientes, devolvendo sempre soluções ótimas ou muito boas. Essa solução é ótima até 8 pontos de *picking*, e aproximada, mas ainda assim perto do ótimo, para um número de pontos de *picking* superior, sendo que neste último caso o algoritmo se apresenta mais rápido no geral.

A solução apresentada, relativamente aos trabalhos analisados na secção 2.5, é constituída por um conjunto alargado de abordagens para cada fase do problema o que a torna numa solução flexível que facilmente se pode adaptar a problemas semelhantes, mas em diferentes contextos. O sistema desenvolvido tem também outra particularidade que o distingue dos demais, que é o facto de fazer a análise do número de pontos de *picking* a percorrer antes de devolver uma solução concreta. Isto permite que se obtenha uma solução ótima para casos em que o tempo computacional não é uma variável crítica. Este tipo de abordagem, em que se analisa primeiro a complexidade computacional antes de escolher a heurística a utilizar, raramente se encontra em problemas de planeamento de rotas.

Tendo os operadores em sua posse, para além da informação dos produtos que têm de recolher, também o trajeto que devem percorrer, é possível reduzir custos associados a desperdícios de tempo e tornar o processo mais eficiente, processo esse que tem um peso percentual significativo relativamente às atividades logísticas da empresa no geral. Adicionalmente, a empresa passa a ter em sua posse o comprimento dos trajetos realizados pelos operadores, que podem ser guardados numa base de dados para uma posterior análise de desempenho, seguindo uma filosofia de melhoria contínua, dado que, com esta informação podem ser detetadas possíveis oportunidades de melhoria. Para além disso, num trabalho futuro onde se realizasse uma implementação detalhada e sincronizada com a infraestrutura digital da empresa, o sistema desenvolvido, iria automatizar a atividade de *picking*. Uma vez que existe uma ligação direta à BD, com um pequeno ajuste ao contexto real da empresa, é possível que a chegada de um pedido de preparação de encomenda acione automaticamente o sistema para calcular a rota de recolha dos produtos, sem que seja necessária a intervenção de terceiros.

Este trabalho fez com que fosse preciso dominar várias ferramentas importantes para qualquer engenheiro. Foi necessário explorar linguagens de programação como *Python* e *SQL*, desenvolver a capacidade de manipulação de bases de dados e a familiarização com a utilização de *API's* e também desenvolver competências para a utilização de modelos com o *software CPLEX*. Para além disso, foi também possível desenvolver conhecimentos

aprofundados na área da logística, principalmente sobre a atividade de *picking*. Ficou claro que esta atividade tem um peso bastante relevante, e as ineficiências a ela associadas, não devem de todo ser descartadas. O desenvolvimento do algoritmo estimulou o uso de raciocínios lógicos para a resolução de problemas, que é um *soft-skill* de bastante valor. Outra capacidade bastante importante, que derivou da realização do presente projeto, foi a perceção da relevância que o peso computacional tem neste tipo de problemas. Por vezes o objetivo pode não ser chegar à solução ótima, pois o problema pode ter tais proporções, que alcançar o ótimo pode se tornar uma tarefa muito difícil ou até mesmo impossível. Assim sendo, é crucial para qualquer trabalho de otimização, haver um equilíbrio entre carga computacional e exatidão da solução, já que, um sistema que encontre a solução ótima para um determinado problema, facilmente se pode tornar inapto, com um ligeiro aumento da complexidade do mesmo.

O sistema desenvolvido tem o intuito de auxiliar esta importante atividade de *picking*, mas também foi pensado para ser uma solução genérica, para que seja possível a sua adaptação a problemas logísticos semelhantes, com uma existência abundante na atualidade.

É cada vez mais importante nos dias de hoje, dar uso a dados que temos em nossa posse que, manipulados corretamente, podem-se transformar em informação que pode ser utilizada para resolver problemas como o do presente trabalho.

O próximo passo seria a implementação deste sistema com a base de dados real da empresa, de forma a rastrear e melhorar a produtividade do processo num contexto real.

No futuro seria interessante desenvolver uma solução para a reorganização do armazém que aumentasse a produtividade de todas as tarefas inerentes, assim como a tarefa de *picking*, adicionando aos corredores vários pontos de *picking* tal como foi considerado na abordagem conceptual realizada. Seria também pertinente para a empresa, a adoção de um sistema de *picking Picker-to-Parts* de alto nível, onde a recolha de produtos seria feita a bordo de um equipamento que automaticamente parava nas zonas de *picking*. Tal sistema poderia ser usado em conjunto com a solução desenvolvida aumentando ainda mais a eficiência do processo.

