Universidade de Lisboa
Faculdade de Ciências
Departamento de Estatística e Investigação Operacional

A Heuristic Approach for Multi-Product Capacitated Single-Allocation Hub Location Problems

Mestrado em Estatística e Investigação Operacional
Investigação Operacional

Eliana Fabíola Correia Fernandes

Dissertation supervised by:
Professor Francisco Saldanha da Gama

2015

RESUMO

Em redes onde o fluxo entre nodos é muito elevado (como pode ser o caso do transporte de pessoas e mercadorias ou até mesmo fluxo de dados numa rede), torna-se menos dispendioso criar pontos onde se concentram os fluxos provenientes das diferentes origens para depois serem consolidados e redistribuídos até aos destinos. A esses pontos dá-se o nome de *hubs*.

O problema de localização de hubs consiste na localização de hubs numa rede e na alocação de todos os nodos da rede a esses hubs, de modo a que se possa encaminhar os fluxos entre os pares origem-destino a menos que sejam hubs.

A rede constituída pelos hubs é normalmente definida como completa e não se permitem ligações diretas entre os pares origem-destino. Para além disso, assume-se que existe um factor de desconto para o fluxo que circula entre hubs.

Neste tipo de redes (*hub-and-spoke networks*) podem aparecer duas variantes, no que diz respeito à alocação dos nodos aos hubs: *single-allocation* e *multiple-allocation*. No primeiro caso, permite-se apenas uma ligação de cada nodo não hub a um hub de modo a que todo o fluxo com origem e destino a cada nodo saia e chegue a esse nodo através de apenas um hub. No caso em que se tem *multiple-allocation*, cada nodo poderá ser afecto a mais do que um hub e o fluxo que chega e sai desse nodo poderá usar mais do que um hub.

Algumas variantes que se poderão considerar para este problema incluem restrições de capacidade nos hubs (restrições que limitam a capacidade de um hub processar uma certa quantidade de fluxo de origem, limitações na capacidade total, limitações no processamento de fluxo que sai do hub, etc.), restrições de capacidade nos arcos, problemas multi-periódicos, presença de incerteza, o número de hubs ser fixo, o tipo de objectivo (minimizar custos, minimizar distâncias entre hubs, etc.) entre outras.

A necessidade de aproximar este tipo de problemas aos casos que se observam no mundo real leva à inclusão de cada vez mais restrições dando origem a mais variantes do problema.

Neste trabalho, será abordado o problema de localização de hubs na variante *single-Allocation*, com restrições de capacidade em relação ao fluxo que cada hub é capaz de processar. Para além disso, considera-se fluxos relativos a mais do que um tipo de produto. Este problema é designado por *Problema Multi-produto de Localização de Hubs com Capacidade*[1].

Cada hub poderá ser dedicado a processar apenas um tipo de produto, poderá processar mais do que um, ou mesmo todos. A rede de hubs é completa para cada produto mas, no entanto, se se considerar a rede de hubs para todos os produtos, esta poderá não ser completa.

[1]Multi-Product Capacitated Single-Allocation Hub Location Problem (MP-CSAHLP), de acordo com Correia et al. [17].

Como constatado em Correia et al. [17], no caso em que cada hub processa todos os tipos de produto, resolver o problema multi-produto ao invés de se resolver vários problemas, um para cada produto em separado, dá origem a melhores resultados.

A complexidade inerente a este tipo de problemas leva a que sejam classificados como problemas $\mathcal{NP}$-Hard pois não existem algoritmos que sejam capazes de os resolver em tempo polinomial. Por esta razão faz sentido desenvolver algoritmos heurísticos de modo a se conseguir obter, em tempo útil, soluções para instâncias maiores do problema .

Como referido em Meyer et al. [51], em problemas de localização de hubs, duas soluções com valores objectivo muito semelhantes poderão ser estruturalmente muito diferentes, e portanto, através um mecanismo de pesquisa local poderá ser muito difícil a passagem de uma boa solução para outra melhor. Por esta razão, neste trabalho opta-se por uma heurística que se baseia num método em que se constroem soluções repetidamente.

Para a construção das soluções, considerando que um processo de construção do tipo *Greedy* poderia dar origem a um número limitado de soluções e que as componentes da solução que são escolhidas por último são as piores, optou-se pelo desenvolvimento de um algoritmo de *Ant Colony Optimization* (ACO).

Esta meta-heurística baseia-se no comportamento apresentado pelas formigas quando estas procuram alimento. Quando uma formiga deixa a colónia em busca de alimento, no seu trajeto, deposita um químico (feromona) que pode ser detectado por outras formigas. Quanto maior a concentração de feromona, maior a atracão de cada formiga por esse trajeto e, portanto, os trajetos com maiores concentrações de feromonas serão percorridos por mais formigas. Por outro lado, se o caminho de ida e volta até ao alimento for mais curto, mais vezes será percorrido e maior será a concentração de feromona nesse caminho. O resultado destes dois tipos de reforço positivo nas concentrações de feromona nos trajetos percorridos pelas formigas, aliados ao facto de que existe evaporação do químico (a concentração de feromona diminui nos caminhos menos percorridos ao longo do tempo) dá origem aos "carreirinhos" de formigas que se podem observar na natureza e que normalmente representam o caminho mais curto entre o alimento e a colónia de formigas.

Considere-se o problema em questão em que se tem n nodos e p produtos. Para a representação das soluções, em vez de se considerar uma matriz binária $n \times n \times p$, onde o valor 1 representa uma afectação, considerou-se uma matriz $n \times p$, em que cada entrada representa, para cada produto, o hub ao qual o nodo foi afecto. O caso em que um nodo é afecto a si mesmo indica que esse nodo é hub para o produto correspondente. Este tipo de representação permite reduzir o tamanho da matriz e diminuir o uso da memória computacional.

Antes da construção de uma solução, é aplicado um pré-processamento que vai evitar, com base nas restrições do problema, que certas componentes da solução sejam consideradas durante o processo de construção da solução. Deste modo, reduz-se o espaço de procura de soluções e algum esforço computacional.

Para a construção de uma solução, escolhe-se o tamanho da colónia (o número de formigas que pertencem à colónia) e cada formiga vai escolhendo, sucessivamente, componentes da solução através de uma regra pseudo-aleatória onde algumas componentes da solução são escolhidas de um modo *greedy* e outras são escolhidas através de *roulette wheel selection.* A cada componente da solução é atribuído um valor inicial de feromona e, à medida que cada formiga vai adicionando componentes à solução, o valor da feromona associado à componente adicionada vai decrescendo, o que resulta na diminuição da probabilidade de que essa componente seja escolhida pela próxima formiga, dando origem à diversificação do conjunto de soluções construído por cada colónia. No fim, depois de todas as formigas terem construído uma solução, escolhe-se a melhor solução e reforça-se a concentração de feromona na melhor solução construída pela colónia. Se, por acaso, uma formiga der origem a uma solução não admissível, a solução construída por essa formiga não é considerada. Para mais detalhe em relação a este processo consultar Dorigo et al. [20].

Este tipo de algoritmo permite a inclusão de métodos de pesquisa local de modo a que a solução obtida por cada colónia seja melhorada. Com o objectivo de obter um algoritmo mais eficiente, escolheu-se incluir esta possibilidade e procedeu-se ao reforço da concentração de feromona após feita uma pesquisa local.

Na pesquisa local efectuada, usaram-se três tipos de vizinhança. Um deles fecha os hubs dedicados que só servem a si próprios e realoca-os a outros já abertos para esse mesmo produto. Outro, escolhe aleatoriamente um nodo alocado a um hub dedicado para um dado produto e realoca-o a outro hub dedicado ao mesmo produto. Um terceiro, escolhe um hub aleatoriamente e transforma-o num nodo, realocando-o a outro hub dedicado ao mesmo tipo de produto.

De modo a obter soluções iniciais melhores, explora-se a possibilidade de atribuir valores iniciais de feromona mais altos às componentes de solução pertencentes à solução da relaxação linear, na proporção do valor correspondente no caso das variáveis 0-1. Uma outra variação explorada consiste em fazer o reforço do valor de feromona às componentes da solução, apenas quando esta é a melhor de todas encontrada até ao momento, permitindo que haja evaporação de certas componentes de solução que poderão estar a ser escolhidas consecutivamente e permitindo que se escape mais facilmente de óptimos locais.

Após implementação do algoritmo procede-se à fase dos testes computacionais em instâncias do problema com 10, 20, 25 e 40 nodos, 1, 2 e 3 produtos e hubs que processam 1, 2 e 3 produtos.

As instâncias usadas nos testes computacionais pertencem ao *Australian Post data set* e foram adaptados por Correia et al. [17] de modo a que se tivesse dados para mais do que um tipo de produto.

Palavras-chave: Localização de hubs, Multi-produto, *Ant colony optimization*, *Single-Allocation*

ABSTRACT

In this thesis, an heuristic procedure is proposed for the the *multi-product capacitated single-allocation hub location problem.*

When addressing a problem in which it is necessary to determine the transportation of large commodity flows between many origin-destination (O-D) pairs, instead of using direct links, it becomes more efficient to design the networks in such a way that some of the nodes become consolidation centers or hubs. The *Multi-Product Capacitated Single-Allocation Hub Location Problem* (MP-CSAHLP according to Correia et al. [17]), is a $\mathcal{NP}$-Hard problem in which several types of flow are considered, making it possible to consider the case when multiple types of products are to be shipped between each O-D pair. It can be seen as an extension of the classical *Capacitated Single-Allocation Hub Location Problem.*

In the problem investigated in this work, no more than one hub can be located in each node and the hubs can be either *dedicated* (each hub can only handle one type of product) or *non-dedicated* (one hub can handle more than one type product). The hubs have capacity limitations regarding the incoming flow. Furthermore, the hub network is complete for each product but, when considering the hub network as a whole, it does not necessarily have to be complete. The goal is to locate the hubs in the network, allocate the non-hub nodes to the opened hubs and route the flow between each O-D pair. The objective is to minimize the total flow routing cost plus the setup costs of the hubs and costs of preparing the hubs to handle the different types of products.

In order to obtain feasible solutions to the above problem, an Ant Colony Optimization procedure is proposed, which is a constructive, population-based meta-heuristic based in the foraging behavior of ants. Indirect communication between the ants through pheromones reflects the colony search experience. High-quality solutions are found as an outcome of the global cooperation among all the ants of the colony. A preprocessing procedure is also proposed in which some solution components are forbidden based on the problems restrictions. Such preprocessing reduces the search space and thus may reduce the computational effort.

The proposed heuristic uses a single ant colony, which simultaneously chooses the hubs and allocates the nodes to the hubs. Once these solutions are found, the routing of the flow is computed in a short amount of time, using the optimization models for the MP-CSAHLP in which some variables (location and allocation) are fixed.

The results show that the proposed heuristic has the potential to find good quality solutions for the MP-CSAHLP and that its performance can be improved with finer parameter tuning, longer runs and more intense local search.

Keywords: Hub Location, Multi-Product, Ant Colony Optimization, Single-Allocation.

ACKNOWLEDGMENTS

I would like to thank Professor Francisco Saldanha da Gama for the all the support and guidance, giving me the tools to finish this this work.

I thank all the Professors that taught me so much during my master degree and contributed for my growth as an individual.

I would like to thank Ricardo Fialho for all the patience and help throughout writing this thesis.

Last but not the least, I would like to thank my family for supporting and believing in me.

Contents

List of Figures

List of Tables

“*Look deep into nature, and then you will understand everything better.*”

Albert Einstein

CHAPTER 1

Introduction

Hubs can be defined as "central facilities which act as switching points in networks connecting a set of interacting nodes" (see O'Kelly [54]). When addressing a problem in which it is necessary to determine the transportation of large commodity flows (such as mail delivery systems, public transportation systems telecommunications networks, etc.) between many origin-destination (O-D) pairs, instead of using direct links, it becomes more efficient to design the networks in such a way that some of the nodes become consolidation centers or hubs.

In such a network, the main goal is to decide which nodes are to be used as hubs and determine the allocation of the other nodes (spokes) to the hubs, in order to route the flows between the O-D pairs. This is called a *Hub Location Problem* (HLP).

It is usually assumed that the resulting hub network is a clique and that no direct connections between non-hub nodes is allowed. This type of network, often called a hub-and-spoke network, takes advantage of economies of scale by applying a discount factor (α) to the flows routed between hubs.

Depending on how spokes are allocated to the hubs, two types of hub networks can be differentiated: *single-allocation* and *multiple-allocation*. In *single-allocation* networks, the flow originated and destined to each spoke is routed through only one hub. Therefore, each spoke can only be allocated to one hub. In *multiple-allocation*, the incoming and outgoing flow to each spoke can be routed through more than one hub and so it can be allocated to more than one hub.

Hub location problems can be of different types, depending on the objective function as well as on the constraints. Regarding the objective, some examples are the *p-Hub Median Problem* (pHMP), in which the number of hubs to be opened is fixed to a value p and the objective is to minimize the total flow cost[1], the *p-Hub Center Problem* (pHCP), a minimax type problem where the objective can be to minimize the maximum cost of any O-D pair, any single link or of movement between a hub and a origin or destination, as defined by Campbell [8]. Concerning the constraints we can distinguish capacity restrictions of different types such as upper (and sometimes lower) limits applied to the flow on each arc, capacity limits to the total/incoming/outgoing flow that each hub can handle or, as already mentioned, defining the number of hubs to be opened *a priori* to a fixed value p.

[1]Fixed costs of opening the hubs can be added to the objective function, resulting in a *p-Hub Location Problem* with fixed costs.

More details on hub location problems can be found in Contreras [13], in Campbell [8], in Alumur and Kara [3], in Campbell and O'Kelly [10] and in Farahani et al. [36]. Furthermore, some other additional features can contribute to the existence of a wider range of types of hub location problems such as the inclusion of stochastic elements, time restrictions, multi-period modeling and many more (see Contreras [13]).

One of those additional features, was proposed by Correia et al. [17], which gave origin to the *Multi-Product Capacitated Single-Allocation Hub Location Problem* (MP-CSAHLP). In this extension, several types of flows are considered, making it possible to consider the case when multiple types of products are to be shipped between the O-D pairs.

In the MP-CSAHLP, the hubs can be classified as dedicated (each hub handles only one type of flow) or non-dedicated (the hubs can handle more than one product). There are capacity constraints for the incoming flow to each hub. Moreover, the hub network is complete for each product although, when considering the hub network as a whole, it does not necessarily have to be complete.

Even though an optimization model was proposed in Correia et al. [17], this is a $\mathcal{NP}$-Hard problem and so, the main focus of this thesis is develop a heuristic for finding feasible solutions to the problem.

As referred by Meyer et al. [51], in hub location problems, two solutions with similar objective values can be structurally very different and moving from good solutions to better solutions, when using a local search mechanism, can be very difficult. For this reason, a heuristic that constructs repeatedly solutions was chosen in the current work. Construction heuristics are often based in a greedy mechanisms, often leading to a limited number of solutions with the disadvantage that in the solution construction process, worse solution components are often chosen in the last steps of the algorithm. In order to avoid this behaviour, Ant Colony Optimization (ACO, Dorigo et al. [20, 21], Dorigo and Caro [22], Dorigo and Stützle [29, 30]) was chosen to find good quality solutions to the problem.

ACO is based on the foraging behaviour of ants (when ants leave their colony in search for food) and on stigmergic communication (indirect communication between the ants trough pheromones). When an ant leaves the colony in search for food, it deposits a certain amount of pheromone in the traversed path. Other ants can perceive the pheromone and tend to travel through the paths that have higher pheromone concentration, also depositing pheromone in those paths, increasing the pheromone concentration. On the other hand, the shortest the path to the food source and back to the colony, the more times it will be traveled. This gives origin to a higher concentration of pheromones in the shortest paths to food. These two *positive feedback* processes, plus the fact that the pheromone evaporates over time (lowering the concentration of that chemical on the least traveled paths) is the reason why a line of ants can often be observed in nature and normally represents the shortest path between the food source and the ant colony.

ACO was first applied to the Traveling Salesman Problem (TSP) by Dorigo and Stützle [30] and can be applied to the problem in a straightforward way. In the case of the MP-CSAHLP, when considering a network with n nodes and p products, instead of considering a binary matrix $n \times n \times p$ as a solution representation (where each matrix entry x_{ij}^{p} equal to 1 represents that node i is affected to hub j for product p), an $n \times p$ matrix is considered, where each entry represents the hub to which the node is allocated for each product (the case where a node is allocated to itself indicates that the node is a hub). This kind of solution representation allows a reduction on the size of the solution matrix by a factor of n resulting in a decrease in the memory usage for the computational tests.

For the solution construction, the size of the colony (number of ants to be considered) is chosen and then, each ant builds a solution by successively adding solution components to the solution being built. Each component is chosen by using a *pseudo-random proportional rule* (Dorigo and Gambardella [25], Dorigo and Stützle [30]) where some solution components are chosen in a greedy fashion and others are chosen using roulette wheel selection.

To each solution component, an initial pheromone value is assigned and, every time an ant adds a solution component, the pheromone value associated with that solution component decreases, making that solution component less attractive to the next ant, resulting in a diversification of the solution set built by each ant colony. After all ants in the colony have built a solution, the one with the best objective value is chosen and the pheromone levels of each of the solution components of that solution is increased. If a solution built by an ant is infeasible, it is not considered and the ant is declared as a dead ant.

Before updating the pheromone levels of the best solution components, local search can be applied in order to improve the solution quality, yielding a local optimum. Adding this feature to the algorithm improves its performance and, since the neighbourhood used during the solution construction is different than the one used in the local search phase, there is a high probability of getting a better solution (Dorigo and Stützle [29]).

For the local search procedure three different neighbourhoods where explored, based on the local search operators for the *Capacitated Single-Allocation Hub Location Problem* (CSAHLP) defined by Ernst and Krishnamoorthy [35]. These neighbourhoods are:

1. **CloseHub**: Dedicated hubs that serve only themselves are made spokes and relocated to another hub dedicated to handling the same product, if the hub only processes one type of product, it is also closed as a hub;

2. **CloseRandomHub**: For a randomly chosen product p, a hub processing that product is chosen also randomly, made spoke and relocated to another hub dedicated to handling the same product. In this case, all spokes allocated to that closing hub are also relocated to other hubs dedicated to the same product. If the hub only processes one type of product, it is also closed as a hub;

3. **RelocateNode**: For some randomly chosen product p, a node is randomly chosen and relocated to another hub handling p.

These moves are applied only if feasibility is maintained. If the solution has a better objective value, the change in the solution is carried out.

Some variations to the ACO algorithm proposed are explored. One uses stronger pheromone initial values for the solution components that belong to the linear relaxation solution (the increase in the values is done proportionally to the value of the solution obtained for components associated with 0-1 decision variables). Another variation tested is based on Elitist Ant System investigated by Dorigo et al. [27, 28] where, instead of providing a pheromone reinforcement on the solution components of the best solution found by each colony, a choice is made on increasing the pheromone values on the solution components, only if it is the best solution found so far. This allows pheromone evaporation on the solution components that are being successively chosen but are not part of a good solution, making it easier to escape from local optima.

After the implementation of the algorithm, a series of computational tests is run on instances with 10, 20, 25 and 40 nodes for 1, 2 and 3 types of products and considering all possibilities for the maximum number of products that a hub can handle. The instances used belong to the Australian Post data set and, in order to consider more than one type of flow, the data were adapted by Correia et al. [17].

The remainder of this thesis is organized as follows. In the next chapter, hub location problems are described and a literature review is done regarding this class of problems, that includes heuristic approaches that have been proposed and the application of ACO heuristics to some problems. In Chapter 3 the MP-CSAHLP is described in detail and a MILP formulation existing in the literature is revisited. In Chapter 4, some basic knowledge on the ACO meta-heuristic is provided and some existing algorithms are presented. Chapter 5 is dedicated to the description of the algorithm used to find the feasible solutions for the problem and also some results are reported and discussed. Finally, in Chapter 6 the performance of the algorithm is discussed and some conclusions are drawn.

