Ant Colony Optimization for a Network Design Problem in Freight Service

Leopold Kuttner[1]

[1] TU Dresden, Faculty of Business and Economics, 01069 Dresden, Germany
leopold.kuttner@tu-dresden.de

Abstract. This paper studies a restrictive variant of the Network Design Problem in the literature also referred to as the Railroad Blocking Problem. The formulation presented here is used by Deutsche Bahn AG to assist in making network design decisions. An unconventional ant colony optimization metaheuristic is proposed, which utilizes a weighted shortest path algorithm in combination with two kinds of pheromones in order to steer the routing. Computational tests are performed on several large-scale, real-world instances. The results of the ant system are compared to those produced by a problem specific simulated annealing based heuristic as well as an exact solver.

Keywords: Railroad Blocking, Network Design, Ant Colony Optimization, Sequential Routing, Capacitated Multi-Commodity Flow.

1 Introduction

The planning of freight transportation is often broken down into multiple consecutive optimization processes, including the design of the transportation network, blocking of cars and routing of trains, as well as creating timetables and crew scheduling, building up to an integrated system for managing the planning process as a whole [1]. These problems are well understood, but with the growing need to more accurately map reality the computational models become ever more challenging. This is why heuristic methods are popular for solving complex planning problems and are used widely in the industry.

This paper studies the subproblem of creating blocking plans for single car transportation, it is the foundation upon which subsequent planning processes are built. A blocking plan describes a succession of stations for single railroad cars [2]. The physical process of consolidating multiple cars is also referred to as classification and does, in addition to the locomotive and car costs, constitute a major fraction of the overall costs incurred by the routing. An efficient blocking plan saves handling costs at classification yards, allows for greater utilization of train capacities and reduces the total kilometers traveled of cars, trains and personnel.

In sections 2 and 3, a brief literature review and a description of the considered planning problem lay out the cornerstones for discussing a mathematical formulation of a Railroad Blocking Problem (RBP) of specific interest to Deutsche Bahn AG (DB

AG). In section 4, an unconventional ant colony optimization approach is introduced together with a brief description of another heuristic currently in use for creating blocking plans at DB AG. In section 5, we perform computational tests on a variety of large-scale, real-world instances and compare the performance of the two heuristic approaches and an exact solver. Concluding remarks are given in section 6.

2 Literature Review

The RBP belongs to the class of Network Design Problems (NDPs) and Multi-Commodity Flow Problems (MCFPs), it is described by [2] and [3] in its most basic formulation. An extensive literature review of these problems is beyond the scope of this paper, we refer to [4] for an overview. Ref. [5] extends the RBP formulation and introduces the Flow Car Routing Problem (FCRP) that is currently used at DB AG. The FCRP considers additional constraints, such as capacities at stations and hierarchical movement rules. Ref. [5] proposes the rip-up and reroute (RR) heuristic to solve the problem, when applied to small and medium-sized, real-world problems the heuristic produces good results in a comparatively short time frame. Improvements to the formulation of the FCRP are provided by [6] with a focus on exact and heuristic cuts. The problem is extended by non-linear turnover time constraints, which are then linearized. Since then, the FCRP has slightly evolved to consider balancing constraints, train capacities at stations and a maximum number of classifications for a commodity, all of which are presented in section 3.

Methods with a focus on exact and heuristic solutions have been devised to solve the RBP, see [7–9]. Ref. [10] presents a non-linear variant of the RBP with direct train deliveries and introduces a conventional ant system enhanced by a variable evaporation rate. The authors claim to produce results comparable to exact approaches with an underlying network of 20 stations. Ref. [11] solves a RBP with yard and classification track constraints for networks of up to 100 nodes with a traditional ACO heuristic and obtain competitive results compared to CPLEX software. However, none of the above focus on instances of scale and restrictiveness comparable to that of the FCRP studied in this paper. Concerning the lack of appropriate methods for the problem presented here, we propose an ACO approach utilizing a local and a global pheromone as well as distinct pheromone graphs for each commodity, in order to solve moderately and highly constrained real-world NDPs. Extensions of and differences to traditional ACO algorithms are discussed in subsection 4.2.