Para concluir, o mercado atual do mundo em que vivemos, exige que qualquer organização se desenvolva tecnologicamente para não ficar para trás, e para além disso, existe também uma necessidade de informação, para perceber se determinadas atividades e processos estão a gerar desperdícios para a organização. Assim sendo, o desenvolvimento de sistemas de informação baseados em tecnologias atuais, como o que foi apresentado neste estudo, revelam uma importância crescente, e isso reflete-se na vontade, que cada vez mais organizações têm, de apostar em projetos que estejam envolvidos nesta revolução tecnológica que vivemos.

Bibliografia

[1] Dr. Tawfik M. Younis M. Tawfik El Masry, "Decision Support System in Supply Chain Management: Literature Review - EA Journals," *EA Journals*, vol. 5, no. 5, pp. 40–51, 2017.

[2] R. Riedel, "Facilities planning – 4th edition by j.a. tompkins, j.a. white, y.a. bozer and j.m.a. tanchoco," *International Journal of Production Research*, vol. 49, no. 24, pp. 7519–7520, 2011.

[3] S. Raquel, *Definição do layout de um armazém para order picking.* PhD thesis, University of Aveiro, 2018.

[4] R. de Koster, T. Le-Duc, and K. J. Roodbergen, "Design and control of warehouse order picking: A literature review," *European Journal of Operational Research*, vol. 182, no. 2, pp. 481–501, 2007.

[5] D. T. B. Santos, *Melhoria das Rotas de Picking em Armazém de Operador Logístico.* Master thesis, Faculdade de Engenharia Universidade do Porto, 2015.

[6] R. Manzini, "A top-down approach and a decision support system for the design and management of logistic networks," *Transportation Research Part E: Logistics and Transportation Review*, vol. 48, no. 6, pp. 1185–1204, 2012.

[7] P. Fernandes, *Otimização do Processo de Picking. Estudo de Caso: Armasul - Distribuidor de Materiais Elétricos, S.A..Dissertação submetida como requisito parcial para obtenção do grau de Mestre em Ciências Empresariais – Ramo Logística.* PhD thesis, Instituto Politécnico de Setúbal, 2017.

[8] G. Figueira, P. Amorim, L. Guimarães, M. Amorim-Lopes, F. Neves-Moreira, and B. Almada-Lobo, "A decision support system for the operational production planning and scheduling of an integrated pulp and paper mill," *Computers and Chemical Engineering*, vol. 77, pp. 85–104, 2015.

[9] C. C. Bozarth, R. B. Handfield, and H. J. Weiss, *Introduction to operations and supply chain management.* Pearson Prentice Hall Upper Saddle River, NJ, 2008.

[10] D. J. Power, *Decision Support Systems: Concepts and Resources for Managers - Daniel J. Power - Google Livros.* 2002.

[11] L. Wang, J. Song, and L. Shi, "Dynamic emergency logistics planning: models and heuristic algorithm," *Optimization Letters*, vol. 9, no. 8, pp. 1533–1552, 2015.

[12] D. M. Lambert Stock, James R., Ellram, Lisa M.,, *Fundamentals of logistics management*. Boston: Irwin/McGraw-Hill, 1998.

[13] J. A. Troche Escobar, M. D. S. F. B. Soares de Carvalho, and F. G. Mendonça Freires, "O uso de tecnologias para o processo de preparação de pedidos: implicações e proposições," *Revista Produção Online*, vol. 15, no. 1, p. 188, 2015.

[14] F. Dallari, G. Marchet, and M. Melacini, "Design of order picking system," *International Journal of Advanced Manufacturing Technology*, vol. 42, no. 1-2, pp. 1–12, 2009.

[15] Dr. Tawfik M. Younis M. Tawfik El Masry, "Decision Support System in Supply Chain Management: Literature Review - EA Journals," *EA Journals*, vol. 5, no. 5, pp. 40–51, 2017.

[16] I. . S. T. H. John J. BARTHOLDI, "PropCytoplasm Intensity StdIntensityEdge CorrBlue," p. 295, 2014.