CHAPTER 2

Literature Review

In order to take advantage of economies of scale, in a network where flows must be routed between many origin-destination pairs, it is better to choose some of the nodes to act as switching, transshipment and sorting points. This idea was first explored in 1969, by Goldman [39] and later, in 1986, by O'Kelly [53, 54]. Since then, much work has been done by many researchers on different classes of *Hub Location Problems* with regard to new formulations and development of exact and heuristic algorithms for tackling such complex problems.

The reaminder of this Chapter is organized as follows. In Section 2.1 some problems and techniques more closely related to the problem to be solved, the MP-CSAHLP, are revisited and in Section 2.2 some heuristic approaches for tackling similar problems are presented. Finally, in in Section 2.3 some conclusions are discussed.

2.1 Hub Location and Hub Median Problems

In 1987, O'Kelly [55] proposed a quadratic integer programming formulation for a hub location problem within the context of airline passenger networks. According to Alumur and Kara [3] this was the first mathematical formulation for this type of problems.

Consider a network where flows must be routed between origin-destination pairs, there are no capacity constraints and the number of hubs to be opened, p, is defined *a priori*. The problem studied by O'Kelly [55] consists of locating p hubs, allocating the nodes to the hubs (each node can only be allocated to one hub) and routing the flows through the hub network in order to minimize the flow routing costs. This problem is known as the *Uncapacitated Single Allocation p-Hub Median Problem.*

Denoting by N the set of nodes in the network, the decision variables are defined as follows:

$$X_{ik} = \begin{cases} 1, & \text{if node } i \in N \text{ is linked to a hub at } k \in N, \\ 0, & \text{otherwise.} \end{cases}$$

$$X_{ii} = \begin{cases} 1, & \text{if node } i \in N \text{ is a hub,} \\ 0, & \text{otherwise.} \end{cases}$$

Next, we introduce some notation, in order to present the model:

W_{ij}	:	number of units of flow between nodes i and j (i, $j \in N$, $W_{ii} = 0$ by assumption);
C_{ij}	:	transportation cost of a unit of flow between node i and node j (i, $j \in N$, $C_{ii} = 0$ by assumption);
p	:	total number of hubs to be constructed;
n	:	total number of cities to be interconnected, i. e., $n = \|N\|$;
α	:	discount factor applied to inter-hub connections (in order to take in to account the economies of scale, $\alpha \leq 1$).

The hub location problem studied by O'Kelly can then be formulated as:

QP :

$$\textit{minimize} \quad \sum_{i\in N}\sum_{j\in N} W_{ij}\left(\sum_{k\in N} X_{ik}C_{ik} + \sum_{m\in N} X_{jm}C_{jm} + \alpha\sum_{k\in N}\sum_{m\in N} X_{ik}X_{jm}C_{km}\right), \tag{2.1}$$

$$\textit{subject to}: \quad (n-p-1)X_{jj} - \sum_{i\in N} X_{ij} \geq 0, \qquad \text{for all } j \in N, \tag{2.2}$$

$$\sum_{j\in N} X_{ij} = 1, \qquad \text{for all } i \in N, \tag{2.3}$$

$$\sum_{j\in N} X_{jj} = p, \tag{2.4}$$

$$X_{ij} \in \{0,1\}, \qquad \text{for all } i\text{ , } j \in N. \tag{2.5}$$

In the above model, the objective function (2.1) quantifies the total flow routing cost, constraints (2.2) ensure that nodes are only allocated to hubs[1], constraints (2.3) guarantee that each node can only be allocated to one hub; constraints (2.4) state that the number of hubs to be opened is p and finally, constraints (2.5) define the domain of the decision variables.

In 1992, in order to extend his formulation, O'Kelly [56] added fixed costs to the objective function, i.e., the cost term $\sum_{j\in N} X_{jj}f_j$ was added, where f_j represents the fixed cost of opening a hub. Even though these formulations contain few variables, the quadratic non-convex nature of the objective function makes the problem difficult to solve.

The quadratic integer programming formulation of O'Kelly was linearized by Campbell [8, 9], Aykin [4], Skorin-Kapov et al. [68] and Ernst and Krishnamoorthy [33]. The first four works considered 4-indexed variables; the latter reduced the problem size by decreasing the number of indices to three.

Campbell [8] starts by presenting a formulation for the *p-Hub Median Problem* where the following decision variables are considered:

Y_{ijkm} : fraction of flow from location (origin) i to location (destination) j that is routed via hubs at locations k and m in that order (i, j, k, $m \in N$);

$$Z_k = \begin{cases} 1, & \text{if location } k \in N \text{ is a hub,} \\ 0, & \text{otherwise;} \end{cases}$$

[1] $X_{jj} = 1$ means that node j is a hub, therefore, any node can be allocated to j and at most $n - p + 1$ nodes can be allocated to j (including the allocation of hub j to itself). On the other hand, $X_{jj} = 0$ means that node j is not a hub and, for that reason, no nodes can be allocated to j ($\sum_{i\in N} = 0$, for all $j \in N$).

Also, the following parameters are considered:

$$\begin{aligned} c_{ij} \quad &: \quad \text{standard cost from location } i \text{ to location } j; \\ C_{ijkm} &= \quad c_{ik} + c_{mj} + \alpha c_{km}. \end{aligned}$$

And the problem is formulated as:

$p - HM$

$$\textit{minimize} \quad \sum_{i \in N} \sum_{j \in N} \sum_{k \in N} \sum_{m \in N} W_{ij} Y_{ijkm} C_{ijkm}, \tag{2.6}$$

$$\begin{aligned} \textit{subject to}: \quad & \sum_{k \in N} Z_k = p, && && (2.7) \\ & \sum_{k \in N} \sum_{m \in N} Y_{ijkm} = 1, && \text{for all } i, j \in N, && (2.8) \\ & Y_{ijkm} \leq Z_k, && \text{for all } i, j, k, m \in N, && (2.9) \\ & Y_{ijkm} \leq Z_m, && \text{for all } i, j, k, m \in N, && (2.10) \\ & 0 \leq Y_{ijkm} \leq 1, && \text{for all } i, j, k, m \in N. && (2.11) \\ & Z_k \in \{0, 1\}, && \text{for all } k \in N, && (2.12) \end{aligned}$$

In the above model, N, W_{ij} and p have the meaning already presented before. The objective function (2.6) aims to minimize the total transportation costs, constraints (2.7) establish that the number of hubs to be opened must be equal to p, constraints (2.8) assure that the flow for every O-D pair is routed via at least one hub, constraints (2.9) and (2.10) allow allocation only to hubs. Constraints (2.12), (2.11) define the domain of the decision variables. The variables Y_{ijkm}, as defined lead to the *Multiple-Allocation* version of the problem but if set to be integer, lead to the *Single-Allocation* version.

Next, Campbell [8] introduces the idea of flow thresholds and fixed costs for the links connecting hubs to spokes. The flow thresholds T_{ik} defines the minimum flow value allowing service on the link between spoke $i \in N$ and hub $k \in N$. If the flow on that link is greater than zero then, a cost S_{ik} is incurred. The larger these values, the lower the number of hub-spoke links that will emerge in the solution. When $T_{ik} = \sum_{j \in N}(W_{ij} + W_{ji})$, the threshold reaches the maximum value, and each demand point is allocated to a single hub, resulting in a *single allocation* problem.

The resulting linearization of the *Uncapacitated Single Allocation p-Hub Median Problem* (USApHMP) can be formulated considering the variables Y_{ijkm} and Z_k introduced before and the variables X_{ik} defined as follows:

$$X_{ik} \quad = \quad \begin{cases} 1, & \text{if location } i \text{ is allocated to the hub at location } k \ (i,\, j \in N), \\ 0, & \text{otherwise;} \end{cases}$$

The linear formulation of the of the USApHMP presented by Campbell [8] is then:

Constraints (2.13) define that, if X_{ik} is equal to one, all the flow originated and destined to node i must go through hub k (*Single-Allocation*) and constraints (2.14) only allow connections from node i to node k, if node k is a hub. Constraints (2.12), (2.11) and (2.15) define the solution variables domain.

$p - HM - 1L$

$$minimize \quad (2.6),$$

$$subject\,to : (2.7), (2.12), (2.11), (2.8);$$

$$\sum_{j \in N} \sum_{m \in N} \big(W_{ij} Y_{ijkm} + W_{ji} Y_{jimk}\big) = \sum_{j \in N} (W_{ij} + W_{ji}) X_{ik}, \qquad \text{for all } i, k \in N. \tag{2.13}$$

$$X_{ik} \leq Z_k, \qquad \text{for all } i, k; \tag{2.14}$$

$$0 \leq X_{ik} \leq 1, \qquad \text{and integer for all } i, k \in N. \tag{2.15}$$

In this work, Campbell also presents a formulation for the *Uncapacitated Hub Location Problem* where the the main difference resides in the fact that the number of hubs is not fixed and a fixed cost (associated with each potential hub location) is added to the objective function. The *p-hub center problem* and the a *hub covering problem* are formulated too. In the first center problem the objective is to minimize the maximum distance between any O-D pair and and in the covering problem, any O-D pair is covered by hubs k and m if the cost from i to j via k and m does not exceed a specific value.

Skorin-Kapov et al. [68] observed that the linear relaxations of the previous model resulted in highly fractional solutions and proposed a modification in which constraints (2.9) and (2.10) are replaced by the following ones:

$$\sum_{m \in N} Y_{ijkm} = X_{ik}, \quad \text{for all } i, j, k \in N, \text{ and} \tag{2.16}$$

$$\sum_{k \in N} Y_{ijkm} = X_{jm}, \quad \text{for all } i, j, m \in N. \tag{2.17}$$

These constraints are stronger and yield tighter linear relaxation bounds with integer solutions in almost all instances tested by Skorin-Kapov et al. [68].

Ernst and Krishnamoorthy [33] proposed a three index mixed integer Linear Programming (LP) formulation for the USApHMP with fewer constraints, reducing the size of the model. By treating the inter-hub transfers as a multi-commodity flow problem, where each commodity represents the traffic flow originating from a particular node, it is possible to remove the Y_{ijkl} variables.

Consider the following additional notation:

χ	:	discount factor applied to node-hub connections (collection);
δ	:	discount factor applied to hub-node connections (distribution);
O_i	:	traffic originated at node i ($O_i = \sum_{j \in N} W_{ij}$ for all $i \in N$);
D_i	:	traffic destined to node i ($D_i = \sum_{j \in N} W_{ji}$ for all $i \in N$);

Additionally, denote by Y_{kl}^i a new decision variable representing the total amount of traffic emanating from node i that is routed via hubs k and l.

The problem can now be written as follows:

$USApHMP-N$

$$minimize \quad \sum_{i\in N}\sum_{k\in N} d_{ik}X_{ik}(\chi O_i + \delta D_i) + \sum_{i\in N}\sum_{k\in N}\sum_{l\in N}\alpha d_{kl}Y^i_{kl} \tag{2.18}$$

$$subject\,to: \sum_{k\in N} X_{kk} = p, \tag{2.19}$$

$$\sum_{k\in N} X_{ik} = 1, \qquad \forall i \in N, \tag{2.20}$$

$$X_{ik} \le X_{kk}, \qquad \forall i,k \in N, \tag{2.21}$$

$$\sum_{l\in N} Y^i_{kl} - \sum_{l\in N} Y^i_{lk} = O_iX_{ik} - \sum_{j\in N} W_{ij}X_{jk}, \qquad \forall i,k \in N, \tag{2.22}$$

$$X_{ij} \in \{0,1\}, \qquad \forall i,k \in N, \tag{2.23}$$

$$Y^i_{kl} \ge 0, \qquad \forall i,k,l \in N. \tag{2.24}$$

In the objective function (2.18), not only the inter-hub discount factor is considered (α) but the the discount factors for collection (χ) and distribution (δ) are considered. The objective is to minimize the total transportation costs. Equation (2.19) defines the the total number of hubs to be opened as p. Equations (2.20) and (2.21) impose that each node is allocated to only one hub and equation (2.21) alone prevents allocations to non-hub nodes. Equations (2.22) are the divergence equations for traffic flow emanating from node i to hub k. The domain constraints are given by (2.23) and (2.24).

In 1998, Sohn and Park [70] tried to reduce the size of the model by reviewing the formulation provided by Skorin-Kapov et al. [68], not by reducing the number indices in the variables but by applying symmetry to the unit flow cost reducing the number of variables and constraints by more than a half.

In the same year, Ernst and Krishnamoorthy [34] proposed a formulation for the *Multiple-Allocation p-Hub Median Problem* based on the ideas of their previous work for the *Single-Allocation* version of the problem and in 1999, the same authors [35] propose new formulations for the *Capacitaded Single-Allocation Hub Location Problem* (CSAHLP). In this work, the formulations are the same as in $USApHMP-N$ but constraint (2.19), which define the number of hubs to be opened as p is removed. The fixed setup costs for the hubs are added to the objective function ($\sum_{k\in N} F_kX_{kk}$) and capacity constraints limiting to Γ_k the incoming flow to each hub $k \in N$ are added. These constrains are defined as follows:

$$\sum_{i \ inN} O_iX_{ik} \le \Gamma_kX_{kk}, \quad \text{for all } i,k \in N. \tag{2.25}$$

The new model is called $CSAHLP-N$ by the authors. In 2010, Correia et al. [16] noticed that these formulations are incomplete and may not give a precise description of the set of feasible solutions to the problem because variables Y^i_{kl} are allowed to be different than zero when variables X_{ik} are zero. The following set of constraints was proposed:

$$\sum_{l \ inN} Y^i_{kl} \le O_iX_{ik}, \quad i,k \in N. \tag{2.26}$$

The formulations for hub location problems have been enhanced and developed through the years, sometimes coupled with heuristic algorithms with the aim of obtaining better results and reducing computational times. Some of that work is mentioned in the next section.

2.2 Heuristics for Hub Location and Hub Median Problems

Heuristic procedures have been proposed in order to help solving hub location problems by obtaining upper or lower bounds as a starting points for exact algorithms or simply to find good feasible solutions for problems that are too hard to solve to optimality.

Along with the first mathematical formulation for the *Single Allocation p-Hub Median Problem*, O'Kelly [55] proposed two heuristics. These heuristics consider exhaustively all possible combinations for the p hubs to be located in the network. In one case, the allocation of the spokes is done to the nearest hub (HEUR1); in the other case, the allocation of the spokes is done to its first or second nearest hub (HEUR2). Computational tests were performed using the CAB data set and instances with up to 25 nodes were solved. The results showed that HEUR2 gives better results but, as the discount factor on inter-hub connections decreases, the difference between the two heuristics becomes smaller.

In 1992, Klincewicz [42] developed two heuristics for the *p-Hub Location Problem* based on Tabu Search (TS) and Greedy Randomized Adaptive Search Procedure (GRASP). These approaches were able to solve instances with up to 52 nodes.

Other authors have used TS obtain feasible solutions to hub location problems. This is the case with Abdinnour-Helm [1], who proposed an hybrid algorithm (GATS) based on Genetic Algorithms (GA) and Tabu Search (TS) to solve the *Uncapacitated Single-Allocation Hub Location Problem* (USAHLP). In this heuristic, a GA is used to determine the number and location of the hubs as well as the allocation of the spokes to the hubs. Afterwards, TS is used improve the allocation of spokes to hubs. This heuristic outperformed a previous known GA heuristic for this problem proposed by Abdinnour-Helm and Venkataramanan [2].

Other examples of the use of TS include the work by Marianov et al. [50], where the problem of locating hubs in a competitive environment is investigated. In this type of problem, the possibility of relocating hubs in an existing network with competitor hubs is studied. For instance, in air passenger transportation systems, when the location of a new hub results in the decrease of the cost for a costumer to go from its origin to its destination, the customer is captured by the new hubs instead of using a competitor hub. The objective in such a problem is to capture as much costumer traffic from the competitor as possible. Marianov et al. [50] proposed a formulation for this problem (which would be later revisited by Wagner [76]) and a heuristic procedure based in a one-opt heuristic modified with some TS. In 2009, the same problem was addressed by Eiselt and Marianov [31] who extended the formulation proposed by Marianov et al. [50]. Also, a heuristic concentration method[2] is proposed where, in first phase, the number of candidate locations for the hubs is reduced from the original n to some (much) smaller number by turning, one by one, the hub nodes to non-hub nodes and, in the second phase, instead of moving one hub at a time, two hubs are moved at a time from their current location to any pair of unused locations.

In 2001, Pamuk and Sepil [57] presented the first heuristic for the *Single Allocation p-Hub Center Problem* [3]. A single-exchange heuristic procedure was proposed where the moves from solution to solution are done by replacing a hub node by a non-hub node. This procedure is coupled with TS in order to avoid getting trapped too early in a "valley" associated with a local optimum.

Marianov and Serra [49] propose a heuristic based on TS for the problem of locating airline hubs with the incoming flow to the hubs behaving according to M/D/c queues. In this work, the problem is modeled considering the congestion effects that arise in hub airports due to the higher

[2]A two-phase metaheuristic combining exact and heuristic methods first proposed by Rosing and ReVelle [63] and applied to the p-median problem by Rosing et al. [64] .

traffic flows. Among other reasons, the probabilistic nature of this model makes it very complex and hard to solve. Hence, the authors propose a heuristic procedure where a greedy ADD heuristic is used to find the initial set of p locations to establish as hubs and a one-opt exchange heuristic coupled with TS is then used to improve the initial solution.

A different type of problem was addressed by Yaman and Carello [78] in 2005. The authors consider a hub location problem where the cost of using an edge is stepwise and the traffic transiting through the hub has some limit, as often observed in the design of telecommunication networks. The heuristic presented is based on a local search approach where the problem is decomposed into two subproblems: the location subproblem and the allocation subproblem. TS is applied to the location subproblem part in order to find the best set of hubs and then, the allocation is done using a greedy algorithm.

In 2007, Chen [11] proposed a hybrid heuristic (SATLUHLP) for the *Capacitated Single Allocation Hub Location Problem* (CSAHLP) combining Simulated Annealing (SA) and TS in order to avoid returning to previous visited solutions. This heuristic outperformed the results obtained by Topcuoglu et al. [75] where a GA based heuristic and a SA heuristic has been proposed to solve the same problem.

Later, in 2009, Silva and Cunha [66] also propose heuristic methods based on TS in order to solve the CSAHLP. The authors propose three variants of a simple and efficient multi-start TS heuristic in order to explore several different initial solutions. Also, a two-stage integrated TS heuristic is proposed to improve the location and the allocation components of a solution to the problem.

One of the assumptions often done in hub location problems is that the hub level network is complete. This is not the case in the work presented by Calık et al. [7]. Not only do the authors not only introduce a tabu-based heuristic for the *Single Allocation Hub Covering Problem Over Incomplete Hub Networks* but also they propose an integer programming formulation for it.

Besides presenting the formulation for the USApHMP (this formulation is in the previous section, in 9), in 1996, Ernst and Krishnamoorthy [33] also proposed a heuristic algorithm based on SA in order to use the results in a LP-based branch-and-bound solution method. Later, in 1999, the same authors also proposed two heuristic algorithms for the CSAHLP [35] based on SA and random descent (RD) also with the same purpose. In 2008, Chen [12] presents a heuristic (SATLCHLP) based in TS and SA for the same problem.

Kratica el al. [44] proposed two genetic algorithms (GAHUB1 and GAHUB2) in order to solve the USApHMP. The authors used specific representation and modified genetic operators in order to keep the feasibility of individuals. Instances with up to 200 nodes and 20 hubs were tackled. In the same year, Cunha and Silva [18] proposed an algorithm based on GA for the USAHLP. They did not consider the discount factor on inter-hub connections as a constant but as a function of the total flow between each hub (the greater the flow, the smaller the discount factor). In this algorithm, GA is combined with local search heuristics in each iteration. GA is used to determine the location of the hubs and the initial allocation of each spoke to the hubs. Then, local search is applied in order to improve the allocation of the nodes to the hubs[3].