3 Flow Car Routing Problem

Rail freight services for commercial cargo transportation can be categorized into two main groups: unit train transportation, where customers book a whole train for the transportation of their goods, and single car transportation, where customers book a single or multiple railroad cars for commodities to be transported from their respective origin to destination. At DB AG, around 100 billion ton-kilometers are shipped in freight service annually, thereof single car transportation makes up the majority of cargo business and also gives rise to a complex planning problem. In the early stages of this planning process, a succession of stations has to be specified for every railroad car, i.e. a blocking plan. At stations where several cars meet they can be consolidated to form a block of cars. This means that they are grouped together, attached to a locomotive and travel as a unit to their next (intermediate) station without being handled in between. Only then, in later stages of the planning process, exact routes for these blocks through the physical railway network have to be generated, before determining timetables and crew schedules. These later stages are not the focus of this paper. Connections between two stations cannot necessarily be interpreted as physical tracks.

Let $\mathcal{V}$ and $\mathcal{A}$ denote the set of stations and relations of the underlying railway network, respectively, then $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ describes a complete directed graph abstracting the network. The relations $\mathcal{A}$ are merely logical links, meaning that trains assigned to arc (i, j) are regrouped or (dis)assembled at i and j, and do not indicate any obligation to traverse a direct, actual track between i and j in later stages of the planning process. Incoming and outgoing arcs of node i are described by δ_i^- and δ_i^+, respectively. Additionally, we introduce $\mathcal{K}$ which describes the set of commodities k that have to be routed from their origin $\sigma^k \in \mathcal{K}^+$ to their destination $\tau^k \in \mathcal{K}^-$, where $\mathcal{K}^+, \mathcal{K}^- \subseteq \mathcal{V}$ describe the sets of all origin and destination nodes. A commodity occupies space on one or multiple cars depending on the physical size and weight of the commodity. Cars are then grouped together and attached to a locomotive, the resulting vehicle is called a train. Classification yards are assigned to different hierarchical levels $\gamma_i \in \{0, 1, 2, 3\}$, where a lower level corresponds to a bigger and less prevalent station. Rules for classification are imposed which state that cars may descend $((i, j): \gamma_i > \gamma_j)$ until they either stay or ascend, thereafter they can only continue to ascend in the hierarchical structure $((i, j): \gamma_i < \gamma_j)$ [5]. Another important term is the uniqueness restriction, it requires all commodities with the same destination to share their route once they are both classified at the same station [12]. In practice the uniqueness restriction is not always relevant, which is why, in section 5, we perform computations without that very constraint.

In [5] the problem formulation for the above mentioned complete directed graph structure is given. It is then reformulated, s.t. the constraints that impose the hierarchical movement rules become obsolete and are factored in by design. This is achieved by transforming the original graph $\mathcal{G}$ into a hierarchical graph $\mathcal{G}'$. The transformation works by multiplying nodes and adding arcs, resulting in a directed acyclic graph (DAG), and requires a mapping of the flow variables of $\mathcal{G}$ to the edges of $\mathcal{G}'$. The hierarchical transformation guarantees feasibility w.r.t. the hierarchy constraints. This enables, otherwise inconceivable, preprocessing steps, which result in significantly shorter computational times for solvers and are beneficial for heuristic solution approaches. To obtain a valid solution the mapping must be inverted after solving the model in the post processing, s.t. the hierarchical edges map to the original edges. As it is not the subject of this paper, the mapping itself and the necessary changes induced on the model are given without further explanation, for details please refer to [5]. For the convenience of modeling we create an auxiliary graph $\mathcal{H}_k'$ from $\mathcal{G}'$ for every commodity k. The graph $\mathcal{H}_k'$ is used to build the mixed-integer program and can be used for the heuristics as well. The mapping X of arc $(i,j) \in \mathcal{A}$ of k to the corresponding arcs of the hierarchical auxiliary graph $\mathcal{H}'_k = (\mathcal{V}'_k, \mathcal{A}'_k)$ is in the form of $X\colon \mathcal{A} \times \mathcal{K} \to \mathbb{B}\colon (i,j,k) \mapsto x^k_{i',j'}$, with $(i',j') \in \mathcal{A}'_k$. Arcs for which $(i',j') \in \{\mathcal{A}'_k\colon i' = j'\}$ applies are denoted as $\mathcal{A}^t_k$. The modified problem can now be formulated as:

$$\min \textstyle\sum_{(i,j)\in\mathcal{A}} (c^{\text{loco}}_{ij} n_{ij} + c^{\text{track}}_i y_{ij}) + \sum_{k\in\mathcal{K},(i,j)\in\mathcal{A}'_k\setminus\mathcal{A}^t_k} (c^{\text{switch}}_i \eta^k x^k_{ij} + c^{\text{car}}_k \eta^k \theta_{ij} x^k_{ij}) + \sum_{k\in\mathcal{K}} c^{\text{penalty}}_k a_k \quad (1)$$

s.t.

$$\textstyle\sum_{(j,i)\in\delta^-_i} x^k_{ji} - \sum_{(i,j)\in\delta^+_i} x^k_{ij} = \begin{cases} 1 - a_k, & \text{if } i = \tau^k, \\ -1 + a_k, & \text{if } i = \sigma^k, \\ 0, & \text{otherwise,} \end{cases} \quad \forall k \in \mathcal{K}, \forall i \in \mathcal{V}'_k, \quad (2)$$

$$\textstyle\sum_{k\in\mathcal{K}} \lambda^k X(i,j,k) \le L_{ij} n_{ij}, \forall (i,j) \in \mathcal{A}, \quad (3)$$

$$\textstyle\sum_{k\in\mathcal{K}} \omega^k X(i,j,k) \le W_{ij} n_{ij}, \forall (i,j) \in \mathcal{A}, \quad (4)$$

$$-T_{ij} + 1 \le n_{ij} - T_{ij} y_{ij} \le 0, \forall (i,j) \in \mathcal{A}, \quad (5)$$

$$\textstyle\sum_{(j,i)\in\delta^-_i} y_{ji} \le Y^-_i, \forall i \in \mathcal{V}, \quad (6)$$

$$\textstyle\sum_{(i,j)\in\delta^+_i} y_{ij} \le Y^+_i, \forall i \in \mathcal{V}, \quad (7)$$

$$\textstyle\sum_{k\in\mathcal{K}\setminus(\mathcal{K}^+_i\cup\mathcal{K}^-_i)} \sum_{v\in\{i_{\text{u}},i_{\text{m}},i_{\text{d}}\}} \sum_{(v,j)\in\delta^+_v\setminus\mathcal{A}^t_k} \eta^k x^k_{vj} \le E_i, \forall i \in \mathcal{V}, \quad (8)$$

$$X(j,i,k) \le 3 s_i, \forall i \in \mathcal{V}, \forall (j,i) \in \delta^-_i, \forall k \in \mathcal{K}\setminus(\mathcal{K}^+_i \cup \mathcal{K}^-_i), \quad (9)$$

$$X(i,j,k) \le 3 s_i, \forall i \in \mathcal{V}, \forall (i,j) \in \delta^+_i, \forall k \in \mathcal{K}\setminus(\mathcal{K}^+_i \cup \mathcal{K}^-_i), \quad (10)$$

$$\textstyle\sum_{(j,i)\in\delta^-_i} n_{ji} \le N^-_i s_i, \forall i \in \mathcal{V}, \quad (11)$$

$$\textstyle\sum_{(i,j)\in\delta^+_i} n_{ij} \le N^+_i s_i, \forall i \in \mathcal{V}, \quad (12)$$

$$X(i,j,k) \le z^d_{ij}, \forall (i,j,k) \in \mathcal{A} \times \mathcal{K}, d = \tau^k, \quad (13)$$

$$\textstyle\sum_{(i,j)\in\delta^+_i} z^d_{ij} \le 1, \forall i \in \mathcal{V}, d \in \mathcal{K}, \quad (14)$$

$$\textstyle\sum_{(i,j)\in\mathcal{A}'_k\setminus\mathcal{A}^t_k} \theta_{ij} x^k_{ij} \le H^k, \forall k \in \mathcal{K}, \quad (15)$$