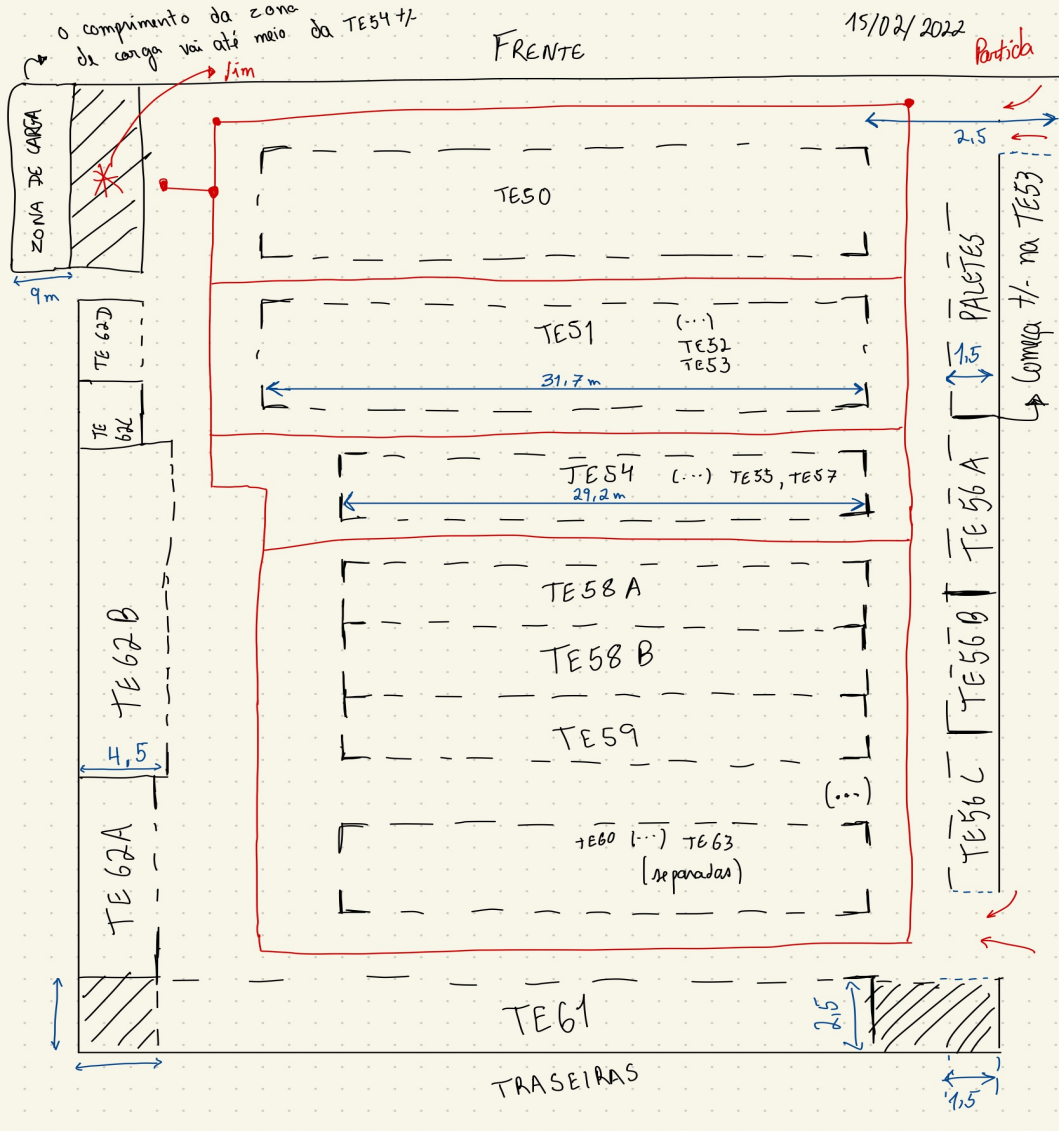
[17] I. Beker, V. Jevtić, and D. Dobrilović, "Shortest-path algorithms as a tools for inner transportation optimization," *International Journal of Industrial Engineering and Management*, vol. 3, no. 1, pp. 39–45, 2012.

[18] F. Mirahadi and B. Y. McCabe, "EvacuSafe: A real-time model for building evacuation based on Dijkstra's algorithm," *Journal of Building Engineering*, vol. 34, no. June, p. 101687, 2021.