The above mentioned papers are only some proposing the use of GA based procedures for HLPs. Nevertheless, other authors have proposed this type of approaches within the context of hub location. This is the case with Takano And Arai [73], Kratica et al. [43, 45], Stanimirović [71], Lin et al. [46], Lüer-Villagra [48].

In 2011, Mohamadi et al. [52], proposed an algorithm to solve a *Hub Covering Location Problem (With Crowdness)* with stochasticity[4] In this work, hubs, which are the most crowded parts of

[3]This combination of genetic algorithms with local search heuristics is sometimes referred to as hybrid genetic algorithm or memetic algorithm.

[4]Constraints and random variables are considered such as the transportation time and rate of arrived trucks.

network, are looked at as M/M/c queuing systems. The algorithm is based on GA and Imperialist Competitive Algorithm (ICA)[5].

In 1996, Campbell [9] proposed a greedy heuristic for the SApHMP where the hubs are added according to their contribution to the transportation cost in a non decreasing way. After the solution is constructed, a hub interchanges location with a non hub repeatedly while the result is a decrease in the transportation cost.

In 2005, Bollapragada et al. [5] studied a fixed-wireless network-planning problem with a two phase planning horizon and a different budget for each phase. There are different hub types (with regard to cost and capacity) and demand is stochastic. An optimization model was proposed for the problem as well as a greedy algorithm in order to maximize the expected demand covered. The greedy heuristic chooses the hub locations in each phase according to the ratio of increase in flow divided by the cost of achieving this increase.

In 2014, Peiró et al. [58] presented a GRASP algorithm for an extension of the pHMP where every node is allocated to r of the p selected hubs ($r \leq p$). The problem was initially introduced by Yaman [77]. In the algorithm presented by Peiró et al. [58], first the hubs are selected, then the nodes are allocated to the hubs and finally the traffic is routed from origin to destination through the selected hubs. Lists of candidate solution components for each phase of the algorithm are created and ordered in a greedy fashion and then, the solution components are randomly chosen from the lists until a feasible solution is produced.

Smith et al. [69], in 1996, solved the USApHMP by mapping the problem onto a modified[6] Hopfield neural network. The Hopfield neural network is a mathematical model of the brain which can be implemented in electronic hardware and can be used to solve optimization problems with binary variables. In order to use a smaller formulation, the authors used the quadratic integer programming formulation of the problem proposed by O'Kelly[55]. The network was simulated but much faster results would have been achieved if the network was implemented in electronic hardware. The results were compared with the SA heuristic proposed by Ernst and Krishnamoorthy [33] and a commercial solver package designed to minimize convex functions (note that the used formulation has a non-convex objective function). The authors found that the Hopfield neural network approach is able to compete effectively with the SA and mentioned that the performance of the solver was inferior to both approaches.

In the same year, Skorin-Kapov et al. [68], as already stated before, proposed tighter LP formulations for the pHMP that resulted in tighter linear relaxations with integer solutions in almost all instances. In this work, the authors also developed a linkage between optimal and heuristic solutions in order to solve the problems in which the linear relaxation solutions were fractional. The best known heuristic solutions found by Skorin-Kapov and Skorin-Kapov [67], in conjunction with the solutions to the LP relaxation, were used as guidance for branching strategy and lead to the optimal solutions of the problem.

Pirkul and Schilling [59] applied Lagrangean Relaxation to the tight LP introduced by Skorin-Kapov et al. [68]. Three sets of constraints were relaxed and the main problem was restructured into two sub-problems. This approach led to very small gaps and short computation times.

In 1999, Sasaki et al. [65] considered the *1-stop p-Hub Median Problem*, a special case of the pHMP where each route in the network is allowed to use only one hub. The authors presented a formulation for this problem and proposed an implicit enumeration branch-and-bound algorithm

[5]"Imperialist Competitive Algorithm (ICA) uses socio-political evolution of human as a source of inspiration for developing a strong optimization strategy. Imperialism is the policy of extending the power and rule of a government beyond its own boundaries. A country may attempt to dominate others by direct rule or by less obvious means such as a control of markets for goods or raw materials." (Mohammadi et al. [52])

[6]The unmodified version of the network often ended in the first local optimum and so a modification was made in order to escape from local optima.

that uses Lagrangean Relaxation by dualizing the constraint on the number of hubs. Computational results show in that their algorithm, the CPU time increases rapidly with the size of the problem and, for this reason, a greedy-type approach (that can be seen as a generalization of the greedy heuristic proposed by Campbell [9]) is suggested.

Other researchers have also proposed Lagrangean Relaxation based algorithms such as Contreras et al. [14, 15] to obtain tight upper and lower bounds for the CSAHLP, solving instances with up to 200 nodes, and Lu and Ting [47] for the CSApHMP.

Rodríguez-Martín and Salazar-González [61], in 2006, proposed a heuristic for the *Capacitated Hub Problem* (CHP) in a telecommunications network with capacity constraints on the hubs and arcs and where the graph connecting the hubs is not assumed to be complete. The proposed heuristic makes use of a linear programming relaxation in an Iterated Local Search scheme where the aim is to create a random walk in the space of local optima defined by the output of a local search procedure. The computational experiments on random instances show that the heuristic method provides good feasible solutions for large CHP instances in a reasonable amount of time. Two years later, these authors proposed a LP-based heuristic using a local search based on Variable Neighbourhood Search (Rodríguez-Martín and Salazar-González [62]). The Variable Neighbourhood Search (VNS) method consists of starting from a given solution and applying iteratively two procedures (the neighbourhood is changed in these two procedures): a *shaking* procedure with the intent of getting a good initial solution, and a *local search* in order to improve the initial solution until a local optimum is reached. This process is repeated until some stopping criterion is met. The heuristic proposed gave better results when applying only one operator for performing the *shaking* procedure (simultaneously closing an open hub and opening a closed one) and also only one operator for performing the *local search* procedure (closing a hub opened in the current solution).

Ilić et al. [41] proposed two General VNS (GVNS) heuristics based on three neighbourhood structures (*Allocate, Alternate and Locate*) for the USApHMP. The paper presents results in which instances with up to 400 nodes are solved in less computational time when compared with some existing algorithms.

In 2009, Ernst et al. [32] developed new mathematical formulations for the *Uncapacitated p-Hub Center Problems* with either *Single-Allocation* or *Multiple-Allocation*. They also proposed a shortest path based branch-and-bound algorithm to solve the multiple allocation case and the numerical results showed that the proposed method is extremely efficient for solving the problem. Instances with up to 200 nodes are solved.

Filipović [37], proposed an electromagnetism (EM) meta-heuristic for the *Uncapacitated Multiple-Allocation Hub Location Problem* (UMAHLP). The EM method is a population based algorithm in which a charge is associated to each electromagnetic point (potential solutions to the problem) in the solution set. The charge for each point is calculated as a function of its objective function and all the other points objective functions. The algorithm is based in an attraction-repulsion mechanism and so, as in electromagnetic interaction, all points interact with each other and the interaction depends on the charge that each point has. The interactions can be described by Coulombs Law meaning that the power of connection between two points will be proportional to the product of charges and reciprocal to the distance between them (points with a higher charge will attract more strongly and the best EM point will stay unchanged). This interaction between particles results in moving the particles towards the optimum solution. Instances up with to 200 nodes were solved and the optimum or a best known solution was reached in 27 of the 28 instances tested.

In 2008, Randall [60] proposed an Ant Colony Optimization (ACO) algorithm to solve the CSAHLP. In this paper, four variations of the the Ant Colony System (ACS) technique[7] are developed. ACS is a construction meta-heuristic and the difference between these four variations is

[7]A population based technique designed to simulate the ability of ant colonies to determine shortest paths to food.

in the way the ant colonies construct the solutions. Prepossessing is also used in order to reduce the search space. After each iteration, a local search procedure consisting of six local search operators is applied, in order to improve the constructed solution. The results reveal that each of the approaches can return optimal solutions very fast. Instances with up to 50 nodes were solved to optimality.

In 2009, Meyer et al. [51] also propose an algorithm that uses ACO to solve the *Single Allocation p-Hub Center Problem.* In this paper, a hybrid 2-phase algorithm is introduced where, in the first phase, a heuristic algorithm based in ACO is used to get a good upper bound for a shortest path based branch-and-bound where a set of optimal hub combinations is computed and in the second phase, allocation is done using a reduced size formulation. A purely based ACO algorithm was also proposed where two different ant colonies were used: one to find the hub locations and the other to perform the spoke allocations to each hub. Preprocessing is also used based on a upper bound applied to the path length between any O-D pair. Instances with up to 400 nodes and 5 hubs were solved to optimality by the ACO algorithm alone and the two-phase algorithm.

2.3 Conclusions

As shown in this chapter, many heuristic approaches have been proposed for hub location problems. Concerning genetic algorithms, there are several studies in the literature describing how to encode solutions and apply the crossover and mutation operators. However, this is done mainly for uncapacitated problems and thus, this heuristic approach was not considered as a candidate for tackling the MP-CSAHLP because the encoding and crossover operator might not preserve the best features of the parents to the offspring.

Also, as referred in Meyer et al. [51], it can be very difficult to move from a good solution to a better one using a local search mechanism, due to the solution structure and the local search neighbourhoods defined.

These aspects motivate the work to be done using a construction mechanism, namely the the ACO algorithm.

Recently, Contreras et al. [13] presented an overview on *Hub Location* that reviews not only some of the topics presented in this Chapter but also focuses on other types of problems not addressed here and on exact methods for solving this class of problems.

CHAPTER 3

The Multi-Product Capacitated Single Allocation Hub Location Problem

In this chapter, the work presented by Correia et al. [17] is revisited. The *Multi-product Capacitated Single-Allocation Hub Location Problem* (MP-CSAHLP) is defined and the model used for evaluating the quality of the heuristic approach proposed in chapter 5 is presented.

The MP-CSAHLP is an extension of the CSAHLP where the main difference is that different types of commodities (products) are to be shipped between the O-D pairs. The problem was first introduced by Correia et al. [17], who assume that no more than one hub can be located in each node and the hubs can be either *dedicated* (each hub can only handle one type of product) or *non-dedicated* (one hub can handle more than one type product). Capacities on the hubs are considered with regard to the non-processed incoming flow to the hubs. The hub network is assumed to be complete for each product but, when considering the hub level network as a whole, that might not be the case. The objective is to minimize the total cost, which includes the flow routing cost, the setup costs of the hubs, and the fixed costs of preparing the hubs to handle the different types of products.

Correia et al. [17] proposed a unified modeling framework for this problem, based on existing formulations for the CSAHLP, covering simultaneously both the variations presented (*dedicated hubs* and *non-dedicated hubs*). Also, some enhancements are proposed in order to reduce the gap of the linear relaxation bound.

The remainder of the chapter is organized as follows. In the next section, the formulations for the MP-CSAHLP are presented; in section 3.2 the model enhancements are presented and in section 3.3 some computational results are discussed and conclusions are presented.

3.1 Proposed Models

Before presenting formulations for the MP-CSAHLP, let us first summarize some of the special features introduced earlier and also introduce some notation.

Regarding the special features of the problem, as defined by Correia et al. [17], we have the following:

1. There are several products (flow categories/types) to be shipped through the network.

2. For each node (potential hub) there is a limit on the number of product types that can be handled by a hub located at that node. By doing so, a unified modeling framework can be proposed which has as particular cases the situations in which:

   i) all hubs are dedicated and

   ii) all hubs handle all products.

   All the cases in between these two extremes are possible.

3. Single-allocation is considered for each product. Accordingly, for each product, each node must be allocated to one and only one hub, but a node can be allocated to different hubs for different products.

4. Hubs are capacitated that is, in each hub a limit is considered for the flow of each product that can be handled by the hub.

5. For each product, all the flow must be collected, transferred and distributed using only hubs that handle the product.

6. For each product, the hub level network is a clique (complete graph).

The following notation, as introduced by Correia et al. [17] is now presented:

N	Set of nodes;
P	Set of products;
d_{ij}	Distance between nodes i and j $(i, j \in N)$;
w_{ij}^p	Flow of product p to be sent from node i to node j $(i, j \in N,\ p \in P)$;
$O_i^p = \sum_{j\in N} w_{ij}^p$	Total flow of product p originated at node i $(i \in N,\ p \in P)$;
$D_i^p = \sum_{j\in N} w_{ji}^p$	Total flow of product p destined to node i $(i \in N,\ p \in P)$.

The distance matrix $[d_{ij}]_{i,j\in N}$ is assumed to be symmetric and also, it is assumed that the distance from any node to itself is zero, $d_{ii} = 0$ $(i \in N)$. This assumption does not imply that for each product $p \in P$, the flow matrix $[w_{ij}^p]_{i,j\in N}$ will have a null diagonal. Additionally, it is assumed that the distances satisfy the triangle inequality. Accordingly, since the hub level structure for each product is a clique, all the flow between every O-D pair traverses at most two hubs.

Correia et al. [17] also define:

Γ_k^p	Capacity of a hub installed at node k for handling product p $(k \in N,\ p \in P)$
L_k	Maximum number of product types that can be handled by a hub located at k $(k \in N)$

Regarding the costs the same authors consider:

χ^p	Cost per unit of flow of product p and per unit of distance between a non-hub node and a hub $(p \in P)$;
δ^p	Cost per unit of flow of product p and per unit of distance between a hub and a non-hub node $(p \in P)$;

α^p	Cost per unit of flow of product p and per unit of distance between hubs ($p \in P$);
c^p_{ijkl}	Total cost for sending one unit of flow of product p from node i to node j through hubs k and l. This concerns the flow following the path $i \to k \to l \to j$. We have $c^p_{ijkl} = \chi^p d_{ik} + \alpha^p d_{kl} + \delta^p d_{lj}$, ($i, j, k, l \in N$, $p \in P$);
g_k	Fixed cost for installing a hub at node k independently from the products that the hub will handle ($k \in N$);
f^p_k	Fixed cost for preparing a hub at node k for handling product p ($k \in N$, $p \in P$).

The decisions to be made are:

- which nodes should become hubs;
- which products should be handled by each hub;
- and the flow that will be routed through the hubs for each O-D pair and for each product.

The following decision variables were proposed by Correia et al. [17]:

$$x^p_{ik} = \begin{cases} 1, & \text{if the flow of product } p \text{ originated at } i \text{ is handled by hub } k, \\ 0, & \text{otherwise.} \end{cases} \quad (i, k \in N,\ p \in P)$$

$x^p_{kk} = 1$ indicates that node k is a hub and handles product p ($k \in N$, $p \in P$).

$$z_k = \begin{cases} 1, & \text{if a hub is installed at node } k, \\ 0, & \text{otherwise.} \end{cases} \quad (k \in N)$$

y^p_{ijkl} = Fraction of the flow of product p originated at node i and destined to node j which is routed via hubs k and l by this order ($i, j, k, l \in N$, $p \in P$)

As seen in the previous chapter, several different formulations have been proposed for hub location problems. As mentioned in the previous chapter, Campbell [8] proposed the first linear integer programming formulation for the USAHLP. Based on that formulation and using the decision variables presented, the first proposal for the extended problem emerges as follows:

P_C

$$\textit{minimize} \sum_{p\in P}\sum_{i\in N}\sum_{k\in N}\sum_{l\in N}\sum_{j\in N} w^p_{ij} c^p_{ijkl} y^p_{ijkl} + \sum_{k\in N} g_k z_k + \sum_{p\in P}\sum_{k\in N} f^p_k x^p_{kk}, \tag{3.1}$$

$$\tag{3.2}$$

$$\textit{subject to} \quad \sum_{k\in N}\sum_{l\in N} y^p_{ijkl} = 1, \qquad i, j \in N,\ p \in P, \tag{3.3}$$

$$\sum_{j\in J}\sum_{l\in N} (w^p_{ij} y^p_{ijkl} + w^p_{ji} y^p_{jilk}) = (O^p_i + D^p_i) x^p_{ik}\,, \qquad i, k \in N,\ p \in P, \tag{3.4}$$

$$x^p_{ik} \le x^p_{kk}\,, \qquad i, k \in N,\ p \in P, \tag{3.5}$$

$$\sum_{i\in N} O^p_i x^p_{ik} \le \Gamma^p_k x^p_{kk}\,, \qquad k \in N,\ p \in P, \tag{3.6}$$

$$\sum_{p\in P} x^p_{kk} \le L_k z_k\,, \qquad k \in N, \tag{3.7}$$

$$x_{ik}^p \in \{0,1\}, \qquad i,k \in N,\ p \in P, \tag{3.8}$$
$$y_{ijkl}^p \geq 0, \qquad i,j,k,l \in N,\ p \in P, \tag{3.9}$$
$$z_k \in \{0,1\}, \qquad k \in N, \tag{3.10}$$

The objective function (3.1) aims to minimize the total cost which is expressed as the sum of the flow routing costs, the cost of opening the hubs and the cost of dedicating the hubs to the products. Constraints (3.3) assure that all the flow between each O-D pair, for each product is shipped. Constraints (3.4) determine that if a node is allocated to hub k for product p, all the flow of product p originated and destined to node i must go through that hub. Constraints (3.5) ensure that each node i is allocated to hub k for product p if the hub is open and handles product p. Constraints (3.6) are the capacity constraints limiting the incoming flow to hub k for product p. Constraints (3.7) limit the number of products that each hub can handle. Constraints (3.8)-(3.10) define the domain of the decision variables.

As referred in the previous chapter, Skorin-Kapov et al. [68] proposed an improvement to the formulations presented by Campbell [8] resulting in tighter lower bounds for the linear relaxation of the model. The same can be done for the MP-SAHLP resulting in another formulation in which constraints (3.4) are replaced by

$$\sum_{l \in N} y_{ijkl}^p = x_{ik}^p\,, \qquad i,j,k \in N,\ p \in P, \tag{3.11}$$

and

$$\sum_{k \in N} y_{ijkl}^p = x_{jl}^p\,, \qquad i,j,l \in N,\ p \in P. \tag{3.12}$$

The new model is denoted by (P_{SK}).

Another linearization of the CSAHLP resulted in a three-index formulation proposed by Ernst and Krishnamoorthy [35] thus reducing the size of the formulation. This formulation can be extended to the multi-product version of the problem we are investigating by changing the flow variables and leaving the rest unchanged.

The flow variables can now be defined as:

y_{ikl}^p = Flow of product p originated at i that is routed via hubs k and l by this order $(i,k,l \in N,\ p \in P)$.

The resulting model is then:

P_{EK}

$$\begin{aligned} \textit{minimize} \ & \sum_{p \in P}\sum_{i \in N}\sum_{k \in N} d_{ik}\left(\chi^p O_i^p + \delta^p D_i^p\right) x_{ik}^p + \sum_{p \in P}\sum_{i \in N}\sum_{k \in N}\sum_{l \in N} \alpha^p d_{kl} y_{ikl}^p \\ & + \sum_{k \in N} g_k z_k + \sum_{p \in P}\sum_{k \in N} f_k^p x_{kk}^p\,, \end{aligned} \tag{3.13}$$

$$\textit{subject to} \quad (3.5), (3.6), (3.7), (3.8), (3.10), \tag{3.14}$$

$$\sum_{k \in N} x_{ik}^p = 1, \qquad i \in N,\ p \in P, \tag{3.15}$$

$$\sum_{l \in N} y_{ikl}^p - \sum_{l \in N} y_{ilk}^p = O_i^p x_{ik}^p - \sum_{j \in N} w_{ij}^p x_{jk}^p\,, \qquad i,k \in N,\ p \in P, \tag{3.16}$$

$$\sum_{l\in N} y^p_{ikl} \le O^p_i x^p_{ik}\,, \qquad i,k\in N,\ p\in P, \tag{3.17}$$

$$y^p_{ikl} \ge 0, \qquad i,k,l\in N,\ p\in P. \tag{3.18}$$

In this model, constraints (3.15) define the single allocation for each product; constraints (3.16) are the divergence equations for commodities (i,p) at node k which refer to flow conservation. Constraints (3.17) only allow flow to go from node i to hub k if node i is allocated to hub k (more precisely, flow variables y^p_{ikl} can only be greater than zero if x^p_{ik} is different from zero and the flow sent through hub k is the same as the flow originated at i, O^p_i). Constraints (3.18) define the domain of variables y^p_{ikl}. The objective function (3.13), to be minimized, quantifies the total cost.