$$\textstyle\sum_{(i,j)\in\mathcal{A}'_k\setminus\mathcal{A}^t_k} x^k_{ij} \le G^k + 1, \forall k \in \mathcal{K}, \quad (16)$$

$$y_{ji} = y_{ij}, \forall i \in \mathcal{V} \cap (\mathcal{K}^+ \cup \mathcal{K}^-), \forall (i,j) \in \{\mathcal{A}\colon (j,i) \in \delta^-_i \wedge (i,j) \in \delta^+_i\}, \quad (17)$$

$$x^k_{ij} \in \mathbb{B}, k \in \mathcal{K}, \forall (i,j) \in \mathcal{A}'_k, \quad (18)$$

$$z^d_{ij} \in \mathbb{B}, \forall (i,j) \in \mathcal{A}, d \in \mathcal{K}^-, \quad (19)$$

$$a_k \in \mathbb{B}, k \in \mathcal{K}, \tag{20}$$

$$s_i \in \mathbb{B}, i \in \mathcal{V}, \tag{21}$$

$$n_{ij} \in \mathbb{Z}_{\geq 0}, \forall (i,j) \in \mathcal{A}, \tag{22}$$

$$y_{ij} \in \mathbb{Z}_{\geq 0}, \forall (i,j) \in \mathcal{A}. \tag{23}$$

Boolean decision variables that indicate whether a commodity k travels along relation (i,j), $x_{ij}^k = 1$ or not $x_{ij}^k = 0$, are introduced to the model. The variables a_k describe whether a shipment k is sent at all, and s_i indicate whether a station is in use. The number of trains that are needed to transport all commodities that travel on any relation is given by n_{ij}. Variables y_{ij} give the number of classification tracks that must be used at stations i and j. The train and classification track variables can both take positive integer values, meaning that multiple trains and tracks might be assigned to any relation. To comply with the uniqueness restriction, the variables z_{ij}^d are introduced and describe whether any commodity destined for node d travels along arc (i,j). The variables x_{ij}^k in equations (2) constitute the flow variables and guarantee that any commodity k entering node i also leaves that node again, except for node i being either the origin σ^k or destination τ^k of k. The variables a_k need to be added to allow for unrouted commodities. This is of interest for solving real-world instances because the capacity of a given network may not be sufficient to transport all commodities.

Let the length and weight of commodity k be represented by λ^k and ω^k, respectively, then (3) and (4) ensure that the maximum length L_{ij} and weight W_{ij} restrictions are not violated by allocating the necessary number of trains n_{ij} to that relation. The train variables n_{ij} are linked to the track variables y_{ij} by (5) and determine the number of classification tracks which must be reserved at both stations i and j, in order to accommodate a block without exceeding the train capacity T_{ij} of a single track. Constraints (6) and (7) bound the number of tracks that are available at station i by the maximum number of incoming Y_i^- and outgoing Y_i^+ tracks. Equations (8) ensure that the number of cars η^k of all commodities that pass station i does not violate the car switch capacity E_i at that station. Which stations are part of a solution is determined by (9) and (10). The maximum number of incoming N_i^- and outgoing N_i^+ trains that can be handled at any station are limited by (11) and (12), respectively. Constraints (15) restrict any path a commodity takes to be less or equal to the maximum transport time H^k. The transport time θ_{ij} is calculated as the sum of the departure processing time θ_{ij}^{dep}, the travel time $\theta_{ij}^{\text{travel}}$ and the arrival processing time θ_{ij}^{arr}, i.e. $\theta_{ij} = \theta_{ij}^{\text{dep}} + \theta_{ij}^{\text{travel}} + \theta_{ij}^{\text{arr}}$. Further, commodities must not be classified more often than a predefined number G^k, i.e. the maximum number of hops through the network, which is cared for by (16). In order to balance the load on the stations the constraints (17) are introduced. For every destination d, z_{ij}^d describes an arc in the rooted tree with root d. Equations (13) link the flow variables x_{ij}^k with the uniqueness variables z_{ij}^d, and (14) guarantee the tree structure.