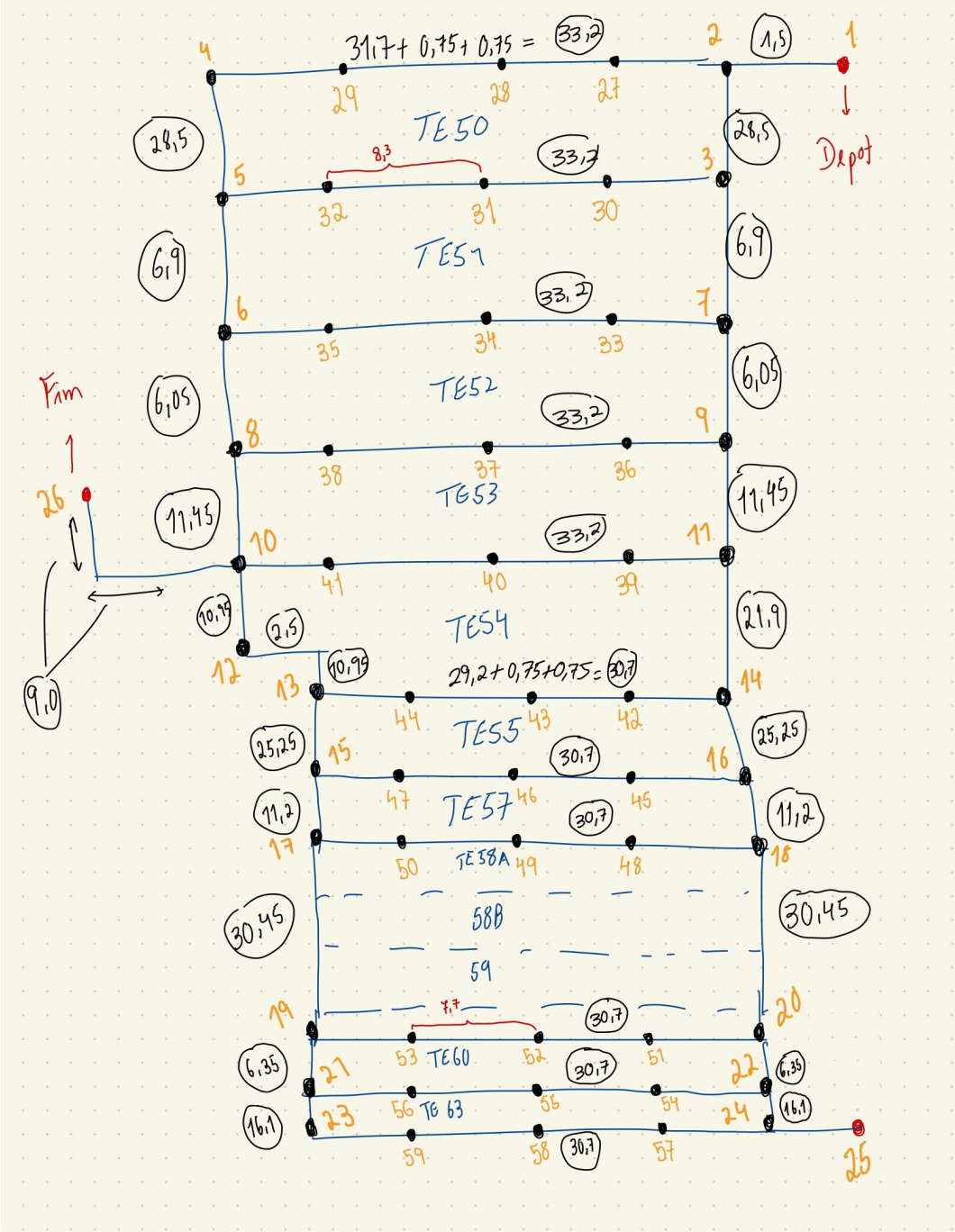
[19] P. E. Hart, N. J. Nilsson, and B. Raphael, "the Heuristic Determination," *IEEE Transactions of systems science and cybernetics*, no. 2, pp. 100–107, 1968.

Apêndice I

Anexos



Anexo A - Esquema do armazém com trajetos possíveis.