It should be noted that constraints (3.17) do not appear in the original formulation presented by Ernst and Krishnamoorthy [35]. As referred in the previous chapter, these constraints were proposed later, by Correia et al. [16] and their presence in the model give a correct description of the set of feasible solutions to the problem.

The three models presented can be reduced to the simple case in which only one type of product is to be shipped through the network by making $|P| = 1$, thus becoming formulations for CSAHLP. Also, if $L_k = 1, k \in N$, the models refer to the case where all hubs are dedicated. In the other extreme, i.e., if $L_k = |P|, k \in N$, all hubs can handle all products.

The importance of using a multi-product formulation, as opposed to solving separately single-product problems and then merging the solutions, resides in the fact that, as demonstrated by Correia et al. [16] with a small example, the merged solutions lead to sub-optimal results.

3.2 Enhancements

With the goal of improving the polyhedral description of the feasibility set, Correia et al. [16] proposed some enhancements to the models P_C, P_{SK} and P_{EK} are suggested by the authors. In this work, as will be explained in more detail in chapter 5, the values of variables x^p_{ik} in the linear relaxation are used as an input to the heuristic algorithm proposed. Only one of the enhancements proposed is added to the formulation used to solve the linear relaxation. Such enhancement is presented below.

Constraints on the Number of Facilities to Use

The goal is to find a lower bound on the number of hubs required for handling each product (as usually done in capacitated facility location problems). Denote such bound by R^p, $p \in P$. The following inequalities can be established:

$$\sum_{k\in N} x^p_{kk} \ge R^p, \qquad p\in P \tag{3.19}$$

$$\sum_{k\in N} z_k \ge max_{p\in P}\{R^p\} \tag{3.20}$$

Each value R^p $(p \in P)$ can be obtained as follows. Let $\Gamma^p_{(1)}, \Gamma^p_{(2)}, \ldots, \Gamma^p_{(n)}$ denote the capacities $\Gamma^p_1, \Gamma^p_2, \ldots, \Gamma^p_n$ sorted in a non-increasing order. R^p is the value that satisfies

$$\sum_{l=1}^{R^p-1} \Gamma^p_{(l)} < \sum_{i\in N} O^p_i \le \sum_{l=1}^{R^p} \Gamma^p_{(l)}, \qquad p\in P \tag{3.21}$$

This means that the lower bound on the number of hubs, for each product p, is obtained as the number of hubs whose summed capacities (in a non-increasing order) is above the total flow originated at the nodes.

In the particular situation in which all hubs handle the same number of products, ($L_k = L, \quad \forall k \in N$), a lower bound for $\sum_{k \in N} z_k$ is $\left\lceil \frac{\sum_{p \in P} R^p}{L} \right\rceil$[1]. Accordingly, in this case, it is possible to do better than (3.20), namely:

$$\sum_{k \in N} z_k \geq \max \left\{ max_{p \in P}\{R^p\}, \left\lceil \frac{\sum_{p \in P} R^p}{L} \right\rceil \right\} \tag{3.22}$$

3.3 A brief summary of Results and Conclusions

In this section we review some of the results and conclusions presented by Correia et al. [16] regarding the work above revisited.

Correia et al. [16] generated instances for the MP-CSACHLP using the AP (Australian Post) data set first introduced by Ernst and Krishnamoorthy [33]. Instances with 10, 20, 25 and 40 nodes were adapted in order to obtain data for different types of products (the reader should refer to please refer to Correia et al. [16] for further details). Instances were generated considering up to three types of products and the maximum number of products handled by each hub was defined to be the same for all hubs, with values ranging from 1 to the maximum number of products considered in each instance.

The computational results presented by Correia et al. [16] show that the model P_{EK} and its enhancements is superior to the other models. With respect to the linear relaxation gap, again, model P_{EK} shows better performance than the other models. Other results led the authors to conclude that models P_C, and P_{SK} and their improvements are not competitive when compared with model P_{EK} and its enhancements.

After a deeper analysis to model P_{EK} and its enhancements, Correia et al. [16] concluded that:

- as expected, the time elapsed until the optimal solution is reached increases with the size of the instances;
- interestingly, in terms of the linear relaxation gaps, the hardest instances are the ones with 25 nodes;
- the instances with fully non-dedicated hubs ($|P| = 1$ with $L_k = 1$, $|P| = 2$ with $L_k = 2$, and $|P| = 3$ with $L_k = 3$) are easier to solve than the others. Furthermore, the instances with fully dedicated hubs ($|P| = 2$ with $L_k = 1$, $|P| = 3$ with $L_k = 1$ are the hardest to solve, i.e. , the more one goes towards dedicated hubs, the more difficult the problem;
- the enhancements proposed lead to more instances being solved and a better lower bound provided by linear relaxation being obtained. Note that among all enhancements proposed Correia et al. [16], the inequalities above presented, (3.19) and (3.22) are extremely easy to derive).

In order to get even a better understanding of the problem (by studying how some characteristics of the solutions change with the different instances considered), the solutions obtained[2] were also analyzed with focus on the optimal cost and the optimal number of hubs located. A simple graphical

[1] $\lceil a \rceil$ denotes the smallest integer greatest or equal than a.

[2]This discussion is restricted to the model which globally behaved better among all the models tested: P_{EK} + (3.19) + (3.22).

analysis lead the authors to conclude that the the optimal cost decreases as we move from instances in which all hubs are dedicated to instances in which all hubs can handle all products and that the gain is higher in larger instances (40 nodes). Also, regarding the number of hubs, the authors conclude that the number of hubs decreases when moving from instances with dedicated hubs towards instances with non-dedicated hubs. These conclusions show that the presence of non dedicated hubs in the network allow different types of products to share hubs, leading to a decrease in the number of hubs needed to consolidate the flows and consequently to a smaller total cost.

The results presented by Correia et al. [16], led to choose as the best performing model P_{EK} + (3.19) + (3.22). This will be the model used later in this thesis for evaluating the quality of the heuristic approach proposed in chapter 5. Also, the same model is used when computing the linear relaxation solutions to be used as input in the algorithm and when solving the traffic routing problem defining the variables x_{ik}^p and z_k as fixed using the heuristic solution values, as later will be described in chapter 5.

CHAPTER 4

Ant Colony Optimization

Ant Colony Optimization is a meta-heuristic that is inspired on the foraging[1] behaviour of ants and on stigmergic communication.

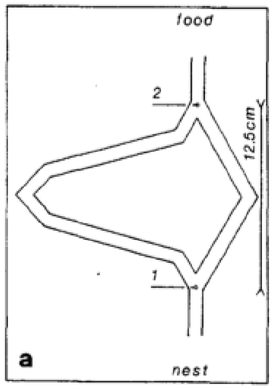
In this chapter, the ant Colony Optimization meta-heuristic is presented and some ACO based algorithms and ACO features are described.

Stigmergic communication is a form of indirect communication based on alterations on the environment that large colonies of insects use in order to achieve self organization. In the case of ants, stigmergic communication is based on chemicals produced by the ants (pheromones) that are deposited on the ground increasing the probability that other ants will follow the same path. Paths that are shorter tend to have higher concentration of pheromone because they are traversed more often, thus, attracting more ants to that path, which, in turn, deposit pheromones, increasing even more the pheromone concentration on those paths. This *positive feedback* mechanism plus the fact that the pheromone evaporates over time (giving origin to a increasingly lower concentration of that chemical on the least traveled paths) is the reason why ants are capable of finding the shortest paths between food and the colony by simply following the most intense pheromone trail.

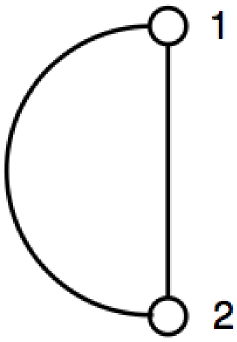
One of the experiments designed to observe the ants behaviour is the double bridge experiment (see Deneubourg et al. [19] and Goss et al. [40]). In this experiment, a double bridge is used to connect a colony of ants to a food source and several trials are conducted in order to examine the ants behaviour when the ratio between the length of the two bridge branches is changed (see Figure 4.1a). When the ratio is set to be 1, the authors observe that, over time, all the ants start to use the same branch. When one of the branches is set to be longer than the other, most of the ants chose the shortest branch. Another variation of this experiment is explored in which the bridge has only one long branch and 30 minutes later the short branch is added. As a result of this variation, the ants were observed to choose the shortest branch only occasionally, leaving the rest of the colony transversing through the longest path. These different results are explained using the concepts of pheromone concentration and pheromone evaporation. In this work, a simple stochastic model is proposed in order to describe the dynamics of the ant colony as observed in the experiment.

In order to simulate the ants behaviour, based on the double bridge experiment, Dorigo and Stützle [29] propose a mathematical model in which artificial ants move through a graph that represents the experimental setup of the double bridge experiment (Figure 4.1b).

[1]foraging: to wander in search of food or provisions.



(a) Double Bridge Experiment Setup.



(b) Double Bridge Experiment Graph.

Figure 4.1: Double Bridge Experiment.

The graph has two nodes (1 represents the food and 2 represents the colony nest) connected by a short, s, and a long, l, arc with an integer ratio r between the two arcs. Time is assumed to be discrete and, at each time step, ants add one unit of pheromone to the arcs they use. When in a node, each ant chooses which arc to travel, with a certain probability $p_{ia}(t)$ which is a function of the pheromone concentration φ_{ia} that the ant in node i ($i \in \{1, 2\}$) encounters on the branch a, ($a \in \{s, l\}$). The resulting discrete time model is based on the average behaviour of the system and is the following:

$$p_{is}(t) = \frac{[\varphi_{is}(t)]^\alpha}{[\varphi_{is}(t)]^\alpha + [\varphi_{il}(t)]^\alpha}, \quad p_{il}(t) = \frac{[\varphi_{il}(t)]^\alpha}{[\varphi_{is}(t)]^\alpha + [\varphi_{il}(t)]^\alpha}. \tag{4.1}$$

The pheromone update on the two branches is performed as follows:

$$\varphi_{is}(t) = \varphi_{is}(t-1)m_i(t-1) + p_{js}(t-1)m_j(t-1), \quad (i = 1, j = 2; i = 2, j = 1) \tag{4.2}$$

$$\varphi_{il}(t) = \varphi_{il}(t-1)m_i(t-1) + p_{jl}(t-r)m_j(t-r), \quad (i = 1, j = 2; i = 2, j = 1) \tag{4.3}$$

where $m_i(t)$ denotes the number of ants at node i in time t and is given by:

$$m_i(t) = p_{js}(t-1)m_j(t-1) + p_{jl}(t-r)m_j(t-r), \quad (i = 1, j = 2; i = 2, j = 1) \tag{4.4}$$

The remainder of this chapter is organized as follows. In the section 4.1, some ACO based algorithms namely Ant System and its extensions are defined in Section 4.2, the effects of coupling ACO with local search are described, and in Section 4.3 some other applications of ACO are also described.

4.1 The ACO Metaheuristic and Optimization Problems

The ACO meta-heuristic has been proposed as a common framework for the existing applications and algorithmic variants of a variety of ant algorithms (Dorigo and Stützle [29]).

The objective of ACO algorithms is to establishing a parallelism between the behaviour of real ants and artificial ants where, in a finite-size colony of artificial ants, each member cooperates with the others using indirect communication trough artificial pheromones, in order to find good-quality solutions for combinatorial optimization problems. Each ant iteratively adds solution components, with probabilities based on the artificial pheromone level of each solution component, until a feasible solution is reached. The amount of pheromone existing in each solution component is changed by each ant and reflects the colony search experience. High-quality solutions are found as an outcome of the global cooperation among all the ants of the colony.

In order relate the shortest paths created by real ants with shortest paths created by artificial ants, a complete graph representing the problem $G_c = (C, L)$ is defined where C is the set of solution components and L is the set of arcs linking the solution components.

The solution components, $c_i \in C$, and the connections between the solution components, $l_{ij} \in L$, can have associated a *pheromone trail* τ (τ_i if associated with components, and τ_{ij} if associated with connections), and a *heuristic visibility information value* η (η_i and η_{ij} respectively). The pheromone trail is a numeric representation of the colony search experience, and is updated by the ants themselves. Differently, the heuristic visibility information value, is an estimate of how good is adding a solution component to the solution under construction and can be computed using information about the problem. In many cases η is the cost, or an estimate of the cost resulting from of adding the component or connection to the solution under construction.

The objective of the colony is to exploit the construction graph $G_c = (C, L)$ in search of the optimal solution of the problem. By performing random walks on the construction graph G_c, ants build solutions by choosing probabilistically the next node to move among those in the neighbourhood of the node in which they are located. The probabilistic choice is biased by the existing pheromone concentration on each arc, resulting from the previously deposited pheromones by other ants. The neighbourhood of each node is defined as the set of available solution components that satisfy the constraints of the problem in a hard way, so that ants always construct feasible solutions, or in a soft way, in which infeasible solutions are penalized as a function of their degree of infeasibility.

The ants can deposit artificial pheromones in the arcs they traversed[2]. The better the quality of the solution, the more pheromone is deposited, hence leading future ants towards better solutions. Also, pheromone evaporation is carried out in order to avoid premature convergence to a local optimum.

Concerning the solution construction, there are two possibilities for implementing it: *parallel* and *sequential*. In the *parallel* implementation, at each construction step all the ants move from their current solution component to the next one, while in the *sequential* implementation an ant builds a complete solution before the next one starts to build another one.

In order to strengthen the solution quality, ACO can be coupled with some *Daemon Actions* that cannot be performed by the ants such as local optimization procedure or the use of collected global information in order to decide if adding additional pheromone is useful or not.

The ACO meta-heuristic, as proposed by Dorigo et al. [23], is presented in Algorithm 4.1.

In this pseudo-algorithm, the activities can be scheduled and synchronized freely by the algorithm designer.

ACO was first applied to the Traveling Salesman Problem (TSP) by Dorigo et al. [26] and can be applied to the problem in a straightforward way as the construction graph $G_c = (C, L)$ is identical to the problem graph. Several different versions of the ACO algorithm have been proposed for the TSP, which are briefly described in the following sub-sections.

[2]The decisions about when the ants should release pheromone and how much pheromone should be deposited depend on the characteristics of the problem and on the design of the implementation. Ants can release pheromone while building the solution, or after a solution has been built. The reader can refer to Dorigo et al. [23] for further details.

Algorithm 4.1 ACOMetaHeuristic()

```
1: while termination_criterion_not_satisfied do
2:     ScheduleActivities
3:         ConstructAntsSolutions
4:         UpdatePheromones
5:         DaemonActions                                  ▷ Optional
6:     EndScheduleActivities
7: end while
```

4.1.1 Ant System

Dorigo et al. [28] developed an ACO based algorithm, Ant System (AS), a "new general-purpose heuristic algorithm which can be used to solve different combinatorial optimization problems", as decribed by the authors. The TSP was used as benchmark due to the easiness in relating the problem structure with the construction graph, as stated above.

The AS algorithm is divided in two main phases: the solution construction and the pheromone update. The pheromone trails are initialized to some value τ_0, a value slightly higher than the expected amount of pheromone deposited by the ants in one iteration, so that the initial tours do not bias the search leading to a poor exploration of the search space[3].

In the construction procedure of the solutions (tours) for the TSP, artificial ants are put on randomly chosen cities. At each construction step, ant k currently at city i, chooses to go to city j with a probability given by:

$$p_{ij}^k = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in \mathcal{N}_i^k} [\tau_{il}]^\alpha [\eta_{il}]^\beta} \ , \quad \text{if } j \in \mathcal{N}_i^k, \tag{4.5}$$

where $\mathcal{N}_i^k$ denotes the set of cities not visited yet (i.e., the feasible neighbourhood of city i). Equation (4.5) is a probabilistic action choice rule, called *random proportional rule*. The probability p_{ij}^k depends on the pheromone concentration, τ_{ij}, and heuristic visibility information, η_{ij}, associated to arc (i, j). The heuristic visibility information is an estimate of how good traveling to j from i is. In the TSP, it can be defined as $\eta_{ij} = 1/d_{ij}$ (where d_{ij} is the distance between cities i and j). This allows an influence on the ants behaviour based on the knowledge of the problem structure.

The parameters α and β are used to define the relative importance of the pheromone concentration and heuristic visibility information balancing exploration and exploitation (see Dorigo et al. [21] for additional details). The larger the value of α, the larger the exploitation of the search experience.

Additionally, each ant k keeps track of the cities already visited, in the order they were visited. This memory is used to define the feasible neighborhood $\mathcal{N}_i^k$ and allows ant k both to compute the length of the tour, T^k, it generated and to retrace the path to deposit pheromone.

At the end of the solution construction performed by all ants, the pheromone values are updated. This process has two steps:

1. the pheromone values on each arc (i, j) are lowered:

$$\tau_{ij} \leftarrow (1 - \rho)\tau_{ij}, \quad \forall (i, j) \in L; \tag{4.6}$$

[3]Conversely, if the initial pheromone values are too high, then many iterations are lost waiting until pheromone evaporation reduces enough pheromone values, so that pheromone added by ants can start to bias the search.

2. pheromone is added to the arcs traversed by the ants:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}. \quad \forall (i,j) \in L \tag{4.7}$$

The evaporation procedure, represented by equation (4.6), is performed using a constant factor ρ where $0 < \rho < 1$ is the pheromone evaporation rate. Pheromone evaporation is done in order to avoid unlimited accumulation of the pheromone on the arcs. This allows the algorithm to escape from the situation where the same solutions are repeatedly chosen, thus avoiding stagnation in a sub-optimal solution and favoring the exploration of other areas in the search space.

In equation (4.7), $\Delta\tau_{ij}^{k}$ is the amount of pheromone that ant k deposits on the arcs and it is defined as:

$$\Delta\tau_{ij}^{k} = \begin{cases} 1/C^{k}, & \text{if arc } (i,j) \text{ belongs to } T^{k}; \\ 0, & \text{otherwise,} \end{cases} \tag{4.8}$$

where C^{k} is computed as the sum of the lengths of the arcs belonging to T^{k} (the tour built by ant k).

4.1.2 Elitist Ant System

One of the improvements done to the AS gave origin to the Elitist Ant System (EAS). This method is called *elitist* because it resembles the elitist strategy used in genetic algorithms (Dorigo et al. [27]), where certain individuals (*elite individuals*) may go directly into the next generation, preserving the best characteristics from past generations.

In the case of the TSP, this improvement provides additional pheromone reinforcement to the arcs belonging to the best tour found since the start of the algorithm (denoted as T^{bs}, the best-so-far tour). By considering this additional feedback, the following ants will have a higher probability of choosing the arcs belonging to the best-so-far tour. This is an example of a *daemon action* of the ACO meta-heuristic where collected global information is used in order to decide if adding additional pheromone is useful or not.

In an elitist ant system, the pheromone update rule (4.7) is modified by adding a quantity $e\Delta\tau_{ij}^{bs}$ to the the best-so-far tour (in the case of the TSP), leading to:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + e\Delta\tau_{ij}^{bs}, \tag{4.9}$$

where e is a parameter that defines the weight given to the best-so-far tour T^{bs}. $\Delta\tau_{ij}^{k}$ is defined as in equation (4.8) and $\Delta\tau_{ij}^{bs}$ is defined as follows:

$$\Delta\tau_{ij}^{bs} = \begin{cases} 1/C^{bs} & \text{if arc}(i,j) \text{ belongs to } T^{bs}; \\ 0 & \text{otherwise,} \end{cases} \tag{4.10}$$

where C^{bs} is the best-so-far tour length. The pheromone evaporation, is defined by equation (4.6) as in an AS.