Given the above set of constraints, the objective is to minimize the overall costs which can be broken down into two major assets. First, there are costs incurred for actually moving a commodity, i.e. c_{ij}^{loco} for every locomotive of a train deployed on a relation, as well as c_k^{car} for hauling a single car of commodity k multiplied by the transport time θ_{ij}. Not delivering commodity k is penalized with c_k^{penalty}. The second group of costs can be attributed to managing the incoming and outgoing vehicles at stations. It is calculated as the costs for reserving a classification track c_i^{track} at station i plus the costs c_i^{switch} for classifying a single car of shipment k.

4 Heuristic Approaches

At DB AG two heuristic approaches are used to tackle this problem because solvers alone cannot produce satisfactory results within a reasonable amount of time (see section 5). Both heuristics follow a common strategy in the sense that they, sequentially, choose a path for each commodity k on a dedicated routing graph $\mathcal{R}_k'$. The routing graph is generated from the auxiliary graph $\mathcal{H}_k'$ by eliminating all infeasibilities regarding constraints (3) – (14) before any one commodity is routed. Feasibility w.r.t. the remaining constraints cannot be ensured with preprocessing and must be taken care of by other mechanisms. A rough outline to the RR heuristic is given in the following, the basic mechanisms as well as a measure of complexity will be conveyed. The ACO heuristic is discussed in greater detail thereafter.

4.1 Rip-up and Reroute

On a higher level, the RR heuristic is divided into a constructive and an improvement stage, both stages exhibit a simulated annealing strategy [5]. The RR heuristic implements a ripping and routing core as depicted in Figure 1. In the routing core, the commodities $k \in \mathcal{K}_\text{u}$ that have to be routed are sorted according to a set of dynamically determined priority scores and are then, one after another, routed utilizing a shortest path algorithm. The scores are designed to factor in constraints (15) – (17) and allow for feasible solutions. Because the underlying graph $\mathcal{R}_k'$ is a DAG we can utilize the DAG algorithm, which runs in $\mathcal{O}(|\mathcal{V}| + |\mathcal{A}|)$ as compared to the runtime complexity $\mathcal{O}\big((|\mathcal{V}| + |\mathcal{A}|)\log|\mathcal{V}|\big)$ of the A^* algorithm. If the path obtained by the DAG algorithm is feasible w.r.t. constraints (15) – (17), then commodity k is assigned to the set of successfully routed commodities $\mathcal{K}_\text{s}$; to the set of the failed commodities $\mathcal{K}_\text{f}$ otherwise. If all unrouted commodities are processed ($\mathcal{K}_\text{u} = \emptyset$) we move on to the next iteration. In the ripping core, a variable ratio ϕ of $\mathcal{K}_\text{s}$ is ripped, i.e. removed from the solution, in order to be rerouted in addition to $\mathcal{K}_\text{f}$. Which of the commodities are ripped depends, again, on a dynamically chosen set of the aforementioned scores and the resulting sorting. The selection mechanism for choosing the scores is dependent on the stage of the heuristic and a multitude of other factors. The ratio ϕ depends on the stage of the heuristic and the temperature of the underlying simulated annealing strategy.

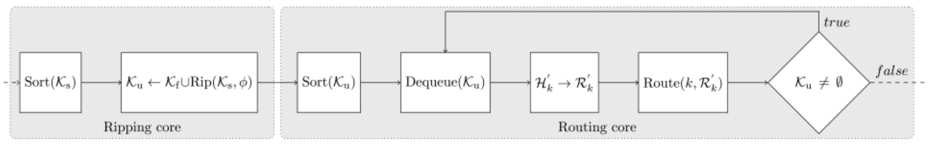
Sort($\mathcal{K}_s$)
$\mathcal{K}_u \leftarrow \mathcal{K}_f \cup \text{Rip}(\mathcal{K}_s, \phi)$
Ripping core
Sort($\mathcal{K}_u$)
Dequeue($\mathcal{K}_u$)
$\mathcal{H}'_k \to \mathcal{R}'_k$
Route($k, \mathcal{R}'_k$)
$\mathcal{K}_u \neq \emptyset$
true
false
Routing core