Anexo B - Esquema do armazém com trajetos, distâncias e pontos de *picking*

```
lista = Pontos_Picking
Pontos_Picking = remove_repetidos(lista)
print(Pontos_Picking)

from itertools import permutations
perm = permutations(Pontos_Picking)


Num_Pontos = Pontos_Picking.__len__()
k=1000
it = 0

if Num_Pontos <= 8:
#Continue statement
    for i in list(perm): #equivalent of...for n in [2,3,4,5,6,7,8,9]:
        y = Num_Pontos
        v = 0
        caminho_total = 0
        b = []
        i = [1] + list(i) + [26]
        it = it + 1
        print("permutação número ",it)
        print(i)
        while y>=0:
            a = nx.astar_path_length(G,source=i[v],target=i[v+1])
            b = b + nx.astar_path(G,source=i[v],target=i[v+1])
            caminho_total = caminho_total + a
            y=y-1
            v=v+1
        if k > caminho_total:
            k = caminho_total
            caminho_mais_curto = b
            continue #Will continue with the next loop
    nums = caminho_mais_curto
    caminho_mais_curto = remove_adjacent(nums)
    print("O Caminho mais curto é ", caminho_mais_curto,"\nO seu comprimento é ",k, "metros")

```

Anexo C - Código desenvolvido para *Picking* de Produto até 8 pontos de *picking*, inclusive.

```
elif Num_Pontos > 8:
    z = 1000
    zz = 1000
    y = Num_Pontos
    remover = 0
    Dicionario = {}
    for i in Pontos_Picking:
        Primeiro_Caminho_comp = nx.astar_path_length(G,source=1,target=i)
        if z > Primeiro_Caminho_comp:
            z = Primeiro_Caminho_comp
            Primeiro_Caminho = list(nx.astar_path(G, source=1,target=i))
            Primeiro_Caminho_length = nx.astar_path_length(G, source=1,target=i)
            remover = i
    Pontos_Picking.remove(remover)
    z = 1000
    y = 0
    Caminho_Meio = []
    Caminho_Meio_length = 0
    Dicionario = {}
    while Pontos_Picking.__len__()>0:
        for i in Pontos_Picking:
            Segundo_Caminho_comp = nx.astar_path_length(G,source=remover,target=i)
            Segundo_Caminho = nx.astar_path(G,source=remover,target=i)
            Dicionario[i] = Segundo_Caminho_comp
        Minimo = min(Dicionario, key=Dicionario.get)
        #Caminho_Meio.append(nx.astar_path(G,source=remover,target=Minimo))
        Caminho_Meio = Caminho_Meio + list(nx.astar_path(G,source=remover,target=Minimo))
        Caminho_Meio_length = Caminho_Meio_length + (nx.astar_path_length(G,source=remover,target=Minimo))
        remover = Minimo
        Pontos_Picking.remove(remover)
        Dicionario={}
    Ultimo_Caminho = list(nx.astar_path(G, source=Minimo, target=26))
    Ultimo_Caminho_length = nx.astar_path_length(G, source=Minimo, target=26)
    caminho_mais_curto = list(Primeiro_Caminho) + list(Caminho_Meio) + list(Ultimo_Caminho)
    caminho_mais_curto_length = Primeiro_Caminho_length + Caminho_Meio_length + Ultimo_Caminho_length
    nums = caminho_mais_curto
    caminho_mais_curto = remove_adjacent(nums)
    print("O Caminho mais curto é ", caminho_mais_curto, "\nO seu comprimento é: ", caminho_mais_curto_length)

```

Anexo D - Código desenvolvido para *Picking* de Produto com mais de 8 pontos de *picking*.

```
lista = Pontos_Picking
Pontos_Picking = remove_repetidos(lista)
print(Pontos_Picking)

k = 1000
for i in Pontos_Picking:

    Comp_Primeiro_Caminho = nx.astar_path_length(G,source=1,target=i)

    if k>Comp_Primeiro_Caminho:
        k=Comp_Primeiro_Caminho

        Primeiro_Caminho = nx.astar_path(G,source=1,target=i) + nx.astar_path(G,source=i,target=26)
        A = i
        print(k)

list.remove(Pontos_Picking,A)

Caminho_Restante = []
Caminho_Restante_comp = 0
for i in Pontos_Picking:
    B = nx.astar_path(G,source=26,target=i) + nx.astar_path(G,source=i,target=26)
    C = nx.astar_path_length(G,source=26,target=i) + nx.astar_path_length(G,source=i,target=26)
    Caminho_Restante = Caminho_Restante + B
    Caminho_Restante_comp = Caminho_Restante_comp + C

Caminho_Total = list(Primeiro_Caminho) + list(Caminho_Restante)
Caminho_Total_comp = k + Caminho_Restante_comp
nums = Caminho_Total
Caminho_Total = remove_adjacent(nums)
print("O Caminho mais curto é ", Caminho_Total,"\nO seu comprimento é ",Caminho_Total_comp, "metros")
```

Anexo E - Código desenvolvido para *Picking* de Palete.