Computational results presented in Dorigo et al. [27] suggest that, in the case of the TSP, the use of the elitist strategy, with a proper value for parameter e, results in better tours discovered and/or in the best tour being discovered earlier, but, if care is not taken when choosing the parameter value, the algorithm may be forcing the ants to explore sub-optimal tours in the early stage of the search, resulting in a poor performance of the algorithm due to early stagnation.

4.1.3 Other Improvements on Ant System

Some other improved versions of an AS have been suggested where the main difference between them is in the way that the pheromones are updated. One example is the Rank-Based Ant System AS_{rank}, proposed by Bullnheimer et al. [6]. In this approach, each ant deposits an amount of pheromone that decreases with its rank.

Taking the TSP as an example, the ants are sorted by non-decreasing order of their tour length. Only ω ants are considered and the pheromone levels in the arcs belonging to the tours constructed by the $\omega - 1$ best-ranked ants are increased according to the rank of the ants (the r-th best ant of the colony contributes to the pheromone update with a weight given by $max\{0, \omega - r\}$). The pheromone levels of the best-so-far tour are also increased (with weight ω). The pheromone update is then given by:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{r=1}^{\omega-1} (\omega - r)\Delta\tau_{ij}^{r} + \omega\Delta\tau_{ij}^{bs}, \tag{4.11}$$

where $\Delta\tau_{ij}^{r} = 1/C^{r}$ and C^{r} is the length of the tour of the r-th ant. $\Delta\tau_{ij}^{bs}$ is given by equation (4.10).

The results obtained by Bullnheimer et al. [6] suggest that AS_{rank} preforms slightly better than EAS and significantly better than AS in the case of the TSP.

Another example is the $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System ($\mathcal{MM}$AS) introduced by Stützle and Hoos [72]. In this work, only the iteration best ant or the best-so-far ant is allowed to update the pheromone trails but, in order to avoid early stagnation due to excessive growth of pheromone trails, an explicit maximum, τ_{max}, and minimum, τ_{min}, value for the pheromone trails are introduced. The pheromone trails are initiated to τ_{max} and after each iteration, evaporation is applied as before (equation (4.6)). Also, the pheromone trails are increased in all arcs (i, j) using the *trail-smoothing mechanism* (given by equation (4.12)) when no improved tour has been generated for a certain number of consecutive iterations.

$$\tau_{ij} \leftarrow \tau_{max} - \tau_{ij} \tag{4.12}$$

4.1.4 Ant Colony System

The Ant Colony System (ACS) algorithm was first proposed by Dorigo and Gambardella [24, 25] as an extension of AS with the goal of improving efficiency when applied to the TSP. The main differences between ACS and AS are (i) in the action choice rule used for ants to transit from one city to the other, (ii) in the fact that the pheromone update is done to the best-so-far ant or the iteration best ant and (iii) in the way that the pheromone evaporation is done.

Regarding the action choice rule, the ant, when in city i, chooses the next city to visit j using the *pseudo-random proportional rule*, given by

$$j = \begin{cases} argmax_{l \in \mathcal{N}_i^k}\{\tau_{il}[\eta_{il}]^{\beta}\}, & \text{if } q \leq q_0; \\ J, & \text{otherwise.} \end{cases} \tag{4.13}$$

$$J = R\left(\begin{cases} \frac{\tau_{il}[\eta_{il}]^{\beta}}{\sum_{m \in \mathcal{N}_i^k} \tau_{im}[\eta_{im}]^{\beta}}, & \text{if } l \in \mathcal{N}_i^k; \\ 0, & \text{otherwise.} \end{cases} \right) \tag{4.14}$$

where $\mathcal{N}_i^k$ in the feasible neighbourhood of city i, τ_{ij} is the pheromone concentration on arc (i, j) and, η_{ij} is the heuristic visibility information associated to arc (i, j) (an estimate of how good traveling to j from i is). The parameter β is used to define the relative importance of the heuristic visibility information and $R(.)$ is the roulette wheel selection function.

Before applying this rule, a random variable, q, with distribution $U[0,1]$ is generated and compared with a parameter chosen *a priori*, q_0 ($0 \leq q_0 \leq 1$). With probability q_0 the city, j, chosen to be the next solution component is chosen by using a greedy rule (the upper branch of (4.13)), in the sense that the best combination of pheromone and heuristic visibility information is chosen (in this case, the ant is exploiting the learned knowledge). With probability $1 - q_0$, the the selection is made using function $R(.)$ to perform roulette wheel selection so that the ant performs a biased exploration of the arcs. Tuning the parameter q_0 allows choosing the degree of exploration as well as whether to concentrate the search of the system around the best-so-far solution or to explore other tours.

After all tours are constructed by the ants, at the end of each iteration, the amount of pheromone deposited on each arc is increased. The increase is ruled by equation (4.15), the global pheromone update rule.

$$\tau_{ij} \leftarrow (1-\gamma)\tau_{ij} + \gamma\Delta\tau_{ij}^{bs}, \quad \forall(i,j) \in T^{bs}, \tag{4.15}$$

where, again, $\Delta\tau_{ij}^{bs} = 1/C^{bs}$ and the parameter γ represents the global pheromone update parameter ($0 < \gamma < 1$). Here, the deposited pheromone is discounted by a factor γ resulting in the new pheromone trail being a weighted average between the old pheromone value and the amount of pheromone deposited. Note that the pheromone level is increased only by the best-so-far ant (meaning that the pheromone levels are only increased in the best-so-far tours).

Additionally, the ants reduce the amount of pheromone deposited on each arc (i, j) immediately after having crossed it during the tour construction. They apply a local pheromone update rule, which is given by:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\tau_0, \tag{4.16}$$

where the parameter ρ represents the local pheromone evaporation rate (as ρ in equation (4.6)) and τ_0 is set to be the same as the initial value for the pheromone trails mentioned in subsection 4.1.1. As stated by Dorigo and Gambardella [25], the local pheromone updating rule decreases the pheromone levels of the tours already explored, decreasing the probability that the next ants will explore the same paths explored by previous ants, increasing the exploration of arcs that have not been visited yet and avoiding premature convergence to local optima.

When comparing the AS with its extensions, Dorigo and Stützle [29], noted that, for the TSP, “ACS is the most aggressive of the ACO algorithms and returns the best solution quality for very short computation times”.

4.2 ACO coupled with Local Search

In an ACO algorithm, after the solution construction procedure is finished, the solution can then be improved by application of local search. This is an example of the *Daemon Actions* that can be implemented in an ACO algorithm as seen in Algorithm 4.1. After the solution improvement done by local search, the pheromone values on the arcs belonging to the best solution found are updated.

The neighborhood used in the solution construction phase is based in the pheromone values and heuristic visibility information but the neighborhood in which local search is based can be different. This gives more chance to improvements of the solution in the local search phase (see Dorigo and Stützle [29] for additional details).

The algorithm can also by applied without local search, relying only on the heuristic visibility information in order to construct good solutions from the beginning, based on the problem characteristics. Since the heuristic visibility information is often calculated as an estimate of the real value, it might be very hard or impossible to compute. Luckily, using an ACO algorithm incorporating local search may be enough to achieve good results.

4.3 Other applications of ACO

The ACO metaheuristic has a very wide applicability: it can be applied to any combinatorial optimization problem for which a solution construction procedure can be conceived, since it is based on a generic problem representation on a construction graph.

Besides being used for the TSP, some ACO algorithms have been improved and modified in order to be able to be applied to other types of problems (Dorigo et al. [23]), such as, job-shop scheduling, vehicle routing, sequential ordering, graph coloring or quadratic assignment problems (Gambardella et al. [38]). As seen in Chapter 2, ACO can also be applied to hub location problems (Meyer et al. [51], Randall [60]) with the outcome of getting efficient results quickly. ACO has also been used in more complex problems (see Dorigo and Stützle [29]) such as dynamic problems (where the instance data may change while solving the problem), stochastic problems (where some part of the information on the problem is probabilistic) or multiple objective problems (where the objective is to optimize two or more objective functions that can be in conflict with each other).

CHAPTER 5

An Ant Colony System for the The MP-CSAHLP

In this chapter, the proposed heuristic for the MP-CSAHLP is proposed. The proposed heuristic is based in Ant Colony System (ACS), an ACO based algorithm referred in the previous chapter.

The remainder of this chapter is organized as follows. In Section 5.1, we introduce some ideas used in the algorithm, in Section 5.2 the solution representation and data structures are presented, in Section 5.3 algorithm is described and in Section 5.4 the results are presented and discussed.

5.1 Preliminaries

Since ACS is one of the improvements done to AS that often returns the best solution quality within acceptable computational time (Dorigo and Stützle [29]) and has shown to return good quality solutions for the CSAHLP (Randall [60]), it has been chosen as the proposed heuristic for the MP-CSAHLP.

This constructive, population-based algorithm uses a single colony of ants to simultaneously determine the location of the hubs and the allocation of the nodes to the hubs. As a *daemon action*, a local search procedure is applied after each solution is constructed in order to improve the solution quality. Another type of *daemon action* used to improve the quality of the first solution constructed, consists of making the initial pheromone values on each solution component biased by the linear relaxation solution. By doing this, better initial solutions can be obtained before local search is applied. Initial solution components having high pheromone values due to this action, when chosen to be a part of the solution, have their pheromone levels gradually decreased by pheromone evaporation. This mechanism avoids unlimited accumulation of pheromones and the decreased of pheromone concentration on a solution component that very rarely (or never) receives additional pheromone.

Preprocessing can help reducing the search space by making unavailable solution components that will always lead to infeasible solutions. The preprocessing performed in the case of the MP-CSAHLP uses some ideas proposed by Ernst and Krishnamoorthy [35] for the CSAHLP:

1. **Capacity**: If for some i, $j \in N$, $O_i^p > \Gamma_j^p - O_j^p$, then the variable x_{ij}^p can be fixed to zero. In other words, if for a particular node i and its potential hub j the capacity of the hub is insufficient to process the inflow from node i, then i cannot be allocated to j.

2. **Fixed cost**: If for some node i and its potential hub j we have $O_i < \Gamma_i$ and $d_{ij}(\chi^p O_i + \delta D_i) > F_i + \alpha^p d_{ij}(O_i + D_i - 2w_{ii}^p)$, then x_{ij}^p can be fixed at zero. This is because it would be cheaper to make node i a hub and transfer all of the flow to hub node j than to allocate i to j and then pay the collection and distribution costs.

3. **Inference**: ... Similarly, if $x_{jj}^p = 0$ for some j, then $x_{ij}^p = 0$ for all $i \in N$, $i \neq j$. ...

When the location and allocation variables are fixed, routing of the flow can be easily computed, using the formulations P_{EK} for the MP-CSAHLP proposed by Correia et al. [17].

5.2 Solution Representation and Data Structures

A construction graph is designed to represent the problem $G_c = (C, L)$ (where C is the set of solution components and L is the set of arcs linking the solution components, as referred in section 4.1). Instead of representing the solution components according to variables x_{ij}^p, as represented in the problem formulation (section 3.1), we propose that solution components are now represented by x_i^p with

$$x_i^p = \begin{cases} \text{j} & \text{if the flow of product } p \text{ originated at } i \text{ is handled by hub } j, \\ 0 & \text{otherwise.} \end{cases}$$
$$(i, k \in N,\ p \in P)$$

This representation reduces the size of the solution matrix by a factor of n.

The solution components that refer to the opened hubs are unnecessary in the solution representation, because $x_j^p = j$ means that the hub j is open and handles product p and so z_j, as defined in section 3.1, can be inferred using this solution representation.

When moving through the construction graph, the arcs linking the solution components represent the possibility of moving from one solution component to the other one. For that reason, a 3-dimensional vector of binary parameters can be used for identifying feasible moves in graph G_c:

$$a_{ij}^p = \begin{cases} 1 & \text{if the flow of product } p \text{ originated at } i \text{ can be handled by hub } j, \\ 0 & \text{otherwise.} \end{cases}$$
$$(i, k \in N,\ p \in P)$$

Not only can these values be computed and updated during the execution of the algorithm but also they can be computed in advance by preprocessing the data. The 3-dimensional vector, $[a_{ij}^p]_{i,j \in N, p \in P}$ is used to define the feasible neighborhood $\mathcal{N}_{ijp}^k$ of each solution component.

The pheromones, $\tau_i^p(j)$, are deposited on each solution component and are updated using the global pheromone update rule (5.1) to increase the pheromone level on the best iteration solutions.

$$\tau_i^p(j) \leftarrow (1-\gamma)\tau_i^p(j) + \gamma \Delta\tau_i^p(j), \quad \forall (i, j, p) \in S^{best}, \tag{5.1}$$

where, $\Delta\tau_i^p(j) = 1/C^{best}$, C^{best} is the best iteration cost, S^{best} is the set of the solution components belonging to the best iteration solution and and γ is the global pheromone update parameter $(0 < \gamma < 1)$.

Just after adding a solution component to the solution under construction, the local pheromone update rule (5.2) is used to reduce the pheromone level on that component.

$$\tau_i^p(j) \leftarrow (1-\rho)\tau_i^p(j) + \rho T_i^p(j), \tag{5.2}$$

where the parameter ρ is the local pheromone decay parameter ($0 < \rho < 1$) and $T_i^p(j)$ represent the initial pheromone levels in solution component $x_i^p = j$.

The initial pheromone levels $T_i^p(j)$ are biased proportionally by the solution values, $\hat{x}_{ij}^p$, of the linear relaxation of the model P_{EK} introduced by Correia et al. [17] and revisited in Chapter 3 according to the following expression:

$$T_i^p(j) = \begin{cases} \tau_0 + \hat{x}_{ij}^p \times c_\tau, & \text{if } \hat{x}_{ij}^p > 0; \\ \tau_0, & \text{otherwise;} \end{cases} \qquad i, j \in N, \, p \in P \tag{5.3}$$

where τ_0 ($\tau_0 > 0$) is the initial pheromone value and $c_\tau > 0$ is a constant that is used to decide how much importance is given to the solution components that are part of the linear relaxation solution.

Since it is not straightforward to estimate how good is to add a solution component to the solution under construction is not straightforward, heuristic visibility information, η_j^p, is only applied to the hubs in order to influence the choice of which hubs to open and what type of products they should handle. The fixed costs, F_j^p, and an estimate of the total flow cost, K_j^p (Randall [60]) are used as stated below. For computing the total flow cost, each hub is considered a single hub in a star network trough which all flow is directed. These costs are computed as follows:

$$F_j^p = f_j^p + g_j, \quad j \in N, \, p \in P, \tag{5.4}$$

$$K_j^p = \sum_{j=1}^{N} \sum_{l=1}^{N} w_{ij}^p (\chi^p d_{ji} + \delta^p d_{il}), \quad j \in N, \, p \in P. \tag{5.5}$$

The heuristic visibility information values, η_j^p, are computed taking in to consideration the 3-dimensional vector, $[a_{ij}^p]_{i,j \in N, p \in P}$, above introduced.

$$\eta_j^p = \begin{cases} \frac{a_{ij}^p c_\eta}{F_j^p K_j^p}, & \text{if } i = j, \\ a_{ij}^p, & \text{otherwise.} \end{cases} \tag{5.6}$$

In this expression, c_η is a constant used to make the heuristic visibility information computed values larger than 1 and is computed as the sum of the hub fixed costs multiplied by the total flow costs ($c_\eta = \sum_{j \in N} \sum_{p \in P} F_j^p K_j^p$).

5.3 The Algorithm

The main tasks considered while designing the algorithm concern (i) reading the instance data, (ii) initialization of the data structures and parameters, (iii) construction of the solution, (iv) managing the the pheromones, and (v) defining the local search procedure. The ACS algorithm is generically presented in Algorithm 5.1.

After the *ReadData* procedure, the Linear Relaxation of the problem is solved in order to get the values $\hat{x}_{ij}^p$ for computing the initial pheromone levels according to equation(5.3). The data is then submitted to the "capacity" preprocessing procedure already mentioned in order to forbid connections of nodes to hubs that only have enough capacity to handle their own flow. Mathematically, we have

$$a_{ij}^p = \begin{cases} 0, & \text{if } O_i^p > \Gamma_j^p - O_j^p \text{ and } i \neq j \\ 1, & \text{otherwise} \end{cases} \qquad (i, k \in N, \, p \in P) \tag{5.7}$$

Algorithm 5.1 `ACS_MP-CSAHLP()`

```
1:  ReadData
2:  ComputeLR
3:  PreProcessing
4:  InitializeParameters
5:  InitializeDataSructures
6:  while termination_criterion_not_satisfied do                ▷ Iteration Loop
7:      ResetAntStructures
8:      for each ant do                                          ▷ Ant Colony Loop
9:          ConstructAntSolution
10:     end for
11:     ant ← GetBestAnt
12:     if ant not dead then
13:         LocalSearch
14:         GlobalPheromoneUpdate
15:         if solution.cost < best.cost then
16:             GetSolution
17:         end if
18:     end if
19: end while
20: RouteFlows
```

In the *InitializeParameters* function, all the parameters used in the algorithm are initialized such as:

- the number of ants in the colony;
- the parameter $q0$ to be used in the *pseudo-random-proportional rule* ((5.8) and (5.9));
- the parameters α and β to be used in equation (5.9) for computing the probabilities of choosing the solution components when using the roulette wheel;
- ρ to be used in the local pheromone update rule (5.2);
- γ for the global pheromone update rule (5.1).

In the *InitializeDataStructures* function, the heuristic visibility information is computed using expression (5.6), the initial pheromone levels ($T_i^p(j)$) are computed using expression (5.3) and $\tau_i^p(j)$ is set equal to $T_i^p(j)$ for all $i, j \in N$, $p \in P$. Finally, the iteration structures are initialized (such as the iteration best solution, the iteration cost, etc.) . Just before the beginning of the *Ant Colony Loop*, the *ResetAntStructures* function resets all the ant structures (such as the ant solution components, the ants costs, the feasibility set, etc.) in order to start the new solution construction.

In the *ConstructAntSolution* function, each ant constructs a solution, according to Algorithm 5.2.

An ant builds a complete solution before the next one starts to build another one. Thus, the solutions are constructed using a *sequential* solution construction.

Next, each function of Algorithm 5.2 is defined, in order to explain how an ant builds a solution.

The *PseudoRandomProportionalRule* is used to choose a solution component using the following equation:

Algorithm 5.2 *ConstructAntSolution*

```
while nr_available_solutions>0 do
    $(i, j, p) = PseudoRandomProportionalRule$
    if $a^p_{ij} > 0$ and $AvailableCapacity_{jp} > O_{ip}$ then
        choose $(i, j, p)$ as the next solution component
    else
        declare ant as dead and break                ▷ Finish Algorithm 5.2 for this ant.
    end if
    if ant is not dead then
        $AddSolutionComponent$
        $a^p_{ij} = 0$
        $LocalPheromoneUpdate$
        $ApplySingleAllocationRules$
        if i=j then
            $ApplyLkRules$
        end if
        $UpdateAvailableCapacities$
        if $i \neq j$ then
            $DedicateHub$
        end if
    end if
    nr_available_solutions= $ComputeNrAvailableSolutions$
    nr_remaining_solutions= $ComputeNrRemainingSolutions$
    if nr_available_solutions=0 and nr_remaining_solutions>0 then
        declare ant as dead and break                ▷ Finish Algorithm 5.2 for this ant.
    end if
end while
```

$$(i,j,p) = \begin{cases} argmax_{(i,j,p)\in\mathcal{N}^k_{ijp}}\{\tau^p_i(j)[\eta^p_j]^\beta\}, & \text{if } q \leq q_0, \\ (I,J,P), & \text{otherwise.} \end{cases} \tag{5.8}$$

$$(I,J,P) = R\left(\begin{cases} \frac{\tau^p_i(j)[\eta^p_j]^\beta}{\sum_{(r,s,t)\in\mathcal{N}^k_{ijp}} \tau^t_r(s)[\eta^t_s]^\beta}, & \text{if } (i,j,p) \in \mathcal{N}^k_{ijp}, \\ 0, & \text{otherwise.} \end{cases}\right) \tag{5.9}$$

where $\mathcal{N}^k_{ijp}$ is the set of feasible solution components for ant k and is defined as the solution components that have $a^p_{ij} > 0$ and the hub j has enough available capacity to handle the flow originated at node i ($AvailableCapacity^p_j > O^p_i$).