Figure 1. Flowchart of the routing and ripping core

For performance reasons, the objective value is evaluated only if the ratio $|\mathcal{K}_\mathrm{f}|/|\mathcal{K}_\mathrm{s}|$ exceeds a given threshold or a threshold set by the best solution found so far. The cost function, which is handed to the shortest path algorithm, is derived from the objective function (1) and is given as

$$c_{ij}^{\mathrm{RR}} = c_k^{\mathrm{car}} \eta^k \theta_{ij} + c_i^{\mathrm{switch}} \eta^k + c_{ij}^{\mathrm{loco}} n_{ij}(\eta^k, \mathcal{K}_\mathrm{s}) + c_i^{\mathrm{track}} y_{ij}(\eta^k, \mathcal{K}_\mathrm{s}). \tag{24}$$

The locomotive $c_{ij}^{\mathrm{loco}} n_{ij}(\cdot)$ and track $c_i^{\mathrm{track}} y_{ij}(\cdot)$ costs that are incurred by commodity k are strongly dependent on the set of hitherto successfully routed commodities $\mathcal{K}_\mathrm{s}$. This imposes an additional level of complexity when routing sequentially. The required number of trains $n_{ij}(\cdot)$ and tracks $y_{ij}(\cdot)$ is determined according to constraints (3), (4) and (5).

4.2 Ant Colony Optimization

From here on, we present the main contributions of this paper by introducing and applying two different ant systems to the studied problem. The ACO-P system is a conventional, probabilistic ACO algorithm, whereas ACO-D is not an inherently probabilistic system. Traditional ACO algorithms, e.g. as described by [13], work by modeling the food foraging behavior of real ant colonies, whereby ants traverse an area around their anthill continuously depositing pheromones. These pheromones evaporate over time and, eventually, some paths exhibit a higher pheromone concentration than others and are therefore more attractive for ants to follow. Ideally, this mechanism results in short paths to food sources, even though under certain circumstances quite the opposite has been observed in nature.[1] Inspiration for the ACO systems presented here is mainly taken from [10], [11] and [14].

Interpreting conventional ACO algorithms and applying them here, we model a colony as an amalgamation of ants, where a single ant represents a specific commodity. Each ant maintains separate pheromone values π_{ij}^k on the arcs of a distinct pheromone graph $\mathcal{P}'_k = (\mathcal{V}'_k, \mathcal{A}'_k)$ [14]. The pheromones are meant to maintain favorability values of edges of commodity paths over time. Additionally, a metric is defined that calculates the visibility value μ_{ij}^k for a given edge, and is representative of the costs incurred by traversing a relation. The edge to visit is then chosen according to a roulette wheel selection with the probabilities given by

[1] The phenomenon is known as an ant mill and causes ants to die of exhaustion after following a circular pheromone trail for an extended period of time.

$$\rho_{\mathrm{ij}}^{k} = \frac{\left(\pi_{ij}^{k}\right)^{\alpha}\left(\mu_{ij}^{k}\right)^{\beta}}{\sum_{(i,v)\in\delta_i^+}\left(\pi_{iv}^{k}\right)^{\alpha}\left(\mu_{iv}^{k}\right)^{\beta}}, \tag{25}$$

with

$$\mu_{ij}^{k} \coloneqq \left(c_{ij}^{\mathrm{RR}}(k)\right)^{-1}. \tag{26}$$

The exponents α and β are used to specify the influence of the two components. In the implementation presented here, we use two kinds of pheromones, a local and a global pheromone [11]. Their weighted sum is the pheromone value used for guiding the search and is given by

$$\pi_{ij}^{k} = (1-\varrho)\pi_{ij}^{k,\mathrm{local}} + \varrho\pi_{ij}^{k,\mathrm{global}}, \tag{27}$$

where $\varrho \in [0;1]$ controls the influence that is attributed to the global pheromone. Accordingly, $1 - \varrho$ is the ratio the local pheromone enters into the equation. Generally, the idea behind those two pheromones is to have the global pheromone direct the search towards promising regions of the search space and act as a long-term memory. Local pheromones are used as a short-term memory and allow for the exploration of the more immediate neighborhood.