In the *AddSolutionComponent* function, x^p_i is set equal to j, if adding this solution component does not make the solution infeasible. Immediately after, the *LocalPheromoneUpdate* function decreases the pheromone level on that component using the Local Pheromone Update Rule (5.2).

After the solution component is chosen and the pheromone level on that component updated, the feasibility set $\mathcal{N}^k_{ijp}$ is updated by making some solution components unavailable. Since this is is a Single-Allocation problem, if the solution component is referent to the allocation of node i to hub j for product p and $i \neq j$, the *ApplySingleAllocationRules* function does not allow any connections from node i to any other nodes for product p (5.10) and, since node i will not become a hub, all connections from all the other nodes to node i for product p are also forbidden (5.11).

$$a^p_{il} \leftarrow 0, \quad \forall l \in N, \tag{5.10}$$

$$a^p_{li} \leftarrow 0, \quad \forall l \in N. \tag{5.11}$$

On the other hand, if the solution component refers to dedicating hub j for product p ($i = j$), the *ApplySingleAllocationRules* function does not allow any connections from hub j to any other nodes (5.10) (these solution components are only defining the location of the hubs and the allocation of the nodes to the hubs and not the inter-hub connections[1].

Additionally, when the hubs are not allowed to handle all products, if dedicating hub j to product p means that the hub has reached the limit of products it can handle, the solution components that allow the hub to handle more products have to be forbidden also:

$$a^p_{lj} \leftarrow 0, \quad \forall l \in N, \forall p \in \overline{P}, \tag{5.12}$$

where $\overline{P}$ is the set of products that the hub j is not handling, i. e., $\overline{P} = \{p : x^p_j \neq j\}$.

After applying the rules that update the feasibility set ($\mathcal{N}^k_{ijp}$), the *UpdateAvailableCapacities* function is used to update the available capacity of the hub chosen. If $i \neq j$ the hub j is dedicated to product p by applying the function *DedicateHub* according to Algorithm 5.3.

After all ants in the colony have constructed a solution, the best solution produced is chosen and a *LocalSearch* procedure is applied in order to further improve its quality. Three different neighborhoods are defined, as follows:

- **Close Hub**: A hub that is dedicated to product p and only handles its own flow is made a non-hub node and allocated to another hub, chosen randomly, handling the same product. If the hub only handles that product, it is closed;

[1]The *GetSolution* function does that.

Algorithm 5.3 *DedicateHub*

```
1: if $x_i^p \neq j$ and $a_{ij}^p > 0$ then
2:     $x_j^p = j$
3:     $a_{jj}^p = 0$
4:     $ApplySingleAllocationRules$
5:     $ApplyLkRules$
6:     $LocalPheromoneUpdate$
7:     $UpdateAvailableCapacities$
8: end if
```

- **Close Random Hub**: A hub dedicated to product p is chosen randomly and is made non-hub. All the nodes allocated to that hub for product p and the hub itself are allocated to another hub handling the same product. The first node to be reallocated is the ex-hub and it is allocated to the available hub with the higher available capacity. The remainder of the nodes are allocated to the available hubs in a decreasing order of the flow emanating from them to the hubs with the higher available capacity. If the hub only handles that product, it is closed;

- **Relocate Node**: A product is chosen randomly and, in the hub-and-spoke network regarding that product, a node is chosen randomly and allocated to another hub.

These moves are applied only if they produce a feasible solution. The search stops as soon as the change in the solution yields an improvement of the cost.

After applying the local search procedure, the pheromone concentration is updated by making the pheromone levels stronger in the components of the solution constructed ($GlobalPheromoneUpdate$). The global pheromone update rule is given by expression (5.1).

If the solution found is better than the best solution found so far, the *GetSolution* procedure saves the best solution.

At the end, the variables y_{ikl}^p are computed using the P_{EK} formulations presented in section 3.1 with the variables x_{ij}^p and z_j fixed and using the general solver IBM ILOG CPLEX 12.3 in the $RouteFlows$ function.

5.4 Computational Tests

In this section, the computational tests are reported. The algorithm described in the previous section was coded in C++, using Microsoft Visual C++ 2010 Express Edition together with IBM ILOG Concert Technology 2.9 and the experimental tests were run on a Intel(R) Core(TM) i7-4770 CPU, 3.4GHz, 16 GB RAM. The model P_{EK}, proposed by Correia et al. [17], was used to compute the linear relaxation of the problem and to solve the problem to optimality (when possible), in order to have a measure of the effectiveness of the proposed heuristic.

5.4.1 Test Data

In hub location research three data sets are usually considered:

- the CAB data set, introduced for hub location by O'Kelly [53] is based on the airline passenger interactions between 25 US cities in 1970 evaluated by the Civil Aeronautics Board (CAB);

- the AP (Australian Post) data set was introduced by Ernst and Krishnamoorthy [33] and refers to postal (ground) operations between 200 locations in metropolitan Sydney, Australia;

- the Turkey data set, introduced by Tan and Kara [74], refers to ground transportation between 81 cities in Turkey.

The CAB data set cannot be used in capacitated hub location problems because the data set does not include either capacity constraints or fixed costs for the hubs. On the other hand, the AP dataset includes capacity constraints and fixed costs making it suitable to be used for the problem investigated in this work. As mentioned, in Section 3.3, Correia et al. [17] generated test instances for the MP-CSAHLP using the AP data set.

Each instance name has a suffix composed of two letters both chosen in the set $\{T, L\}$, with T standing for tight and L for loose. The first letter on the suffix is associated with the costs. The second letter refers to the tightness of the capacity constrains with T meaning tighter capacity constraints and L meaning loose capacity constraints.

Following the two letters, each instance has two numbers: the first concerning the different types of product flows in the network and the second regards the number of products that each hub can handle. The reader should refer to Correia et al. [17] for further details. Networks with 10, 20, 25 and 40 nodes are considered; 1, 2, and 3 types of products are assumed. The number of products that each hub can handle depends on the number of products considered, ranging from 1 to the maximum number of products considered.

5.4.2 Preliminary Tests

In some preliminary computational tests, the **Fixed cost** preprocessing procedure introduced above was considered. However, analyzing the small decrease in the number of solution components available and the little effect this preprocessing had on the solutions quality, it was decided not to used it. The **Capacity** preprocessing coupled with the **Inference** preprocessing led to a lowering in the number of infeasible solutions found and thus, it was considered in the more extensive tests performed later.

A change in the global pheromone update rule was performed because the solution cost was not converging fast enough to lower values. A scaling parameter SP was added in order to speed up convergence so, the the global pheromone update rule was changed to

$$\tau_i^p(j) \leftarrow (1-\gamma)\tau_i^p(j) + \gamma SP \Delta\tau_i^p(j), \quad \forall (i,j) \in S^{best}, \tag{5.13}$$

Initially, SP was considered as the sum of all fixed costs multiplied by a constant c_{SP}. Nevertheless, this turned out not to be very efficient so it was changed to the solution cost of the linear relaxation again multiplied by a constant c_{SP}.

Another change made is related with the estimated costs to be associated with each solution component. In an early stage, the only costs considered were the fixed costs of opening a hub, g_j, and dedicating a hub, f_j^p, plus the *collection* and *distribution* costs computed as follows:

$$d_{ij}(\chi^p O_i^p + \delta^p D_i^p). \tag{5.14}$$

In addition to these costs, after each ant finishes constructing a solution, an estimate of the inter-hub flow routing costs was added. Constraints (3.17) were taken into consideration for estimating the inter-hub flow routing costs as follows:

$$\sum_{p \in P} \sum_{i \in N} \sum_{k \in N} \alpha^p O_i^p x_{ik}^p \left(\sum_{l \in N} d_{kl} \right) \tag{5.15}$$

After constructing a solution, the inter-hub connections were taken into account in order to add these costs to the costs computed so far.

Instead of using the term ($\sum_{l \in N} d_{kl}$) in the previous expression, the average of all inter-hub distances ($\frac{\sum_{l \in N} d_{kl}}{n_{connections}}$, where $n_{connections}$ is the total number of inter-hub connections) gave origin to better results and was used instead.

5.4.3 Parameter Tuning

After some limited experimentation, several parameters were set. The number of ants, n_{ants}, was set equal to 10; the maximum number of iterations, n_{iter}, was set equal to 5000; the algorithm was set to stop after 180 CPU seconds or if a better solution was not found after $n_{iter}/10$ iterations (stopping criteria). The pheromone update parameters were set to be $\rho = \gamma = 0.5$, and p_0 fixed to 0.1 in the *pseudo-random-proportional rule* (5.8) in order to add more randomness by using the roulette wheel procedure to choose the solution components more often.

In order to set the constant (c_{SP}) for computing the scaling parameter, some graphical analysis was performed using one of the 25-node instances ($AP25LL\_3\_2$). No heuristic visibility information was computed, the initial pheromone values were all set to be the same $\tau_0 = 10$ and no local search was used. This way, it was possible to analyze the behavior of the algorithm as a pure ACS algorithm.

After some trial and error, the algorithm was run three times using $c_{SP} = 5 \times 10^7$, $c_{SP} = 1 \times 10^8$ and $c_{SP} = 5 \times 10^8$. The solution costs are plotted as can be seen in Figure 5.1.

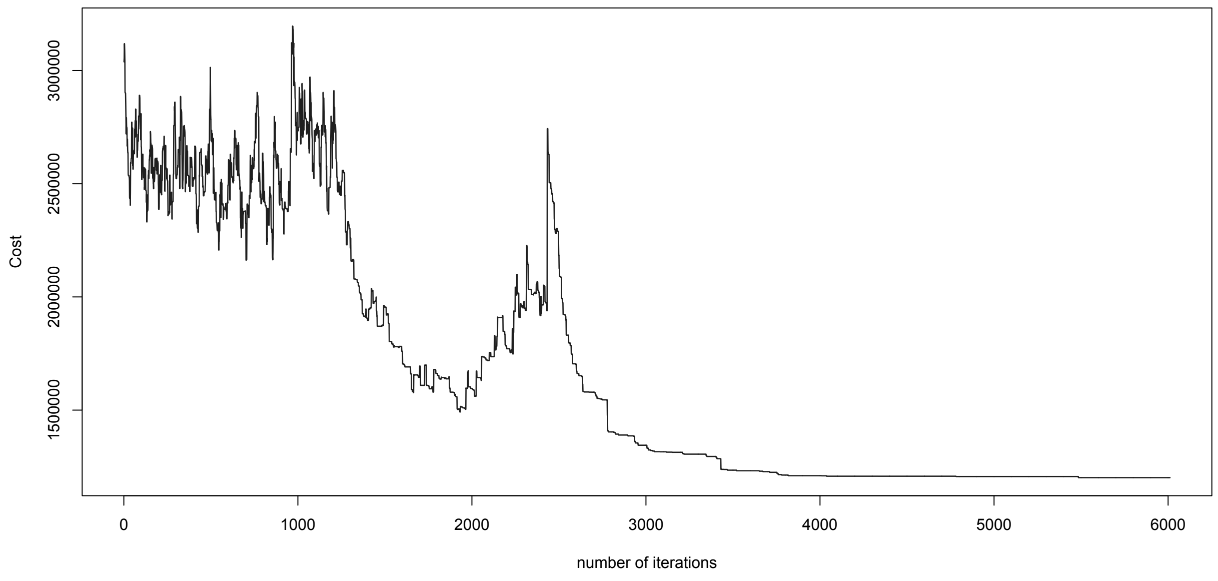


Figure 5.1: c_{SP} tuning ($c_{SP} = 5 \times 10^7$, $c_{SP} = 1 \times 10^8$ and $c_{SP} = 5 \times 10^8$).

The plot has all the data from the three runs. From zero to roughly 1000 iterations $c_{SP} = 5 \times 10^7$, from that point to the peak between 2000 and 3000 iterations $c_{SP} = 1 \times 10^8$ and from there until the end, $c_{SP} = 5 \times 10^8$. This illustrates the importance of the scaling parameter If too low, there is no early convergence to a minimum; if too high, it quickly converges to a low cost solution. In between, there is convergence but good solutions quickly evaporate and the quality of the solutions decrease. In the first run, the gap is 94.96%, in the second, the gap is 48.18% and in the third, the gap is 17.36%.

In order to analyze the influence of the local and global pheromone update parameters, ρ and γ, c_{SP} was set to 5×10^8 (the value that, in the previous runs led to a lower gap (17.36%) and ran for more iterations before stopping) and, three trials were made for $\gamma = 0.5$ and $\rho = \{0.1, 0.5, 0.9\}$ (Figure 5.2) $\rho = 0.5$ and and another three trials were made for $\gamma = \{0.1, 0.5, 0.9\}$ (Figure 5.3).

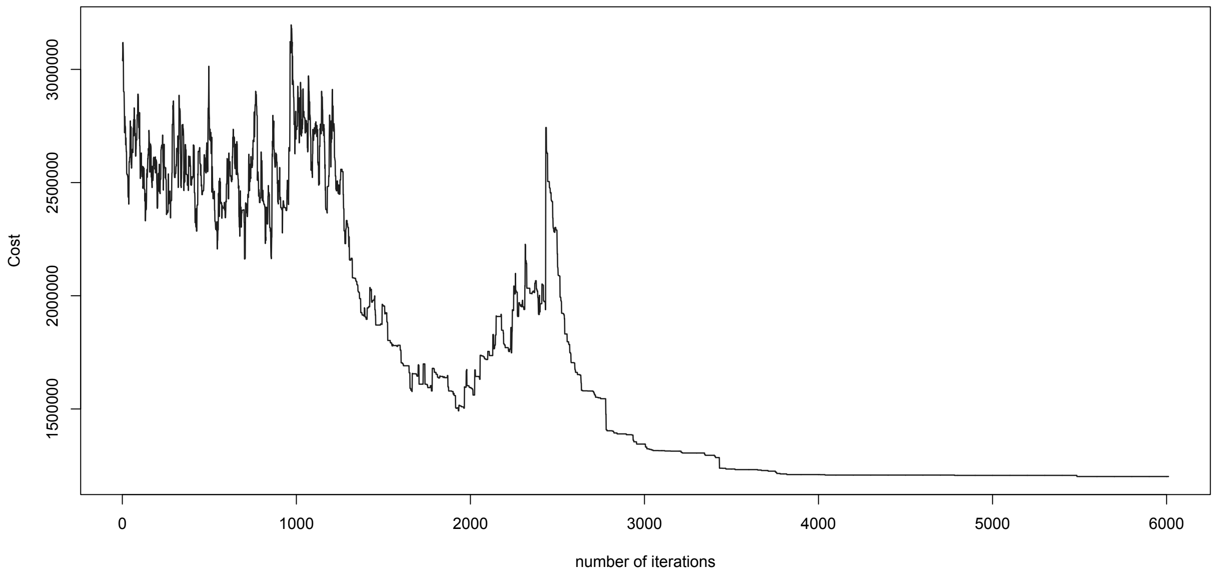


Figure 5.2: $\gamma = 0.5$ and $\rho = \{0.1, 0.5, 0.9\}$.

Graphical analysis on the variations in the value of ρ show that the higher this value, the more evaporation occurs when using the local pheromone update rule (5.2) and more oscillations occur in the solutions cost leading to a slower convergence (to good quality solutions).

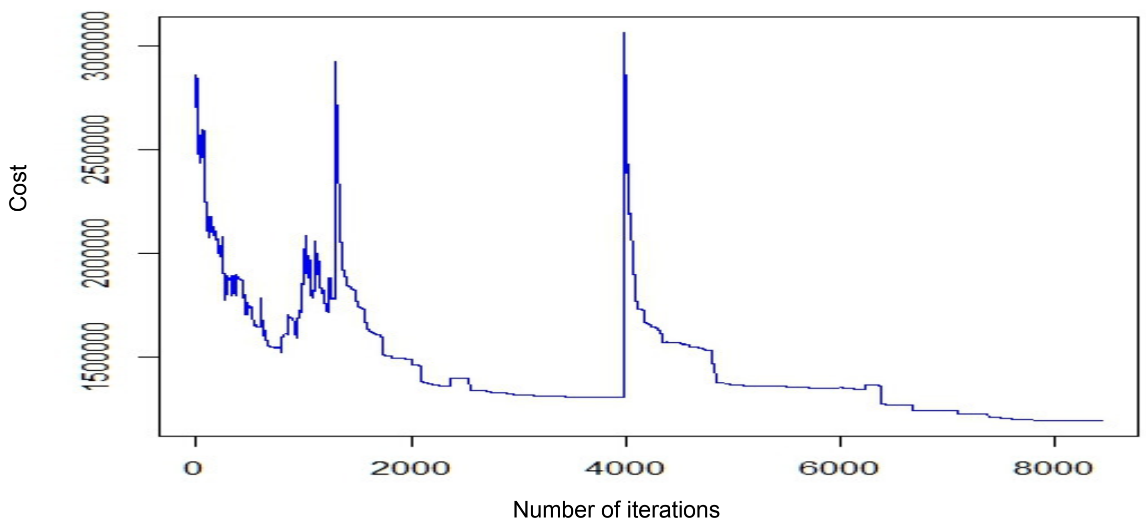


Figure 5.3: $\rho = 0.5$ and $\gamma = \{0.1, 0.5, 0.9\}$.

Concerning γ, the bigger its value, the more importance is given to better solutions when the global pheromone values are updated (5.1),which leads to better solutions.

After this graphical analysis, the parameters were chosen to be $\rho = \gamma = 0.5$. This allows a balance between evaporation (exploration) and the importance given to better solutions (exploitation). The parameter c_{SP} was set to 5×10^8 for the following tests.

Another possibility explored, was the possibility of applying the global pheromone update only when a better solution arises, in order to allow more exploration of the solutions space. The results showed that no convergence was achieved and that idea was abandoned.

In order to decide whether to use or not the heuristic visibility information, since it is only being used as an estimate for how good it is to open a hub, again, a graphical analysis was performed in order to better understand how the solutions change. The parameter β used in the *pseudo-random-proportional rule* equations (5.8) and (5.9) was set to 1. The algorithm was run using five different combinations: (i) no heuristic visibility information was used, (ii) the heuristic visibility information was used, (iii) the Close Hub neighborhood was used in the local search procedure, (iv) all neighborhoods defined were used in the local search procedure and finally, (v) the algorithm was run without using the heuristic visibility information but using the local search procedure in all neighborhoods.

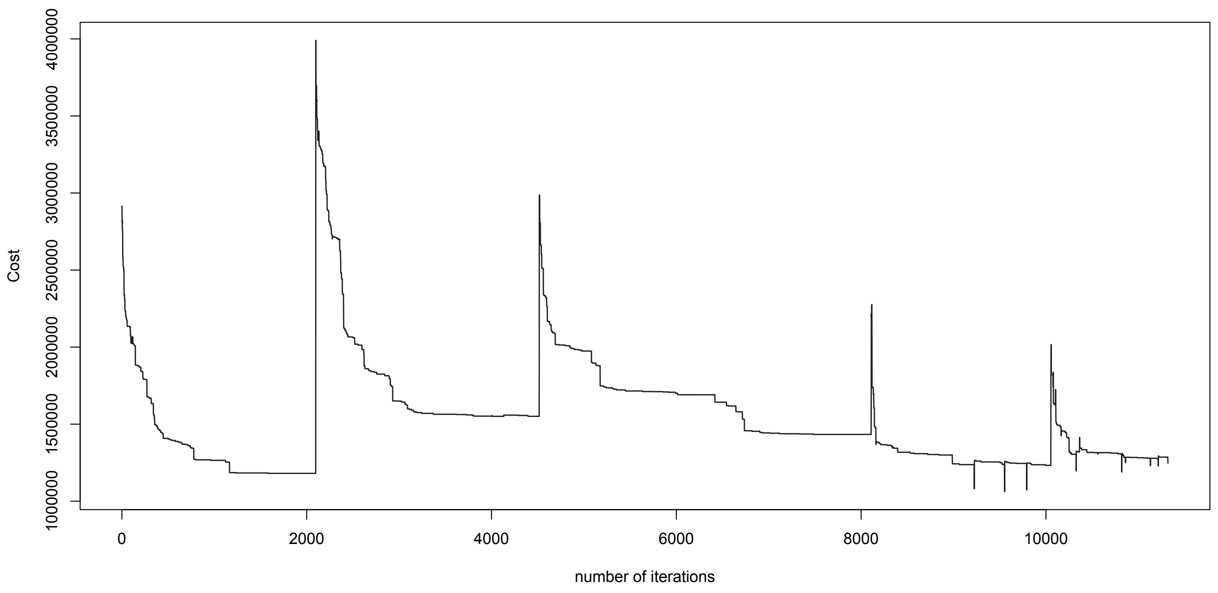


Figure 5.4: Different strategies for evaluating impact of the heuristic visibility information on the algorithm.

The resulting costs can be observed in Figure 5.4. Each peak represents the beginning of a new run (five peaks correspond to the five different combinations explained earlier, by that order). Using the heuristic visibility information leads to worse initial solutions as can be seen in the height difference between the first two peaks and the last two peaks in Figure 5.4. This is because the heuristic visibility information (see expression (5.6)) is only being considered for the hubs, resulting in more expensive initial solutions because more hubs are being opened. When coupled with local search, the solutions improve their quality (third and fourth peaks) and the more neighbourhoods are used in the local search, the better the solution quality (the fourth peak is lower than the third).

Finally, the biasing of the initial pheromone values, using (5.3) is introduced with $\tau_0 = 10$ and, after some trial and error, $c_\tau = 1000$, in order to really influence the initial solutions. Since the algorithm parameters have been changed, the constant (c_{SP}) for computing the scaling parameter is reset to 5×10^8 [2].

5.4.4 Computational Results

The optimal solutions resulting of the application of model P_{EK} proposed by Correia et al. [17] to the 96 instances tested are reported in Table 5.1 and in Table 5.2. In these tables, the costs (optimal or the best known values when the algorithm exceeds the time limit, which was set to 21600 CPU seconds), the CPU time (in seconds), and the Linear Relaxation gap is reported.

The gap LR(%) is computed according to:

$$100 \times \frac{v_{opt} - v_{lr}}{v_{opt}}, \tag{5.16}$$

where v_{opt} is the optimal value and v_{lr} is the optimal value of the linear relaxation.

Instance Name	Cost	CPU	gap LR (%)	Instance Name	Cost	CPU	gap LR (%)
AP10TT_1_1	373305.73	0.44	1.13	AP20TT_1_1	454585.44	1.27	3.35
AP10TT_2_1	802460.72	0.47	1.33	AP20TT_2_1	1031543.51	34.22	5.69
AP10TT_2_2	669140.22	0.66	1.41	AP20TT_2_2	812319.78	13.91	5.49
AP10TT_3_1	1324903.17	0.97	2.29	AP20TT_3_1	1698037.43	10199.02	7.13
AP10TT_3_2	1072251.82	1.16	3.67	AP20TT_3_2	1320147.80	1006.41	6.26
AP10TT_3_3	988190.15	0.48	1.69	AP20TT_3_3	1174193.95	64.47	5.68
AP10TL_1_1	360542.31	0.42	0.48	AP20TL_1_1	372396.86	2.38	1.14
AP10TL_2_1	759238.22	0.56	1.39	AP20TL_2_1	868492.64	46.06	5.60
AP10TL_2_2	651090.71	0.36	1.68	AP20TL_2_2	652470.36	2.02	1.54
AP10TL_3_1	1248820.32	1.19	1.96	AP20TL_3_1	1423219.92	5311.23	5.87
AP10TL_3_2	1032840.89	1.14	3.49	AP20TL_3_2	1118691.35	258.55	7.63
AP10TL_3_3	951846.09	0.56	1.20	AP20TL_3_3	958396.80	6.61	2.13
AP10LL_1_1	328309.70	0.08	0.00	AP20LL_1_1	325977.47	0.30	0.00
AP10LL_2_1	680443.07	0.14	0.00	AP20LL_2_1	704259.25	4.22	1.53
AP10LL_2_2	597717.67	0.53	0.23	AP20LL_2_2	610566.20	1.30	0.07
AP10LL_3_1	1080545.91	0.36	0.00	AP20LL_3_1	1120000.64	38.75	2.53
AP10LL_3_2	965614.24	2.08	2.51	AP20LL_3_2	983010.81	11.16	2.85
AP10LL_3_3	868749.91	0.56	0.25	AP20LL_3_3	887390.16	2.14	0.07
AP10LT_1_1	330606.63	0.06	0.00	AP20LT_1_1	335041.45	0.28	0.00
AP10LT_2_1	735440.65	0.11	0.00	AP20LT_2_1	753599.77	2.61	1.61
AP10LT_2_2	631761.86	0.09	0.00	AP20LT_2_2	636292.45	0.53	0.00
AP10LT_3_1	1173705.80	0.30	0.00	AP20LT_3_1	1205800.85	6.45	1.19
AP10LT_3_2	1050473.57	1.72	3.91	AP20LT_3_2	1081202.66	37.34	5.70
AP10LT_3_3	956319.65	0.23	0.00	AP20LT_3_3	959935.38	0.91	0.00

Table 5.1: Results for model P_{EK} for n=10 and n=20.

[2] The instance is run ten times for $c_{SP} = 1 \times 10^6$, $c_{SP} = 5 \times 10^6$, $c_{SP} = 1 \times 10^7$, $c_{SP} = 5 \times 10^7$, $c_{SP} = 1 \times 10^8$ and the parameter setting that resulted in a smaller mean gap was chosen.

Instance Name	Cost	CPU	gap LR (%)	Instance Name	Cost	CPU	gap LR (%)
AP25TT_1_1	577318.31	5.25	2.94	AP40TT_1_1	612664.89	71.00	6.64
AP25TT_2_1	1247408.79	104.97	3.08	AP40TT_2_1	1364597.72	4027.14	4.92
AP25TT_2_2	971393.52	48.25	2.63	AP40TT_2_2	1006896.99	259.44	4.77
AP25TT_3_1	2006562.93	5702.88	3.15	AP40TT_3_1	2301591.15	21600.52	5.10
AP25TT_3_2	1589773.98	5165.44	4.20	AP40TT_3_2	1738735.97	21600.77	9.49
AP25TT_3_3	1394724.32	125.70	2.87	AP40TT_3_3	1431168.52	2090.33	3.87
AP25TL_1_1	435894.81	3.88	2.54	AP40TL_1_1	434481.19	13.86	1.53
AP25TL_2_1	911486.56	50.73	2.80	AP40TL_2_1	930862.15	2632.98	3.30
AP25TL_2_2	740127.69	17.67	1.65	AP40TL_2_2	749877.33	81.39	2.50
AP25TL_3_1	1453956.36	748.19	3.54	AP40TL_3_1	1513020.42	21600.41	6.02
AP25TL_3_2	1196132.85	565.08	3.40	AP40TL_3_2	1240579.02	21600.67	5.84
AP25TL_3_3	1069550.51	74.08	1.83	AP40TL_3_3	1074257.17	470.53	2.18
AP25LL_1_1	358990.69	4.98	2.59	AP40LL_1_1	347298.69	6.39	0.17
AP25LL_2_1	742435.64	44.47	2.08	AP40LL_2_1	737969.08	92.22	0.88
AP25LL_2_2	621980.22	3.16	0.49	AP40LL_2_2	610494.06	18.88	0.20
AP25LL_3_1	1159701.17	359.64	2.33	AP40LL_3_1	1161202.48	1271.52	1.11
AP25LL_3_2	1025360.32	2574.88	5.66	AP40LL_3_2	987953.37	540.42	2.65
AP25LL_3_3	903161.79	6.44	0.31	AP40LL_3_3	894546.29	31.23	0.21
AP25LT_1_1	451019.52	3.20	6.60	AP40LT_1_1	425559.05	4.31	0.03
AP25LT_2_1	995526.01	84.42	9.16	AP40LT_2_1	924138.25	44.72	0.89
AP25LT_2_2	807650.76	5.98	3.51	AP40LT_2_2	751983.95	13.20	0.09
AP25LT_3_1	1607356.52	21601.52	8.66	AP40LT_3_1	1489401.00	464.72	1.33
AP25LT_3_2	1313391.47	664.70	5.68	AP40LT_3_2	1230197.58	125.50	3.23
AP25LT_3_3	1189573.44	13.81	2.63	AP40LT_3_3	1102584.51	20.88	0.11

Table 5.2: Results for model P_{EK} for n=25 and n=40.

As it can be observed in the tables, when solving instances $AP40TT\_3\_1$, $AP40TT\_3\_2$, $AP40TL\_3\_1$ and $AP40TL\_3\_2$ the time limit was reached. Accordingly, the value reported might not be the optimal value. The instances of the same nature (with tight costs, tight and loose capacities, 3 types of products and each hub handling 2 or 1 products — TX_3_x, $X \in \{T, L\}$, $x \in \{1, 2\}$) for 10, 20 and 25 nodes are the ones that require more computational time, which indicates that these instances are harder to solve. We note that the time limit was also reached when solving instance $AP25LT\_3\_1$ the time limit was reached. A deeper analysis of the results is out of the context of this thesis and can be found in Correia et al. [17].

Using the parameter settings described in subsection 5.4.3, all instances were run 10 times. In Tables 5.3 – 5.6 the results of applying the heuristic to the instances is presented. The minimum, maximum and average gap (the first three columns) are reported. The CPU time (in seconds) required when using model P_{EK} is also reported (column "Opt") in this table, for comparison with the average of the total CPU run time (column "Heuristic") and the average elapsed CPU (in seconds) time until the best solution was found are reported (column "Heuristic Best").

The gap(%) is computed according to:

$$100 \times \frac{v_{heur} - v_{opt}}{v_{opt}}, \tag{5.17}$$

where v_{opt} is the optimal value and v_{heur} is the optimal value of the linear relaxation.

As can be seen in Table 5.3, for the 10-node instances, the optimum was reached for 19 out of the 24 instances in some run (by analysis of the Min Gap (%) column). The instances where the optimum solution was never reached are the ones with a high linear relaxation gap but that does not happen for all such instances, e. g., instance $AP10TT\_3\_1$ has a gap of 2.21% and the optimum solution was found. The instances with a 0% linear relaxation gap reach the optimal solutions in all runs but generally, in a higher amount time than the time elapsed when using P_{EK}.

Instance Name	Gap (%)			CPU (seconds)		
	Min	Average	Max	Opt	Heuristic Best	Heuristic
AP10TT_1_1	0.00	1.62	13.60	0.44	1.25	3.58
AP10TT_2_1	0.00	0.06	0.16	0.47	0.95	5.34
AP10TT_2_2	0.00	0.32	0.70	0.66	1.44	4.76
AP10TT_3_1	0.00	7.69	14.10	0.97	2.82	8.86
AP10TT_3_2	0.10	2.67	5.83	1.16	6.55	10.29
AP10TT_3_3	0.47	0.86	0.95	0.48	6.87	9.58
AP10TL_1_1	0.00	0.00	0.00	0.42	0.14	3.79
AP10TL_2_1	0.00	1.61	15.75	0.56	0.79	5.45
AP10TL_2_2	0.00	1.12	11.95	0.36	2.93	5.15
AP10TL_3_1	0.00	3.29	10.98	1.19	4.45	9.85
AP10TL_3_2	0.26	3.21	5.66	1.14	4.37	10.02
AP10TL_3_3	0.00	1.12	6.28	0.56	3.47	8.23
AP10LL_1_1	0.00	0.00	0.00	0.08	1.78	4.76
AP10LL_2_1	0.00	0.00	0.00	0.14	1.19	5.96
AP10LL_2_2	0.00	4.70	17.22	0.53	1.21	5.89
AP10LL_3_1	0.00	0.00	0.00	0.36	0.15	8.60
AP10LL_3_2	0.62	2.00	10.82	2.08	4.41	10.53
AP10LL_3_3	0.00	2.33	12.81	0.55	2.85	8.81
AP10LT_1_1	0.00	0.00	0.00	0.06	0.12	4.86
AP10LT_2_1	0.00	0.00	0.00	0.11	0.80	6.20
AP10LT_2_2	0.00	0.00	0.00	0.09	0.37	5.89
AP10LT_3_1	0.00	0.00	0.00	0.30	3.81	9.79
AP10LT_3_2	1.98	2.38	3.12	1.72	2.08	10.22
AP10LT_3_3	0.00	0.00	0.00	0.23	4.33	9.59

Table 5.3: Results for instances with 10 nodes.

Instance Name	Gap (%)			CPU (seconds)		
	Min	Average	Max	Opt	Heuristic Best	Heuristic
AP20TT_1_1	9.58	12.40	18.71	1.27	3.22	9.27
AP20TT_2_1	0.00	3.73	10.20	34.22	10.88	16.43
AP20TT_2_2	2.91	8.14	13.24	13.91	14.89	19.68
AP20TT_3_1	2.26	9.78	43.56	10199.02	18.65	30.46
AP20TT_3_2	3.49	7.45	20.38	1006.41	25.50	35.16
AP20TT_3_3	1.02	5.61	7.39	64.47	21.61	31.61
AP20TL_1_1	0.00	3.01	17.61	2.38	7.80	14.79
AP20TL_2_1	0.81	7.37	23.97	46.06	16.91	28.87
AP20TL_2_2	0.00	2.92	18.53	2.02	13.71	25.337
AP20TL_3_1	1.81	6.22	15.77	5311.23	30.58	44.71
AP20TL_3_2	0.00	7.13	35.63	258.55	17.98	34.81
AP20TL_3_3	0.00	4.54	19.80	6.61	22.30	38.41
AP20LL_1_1	0.00	0.00	0.00	0.30	4.87	17.01
AP20LL_2_1	11.54	33.64	36.10	4.22	2.96	25.08
AP20LL_2_2	0.00	5.07	15.53	1.30	10.28	28.10
AP20LL_3_1	0.67	14.31	32.39	38.75	23.15	44.93
AP20LL_3_2	0.00	6.22	24.78	11.16	8.88	36.06
AP20LL_3_3	0.00	13.78	31.45	2.14	11.22	36.35
AP20LT_1_1	0.00	1.56	15.58	0.28	2.46	18.56
AP20LT_2_1	0.07	2.07	17.97	2.61	12.49	31.23
AP20LT_2_2	0.00	1.11	11.07	0.53	3.97	27.96
AP20LT_3_1	0.00	11.97	37.83	6.45	30.77	52.14
AP20LT_3_2	0.34	2.36	11.17	37.34	25.63	49.28
AP20LT_3_3	0.00	3.27	13.37	0.91	8.09	39.89

Table 5.4: Results for instances with 20 nodes.

Instance Name	Gap (%)			CPU (seconds)		
	Min	Average	Max	Opt	Heuristic Best	Heuristic
AP25TT_1_1	0.71	3.24	19.77	5.25	9.34	15.37
AP25TT_2_1	1.30	5.25	27.18	104.97	24.82	33.30
AP25TT_2_2	1.24	1.83	2.70	48.25	19.49	25.89
AP25TT_3_1	3.30	9.14	23.04	5702.88	41.24	51.68
AP25TT_3_2	1.83	4.21	9.01	5165.44	34.63	45.40
AP25TT_3_3	1.05	4.18	8.41	125.70	40.08	50.91
AP25TL_1_1	0.00	12.11	34.44	3.88	7.78	15.40
AP25TL_2_1	0.83	7.65	13.21	50.73	16.23	25.76
AP25TL_2_2	0.68	4.66	15.01	17.67	23.31	36.83
AP25TL_3_1	4.43	9.68	32.70	748.19	36.38	50.82
AP25TL_3_2	1.63	6.61	11.99	565.08	36.21	57.25
AP25TL_3_3	0.61	4.98	16.67	74.08	25.62	37.30
AP25LL_1_1	2.37	12.59	76.19	4.98	3.24	13.21
AP25LL_2_1	2.49	7.51	30.90	44.47	18.76	26.70
AP25LL_2_2	0.00	11.03	41.28	3.16	12.50	22.55
AP25LL_3_1	0.45	14.89	85.49	359.64	22.54	37.46
AP25LL_3_2	0.94	4.92	13.95	2574.88	19.95	36.01
AP25LL_3_3	0.00	31.48	79.82	6.44	12.81	33.09
AP25LT_1_1	4.79	5.58	10.19	3.20	5.93	14.93
AP25LT_2_1	0.25	2.91	11.90	84.42	18.84	26.38
AP25LT_2_2	0.00	0.00	0.00	5.98	14.99	22.33
AP25LT_3_1	1.29	10.54	31.27	21601.52	30.58	41.44
AP25LT_3_2	0.44	9.61	61.51	664.70	28.38	41.45
AP25LT_3_3	0.00	4.98	12.74	13.81	22.78	33.60

Table 5.5: Results for instances with 25 nodes.

Instance Name	Gap (%)			CPU (seconds)		
	Min	Average	Max	Opt	Heuristic Best	Heuristic
AP40TT_1_1	4.55	12.02	24.08	71.00	26.04	40.55
AP40TT_2_1	4.15	14.58	29.13	4027.14	79.60	105.09
AP40TT_2_2	0.36	14.56	32.62	259.44	69.08	95.39
AP40TT_3_1	5.46	13.15	23.74	21600.52	143.79	165.15
AP40TT_3_2	**-0.05**	7.29	30.47	21600.77	109.01	141.48
AP40TT_3_3	7.84	18.36	31.21	2090.33	117.10	159.13
AP40TL_1_1	0.43	9.38	36.56	13.86	25.03	40.84
AP40TL_2_1	5.34	11.17	22.72	2632.98	81.81	105.44
AP40TL_2_2	0.03	3.56	18.13	81.39	66.68	89.25
AP40TL_3_1	1.41	9.29	19.96	21600.41	116.85	160.05
AP40TL_3_2	2.47	9.19	17.73	21600.67	123.61	169.22
AP40TL_3_3	0.29	15.13	46.37	470.53	112.31	158.95
AP40LL_1_1	0.22	9.66	52.37	6.39	18.76	37.51
AP40LL_2_1	0.11	10.34	29.68	92.22	64.98	92.31
AP40LL_2_2	0.26	3.01	19.39	18.88	34.89	63.29
AP40LL_3_1	2.01	14.80	42.72	1271.52	101.97	150.63
AP40LL_3_2	4.45	25.79	68.84	540.42	84.86	132.20
AP40LL_3_3	0.28	11.77	45.73	31.23	71.48	119.25
AP40LT_1_1	0.00	10.03	56.08	4.31	20.02	35.08
AP40LT_2_1	0.00	21.27	67.63	44.72	78.16	104.68
AP40LT_2_2	0.05	2.79	19.90	13.20	46.48	72.17
AP40LT_3_1	7.51	16.16	32.70	464.72	122.76	166.17
AP40LT_3_2	0.60	15.44	32.68	125.50	104.72	145.47
AP40LT_3_3	0.08	7.79	24.53	20.88	98.38	141.02

Table 5.6: Results for instances with 40 nodes.

For the instances with 20 nodes (Table 5.4), in 13 out of the 24 instances the optimum solution was found. Only one instance with the linear relaxation gap of 0% reached the optimum value in all runs. The worse behaviour was observed for instance $AP20LL\_2\_1$, where a solution with a gap of 36.10% is found for 9 out of the 10 runs.

In Table 5.5 (instances with 25 nodes) the CPU time elapsed using the heuristic was generally smaller than when using the model P_{EK}. In 5 instances an optimal solution was found and in 12 the minimum gap is below 1%. Considering the average gap, 8 out of the 24 instances reached gaps lower than 5%. Instance $AP25LT\_3\_1$ was not solved to an optimum by model P_{EK} but the best result by the heuristic reached a 1.29% gap. The worst gap was of 79.82%, for instance $AP25LL\_3\_3$ but in other runs, an optimal solution was found for that instance.

Finally, for the instances with 40 nodes (Table 5.6) a solution better than the solution found using the model P_{EK} was found for instance $AP40TT\_3\_2$ with a gap of -0.05%. Optimum solutions were found for 2 instances and the minimum gap was lower than 1% for 14 out of the 24 instances.

	Tot Nr It	Best It	vACO	tACO	totaltACO	gap (%)
	2269	2239	1751968.55	177.96	181.23	0.76
	812	310	1977792.12	41.77	78.14	13.75
	1816	1314	1740344.62	120.94	158.54	0.09
	982	480	1890740.68	53.72	90.22	8.74
	1345	843	1908190.98	91.52	129.50	9.75
	511	9	2268605.47	16.81	90.22	30.47
	1614	1429	1747049.70	165.92	181.44	0.48
	1412	1116	1880479.64	156.26	181.44	8.15
	1362	971	1766101.34	147.86	181.51	1.57
	1349	**1255**	**1737932.42**	**165.19**	**181.51**	**-0.05**
	897	652	1762083.35	47.27	83.36	1.34
	1877	1375	1954253.86	122.92	160.65	12.40
Mean	1353.83	999.42	1865461.89	109.01	141.48	7.29

Table 5.7: Computational results for 12 runs of the heuristic on instance $AP40TT\_3\_2$.

In Table 5.7, results for 12 runs of instance $AP40TT\_3\_2$ (after finding the best solution, the algorithm ran two more times in order to see if a new best was observed). The table reports: the total number of iterations (Tot Nr Iter), the number of iterations until the best solution was found (Best It), the solution cost found by the algorithm (vACO), the time elapsed until the best solution was found (tACO, in CPU seconds), the total run time (totaltACO, in CPU seconds) and the gap between the solution cost found by the algorithm and the best found by applying the model PEK. As can be seen in Table 5.7[3], the run that found the best gap was stopped because the time limit imposed by the stopping criterion of 180 CPU seconds was reached and so, an even better solution could have been found if the algorithm kept running. Overall, the CPU time elapsed using the heuristic was generally smaller than when using the model P_{EK} and the minimum gap found averaged 1.99%.

[3]There are 96 of these tables regarding the results found for all instances but Tables 5.3 –5.3 are the only ones presented in this thesis because they are enough to understand the results.

N	Gap (%)			CPU (seconds)		
	Min	Average	Max	Opt	Heuristic Best	Heuristic
10	0.14	1.46	5.41	0.61	2.46	7.33
20	1.44	7.24	20.50	710.51	14.53	30.67
25	1.28	7.90	27.89	1582.47	21.93	33.13
40	1.99	11.94	34.37	4111.79	79.93	112.10

Table 5.8: Mean results found for 10, 20, 25 and 40 nodes.

In Table 5.8, the results obtained are aggregated by number of nodes and it can be seen that for instances with more than 20 nodes, the heuristic proposed finds solutions faster than when using the model P_{EK}. The minimum gaps never exceed 2% but, on average, the gaps exceed the 2%.

CHAPTER 6

Conclusions

In this chapter the work done is summarized and some conclusions are drawn. Finally, some ideas on how to improve the work done are presented.

6.1 Summary and Conclusions

In this work, a heuristic was proposed for tackling the MP-CSAHLP, which is an $\mathcal{NP}$-Hard problem. The new heuristic is based on Ant Colony System, a population based constructive heuristic that is based on the foraging behaviour of ants. Local search was used in the algorithm as a *daemon action* and the solution components of the linear relaxation solved using model P_{EK} proposed by Correia et al. [17] were used to bias the initial pheromone values on each solution component.

In preliminary computational tests, the algorithm parameters were set using one of the instances and a graphical analysis.

In some cases the optimal solution was found and in one case, a solution better than the one found using the model P_{EK} was obtained in a shorter amount of time. This best solution was found for one of the hardest instances to solve (with tight costs and tight capacity constraints, three types of products flowing through the network and the hubs not being allowed to handle more than 2 products). This is an indication that the algorithm can be better tuned to solve the problem and find better solutions than the ones found using the model in a smaller amount of time, even for the hardest instances. In this particular case, the algorithm stopped because it reached the time limit defined as a stopping criterion and so, again there is an indication that the parameters defined for the algorithm to stop, if better tuned, can help improve the performance of the algorithm.

The local search procedure applied was chosen to be very simple in order to reduce the computational time. It stops once a better solution is found and explores only three different types of neighborhoods.

Additionally, $c_{\tau} = 1000$ is a high value that leads to good initial solutions but perhaps biasing too much the initial pheromone levels.

In conclusion the results show that the proposed heuristic has the potential to have a better performance with finer parameter tuning, longer runs and more intense local search.

6.2 Overview and Future Work

In a work like the one presented in this thesis, several improvements can be performed which emerge in the sequel of all the tests performed. The improvements proposed are described as follows.

- **Parameter Tuning**

  In order to improve the performance of the algorithm, more intensive computational tests will have to be carried out in order to set the parameters. These parameters should have been set using a more fine parameter tuning by running all instances for the a given number of nodes.

  The algorithm has to be run more times, in order to use the average results associated with the graphical analysis.

  The number of iterations on each run has to be higher and more time has to be allowed for the algorithm to find better results.

  The algorithm was shown to be sensible to the scaling parameter since better results were obtained when the scaling parameter, SP ,changed from considering the fixed costs to considering the linear relaxation solution cost. The scaling parameter should be redefined in order for it not to depend on the constant c_{SP}.

  Also, a good idea is to study the evolution of the pheromone values in order to better understand how to decide the pheromone update parameters ρ and γ and a better estimate of the solution cost has to be defined in order to increase the pheromone levels, in the global pheromone update more efficiently.

- **Local Search**

  A more intensive local search should be implemented by defining additional neighborhoods in order to improve the solution quality. This might result in longer computational time used when running the algorithm thus, a compromise between the computational effort put into the local search and the run time of the algorithm will have to be defined.

- **Larger Instances**

  The algorithm should be used to solve larger instances of the MP-CSAHLP and the upper bound provided by the heuristic used an an upper cut off to be used in CPLEX.

Bibliography

[1] S. Abdinnour-Helm. A hybrid heuristic for the uncapacitated hub location problem. *European Journal of Operational Research*, 106(2):489–499, 1998.

[2] S. Abdinnour-Helm and M. A. Venkataramanan. Solution approaches to hub location problems. *Annals of Operations Research*, 78:31–50, 1998.

[3] S. Alumur and B. Y. Kara. Network hub location problems: The state of the art. *European Journal of Operational Research*, 190(1):1–21, 2008.

[4] T. Aykin. Networking policies for hub-and-spoke systems with application to the air transportation system. *Transportation Science*, 29(3):201–221, 1995.

[5] R. Bollapragada, J. Camm, U. S. Rao, and J. Wu. A two-phase greedy algorithm to locate and allocate hubs for fixed-wireless broadband access. *Operations Research Letters*, 33(2):134–142, 2005.

[6] B. Bullnheimer, R. F. Hartl, and C. Strauss. A new rank based version of the ant system - a computational study. *Central European Journal for Operations Research and Economics*, 7:25–38, 1997.

[7] H. Calık, S. A. Alumur, B. Y. Kara, and O. E. Karasan. A tabu-search based heuristic for the hub covering problem over incomplete hub networks. *Computers & Operations Research*, 36(12):3088–3096, 2009.

[8] J. F. Campbell. Integer programming formulations of discrete hub location problems. *European Journal of Operational Research*, 72(2):387–405, 1994.

[9] J. F. Campbell. Hub location and the p-hub median problem. *Operations Research*, 44(6):923–935, 1996.

[10] J. F. Campbell and M. E. O'Kelly. Twenty-five years of hub location research. *Transportation Science*, 46(2):153–169, 2012.

[11] J. Chen. A hybrid heuristic for the uncapacitated single allocation hub location problem. *Omega*, 35(2):211–220, 2007.

[12] J. Chen. A heuristic for the capacitated single allocation hub location problem. In A. Chan and S. Ao, editors, *Advances in Industrial Engineering and Operations Research*, volume 5, chapter Lecture Notes in Electrical Engineering, pages 185–196. Springer, 2008.

[13] I. Contreras. Hub location problems. In Stefan Nickel Gilbert Laporte and Francisco Saldanha da Gama, editors, *Location Science*, chapter 12, pages 311–344. Springer, Berlin-Heidelberg, 2015.

[14] I. Contreras, J. A. Díaz, and E. Fernández. Lagrangean relaxation for the capacitated hub location problem with single assignment. *OR Spectrum*, 31(3):483–505, 2009.

[15] I. Contreras, J. A. Díaz, and E. Fernández. Branch and price for large-scale capacitated hub location problems with single assignment. *INFORMS Journal on Computing*, 23(1):41–55, 2011.

[16] I. Correia, S. Nickel, and F. Saldanha-da-Gama. The capacitated single-allocation hub location problem revisited: A note on a classical formulation. *European Journal of Operational Research*, 207(1):92–96, 2010.

[17] I. Correia, S. Nickel, and F. Saldanha-da-Gama. Multi-product capacitated single-allocation hub location problems: formulations and inequalities. *Networks and Spatial Economics*, 14(1):1–25, 2014.

[18] C. B. Cunha and M. R. Silva. A genetic algorithm for the problem of configuring a hub-and-spoke network for a ltl trucking company in brazil. *European Journal of Operational Research*, 179(3):747–758, 2007.

[19] J. L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels. The self-organizing exploratory pattern of the argentine ant. *Journal of Insect Behavior*, 3(2):159–168, 1990.

[20] M. Dorigo, M. Birattari, and T. Stützle. Ant colony optimization. *Computational Intelligence Magazine, IEEE*, 1(4):28–39, 2006.

[21] M. Dorigo, M. Birattari, and T. Stützle. Artificial ants as a computational intelligence technique. *IEEE Computational Intelligence Magazine*, 1:28–39, 2006.

[22] M. Dorigo and G. D. Caro. The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo, F. Glover, D. Dasgupta, P. Moscato, R. Poli, and K. V. Price, editors, *in New Ideas in Optimization*, pages 11–32. McGraw-Hill Ltd., UK, Maidenhead, UK, England, 1999.

[23] M. Dorigo, G. D. Caro, and L. M. Gambardella. Ant algorithms for discrete optimization. *Artificial Life*, 5(2):137–172, 1999.

[24] M. Dorigo and L. M. Gambardella. Ant colonies for the travelling salesman problem. *BioSystems*, 43(2):73–81, 1997.

[25] M. Dorigo and L. M. Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, 1997.

[26] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: An autocatalytic optimizing process. Technical report, Technical report, 1991.

[27] M. Dorigo, V. Maniezzo, and A. Colorni. Positive feedback as a search strategy. Technical Report 91-016, Politecnico di Milano, Dipartimento di Elettronica, Milan, Italy, 1991.

[28] M. Dorigo, V. Maniezzo, and A. Colorni. Ant system: optimization by a colony of cooperating agents. *Systems, Man, and Cybernetics, Part B: IEEE Transactions on Cybernetics*, 26(1):29–41, 1996.

[29] M. Dorigo and T. Stützle. *Ant Colony Optimization.* MIT Press, Cambridge, MA, 2004.

[30] M. Dorigo and T. Stützle. Ant colony optimization: overview and recent advances. In M. Gendreau and J. Potvin, editors, *Handbook of Metaheuristics*, volume 146 of *International Series in Operations Research & Management Science*, pages 227–263. Springer US, 2010.

[31] H. A. Eiselt and V. Marianov. A conditional p-hub location problem with attraction functions. *Computers & Operations Research*, 36(12):3128–3135, 2009.

[32] A. T. Ernst, H. Hamacher, H. Jiang, M. Krishnamoorthy, and G. Woeginger. Uncapacitated single and multiple allocation p-hub center problems. *Computers & Operations Research*, 36(7):2230–2241, 2009.

[33] A. T. Ernst and M. Krishnamoorthy. Efficient algorithms for the uncapacitated single allocation p-hub median problem. *Location Science*, 4(3):139–154, 1996.

[34] A. T. Ernst and M. Krishnamoorthy. Exact and heuristic algorithms for the uncapacitated multiple allocation p-hub median problem. *European Journal of Operational Research*, 104(1):100–112, 1998.

[35] A. T. Ernst and M. Krishnamoorthy. Solution algorithms for the capacitated single allocation hub location problem. *Annals of Operations Research*, 86:141–159, 1999.

[36] R. Z. Farahani, M. Hekmatfar, A. B. Arabani, and E. Nikbakhsh. Hub location problems: A review of models, classification, solution techniques, and applications. *Computers & Industrial Engineering*, 64(4):1096–1109, 2013.

[37] V. Filipović. An electromagnetism metaheuristic for the uncapacitated multiple allocation hub location problem. *Serdica Journal of Computing*, 5(3):261–272, 2011.

[38] L. M. Gambardella, E. D. Taillard, and M. Dorigo. Ant colonies for the quadratic assignment problem. *Journal of the Operational Research Society*, pages 167–176, 1999.

[39] A. J. Goldman. Optimal locations for centers in a network. *Transportation Science*, 3(4):352–360, 1969.

[40] S. Goss, S. Aron, J. Deneubourg, and J. M. Pasteels. Self-organized shortcuts in the argentine ant. *Naturwissenschaften*, 76(12):579–581, 1989.

[41] A. Ilić, D. Urošević, J. Brimberg, and N. Mladenović. A general variable neighborhood search for solving the uncapacitated single allocation p-hub median problem. *European Journal of Operational Research*, 206(2):289–300, 2010.

[42] J. G. Klincewicz. Avoiding local optima in the p-hub location problem using tabu search and grasp. *Annals of Operations Research*, 40(1):283–302, 1992.

[43] J. Kratica, M. Milanović, Z. Stanimirović, and D. Tošić. An evolutionary-based approach for solving a capacitated hub location problem. *Applied Soft Computing*, 11(2):1858–1866, 2011.

[44] J. Kratica, Z. Stanimirović, D. Tošić, and V. Filipović. Two genetic algorithms for solving the uncapacitated single allocation p-hub median problem. *European Journal of Operational Research*, 182(1):15–28, 2007.

[45] J. Kratica, Z. Stanimirović, D. Tošić, and V. Filipović. Genetic algorithm for solving uncapacitated multiple allocation hub location problem. *Computing and Informatics*, 24(4):415–426, 2012.

[46] C. Lin, J. Lin, and Y. Chen. The capacitated p-hub median problem with integral constraints: An application to a chinese air cargo network. *Applied Mathematical Modelling*, 36(6):2777–2787, 2012.

[47] K. Lu and C. Ting. Lagrangian relaxation for the capacitated single allocation p-hub median problem. *Journal of the Eastern Asia Society for Transportation Studies*, 10(0):851–863, 2013.

[48] A. Lüer-Villagra and V. Marianov. A competitive hub location and pricing problem. *European Journal of Operational Research*, 231(3):734–744, 2013.

[49] V. Marianov and D. Serra. Location models for airline hubs behaving as m/d/c queues. *Computers & Operations Research*, 30(7):983–1003, 2003.

[50] V. Marianov, Daniel Serra, and Charles ReVelle. Location of hubs in a competitive environment. *European Journal of Operational Research*, 114(2):363–371, 1999.

[51] T. Meyer, A. T. Ernst, and M. Krishnamoorthy. A 2-phase algorithm for solving the single allocation p-hub center problem. *Computers & Operations Research*, 36(12):3143–3151, 2009.

[52] M. Mohammadi, F. Jolai, and H. Rostami. An m/m/c queue model for hub covering location problem. *Mathematical and Computer Modeling*, 54(11):2623–2638, 2011.

[53] M. E. O'Kelly. Activity levels at hub facilities in interacting networks. *Geographical Analysis*, 18(4):343–356, 1986.

[54] M. E. O'kelly. The location of interacting hub facilities. *Transportation Science*, 20(2):92–106, 1986.

[55] M. E. O'kelly. A quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research*, 32(3):393–404, 1987.

[56] M. E. O'Kelly. Hub facility location with fixed costs. *Papers in Regional Science*, 71(3):293–306, 1992.

[57] F Selcen Pamuk and Canan Sepil. A solution to the hub center problem via a single-relocation algorithm with tabu search. *IIE Transactions*, 33(5):399–411, 2001.

[58] J. Peiró, A. Corberán, and R. Martí. Grasp for the uncapacitated r-allocation p-hub median problem. *Computers & Operations Research*, 43:50–60, 2014.

[59] H. Pirkul and D. A. Schilling. An efficient procedure for designing single allocation hub and spoke systems. *Management Science*, 44(12-part-2):S235–S242, 1998.

[60] M. Randall. Solution approaches for the capacitated single allocation hub location problem using ant colony optimization. *Computational Optimization and Applications*, 39(2):239–261, 2008.

[61] I. Rodríguez-Martín and J. Salazar-González. An iterated local search heuristic for a capacitated hub location problem. In F. Almeida, M. J. Blesa Aguilera, C. Blum, and V. Moreno, editors, *Hybrid Metaheuristics*, volume 4030 of *Lecture Notes in Computer Science*, pages 70–81. Springer, Berlin-Heidelberg, 2006.

[62] I. Rodríguez-Martín and J. Salazar-González. Solving a capacitated hub location problem. *European Journal of Operational Research*, 184(2):468–479, 2008.

[63] K. E. Rosing and C. S. ReVelle. Heuristic concentration: Two stage solution construction. *European Journal of Operational Research*, 97(1):75–86, 1997.

[64] K. E. Rosing, C. S. Revelle, and D. A. Schilling. A gamma heuristic for the p-median problem. *European Journal of Operational Research*, 117(3):522–532, 1999.

[65] Mihiro Sasaki, Atsuo Suzuki, and Zvi Drezner. On the selection of hub airports for an airline hub-and-spoke system. *Computers & Operations Research*, 26(14):1411–1422, 1999.

[66] M. R. Silva and C. B. Cunha. New simple and efficient heuristics for the uncapacitated single allocation hub location problem. *Computers & Operations Research*, 36(12):3152–3165, 2009.

[67] D. Skorin-Kapov and J. Skorin-Kapov. On tabu search for the location of interacting hub facilities. *European Journal of Operational Research*, 73(3):502–509, 1994.

[68] D. Skorin-Kapov, J. Skorin-Kapov, and M. E. O'Kelly. Tight linear programming relaxations of uncapacitated p-hub median problems. *European Journal of Operational Research*, 94(3):582–593, 1996.

[69] K. Smith, M. Krishnamoorthy, and M. Palaniswami. Neural versus traditional approaches to the location of interacting hub facilities. *Location Science*, 4(3):155–171, 1996.

[70] J. Sohn and S. Park. Efficient solution procedure and reduced size formulations for p-hub location problems. *European Journal of Operational Research*, 108(1):118–126, 1998.

[71] Z. Stanimirović. A genetic algorithm approach for the capacitated single allocation p-hub median problem. *Computing and Informatics*, 29(1):117–132, 2012.

[72] T. Stützle and H. Hoos. Max-min ant system and local search for the traveling salesman problem. In *IEEE International Conference on Evolutionary Computation (ICEC'97)*, pages 309–314. IEEE, IEEE Press, 1997.

[73] K. Takano and M. Arai. A genetic algorithm for the hub-and-spoke problem applied to containerized cargo transport. *Journal of Marine Science and Technology*, 14(2):256–274, 2009.

[74] P. Z. Tan and B. Y. Kara. A hub covering model for cargo delivery systems. *Networks*, 49(1):28–39, 2007.

[75] H. Topcuoglu, F. Corut, M. Ermis, and G. Yilmaz. Solving the uncapacitated hub location problem using genetic algorithms. *Computers & Operations Research*, 32(4):967–984, 2005.

[76] B. Wagner. A note on "location of hubs in a competitive environment". *European Journal of Operational Research*, 184(1):57–62, 2008.

[77] H. Yaman. Allocation strategies in hub networks. *European Journal of Operational Research*, 211(3):442–451, 2011.

[78] H. Yaman and G. Carello. Solving the hub location problem with modular link capacities. *Computers & Operations Research*, 32(12):3227–3245, 2005.