The natural evaporation of pheromones is modeled and applied when updating the pheromone trail to counter premature convergence to a suboptimal solution. Different pheromone update strategies have been devised in the literature, one of the most common is the following, where ϵ denotes the evaporation rate of the pheromone:

$$\pi_{\mathrm{ij}}(t+1) = (1-\epsilon)\pi_{ij}(t) + \epsilon F(s). \tag{28}$$

Suppose the objective is to minimize a function $f(s)$, then $F(s)$ describes the fitness of a given solution s where $f(s) \leq f(s')$ implies $F(s) \geq F(s')$. The specifics of $F(s)$ are highly dependent on the underlying problem structure, the ant system and the parametrization thereof. The ant system presented here employs a pheromone update strategy similar to equation (28), with the variation that the global pheromones are only updated when a new best solution is found. The local pheromones are updated after routing a single commodity. Both updates only apply to edges that are part of the current solution. The set of edges that form the path of commodity k in a solution is denoted as $\mathcal{S}_k \subseteq \mathcal{P}_k'$. The evaporation rate of the global pheromone ε is different to that of the local pheromone ϵ, in order to be able to control the influence of the short- and long-term memory separately. The fitness $F(s)$ of a solution is calculated as the ratio of the best-known solution s^{best} and the current solution s^{current}. The update function (28) in combination with this fitness function has the positive side effect of limiting the pheromone values to be in the range of $[0;1]$, what is also known as the $\mathcal{MAX}$-$\mathcal{MIN}$ ant system. With the ant system and pheromone strategies explained above, the update functions can be formulated as:

$$\pi_{ij}^{k,\mathrm{local}}(t+1) = \begin{cases} (1-\epsilon)\pi_{ij}^{k,\mathrm{local}}(t) + \epsilon\frac{f(s^{\mathrm{best}})}{f(s^{\mathrm{current}})}, & \text{if } (i,j) \in \mathcal{S}_k^{\mathrm{current}}, \\ \pi_{ij}^{k,\mathrm{local}}(t), & \text{otherwise}, \end{cases} \tag{29}$$

$$\pi_{ij}^{k,\text{global}}(t+1) = \begin{cases} (1-\varepsilon)\pi_{ij}^{k,\text{global}}(t) + \varepsilon \frac{f(s^{\text{best}})}{f(s^{\text{current}})}, \text{if } f(s^{\text{current}}) \leq f\left(s^{\text{best}}\right) \\ \qquad\qquad\qquad\qquad\qquad\qquad \wedge\, (i,j) \in \mathcal{S}_k^{\text{current}}, \\ \pi_{ij}^{k,\text{local}}(t), \qquad\qquad\qquad\qquad\qquad \text{otherwise.} \end{cases} \quad (30)$$

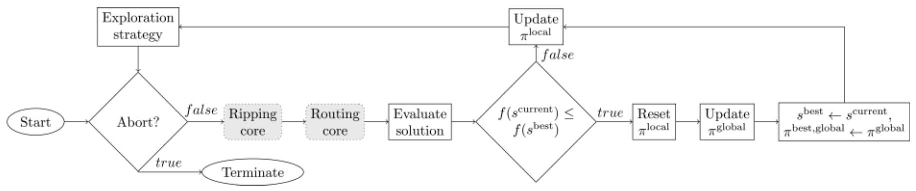
Exploration strategy
Update π^{local}
$false$
Start
Abort?
$false$
Ripping core
Routing core
Evaluate solution
$f(s^{\text{current}}) \leq f(s^{\text{best}})$
$true$
Reset π^{local}
Update π^{global}
$s^{\text{best}} \leftarrow s^{\text{current}}$, $\pi^{\text{best,global}} \leftarrow \pi^{\text{global}}$
$true$
Terminate

Figure 2. Flowchart of the ACO algorithm

The flowchart in Figure 2 depicts the routing procedure of the ant system as described previously. The ripping and routing core are basically the same as depicted in Figure 1. The difference to the RR heuristic is that ϕ is constant and the commodities do not have to be sorted before ripping as the ripping happens in a first-in-first-out fashion, i.e. in the order in which the commodities have been routed in the preceding iteration. After ripping, the commodities undergo a random shuffle instead of a sorting by scores, before the sequential choice of arcs takes place according to equations (25) and (26). The evaluation of the objective function is performed in every iteration. An exploration strategy is triggered if there is no improvement in the objective value for a certain number of iterations. This strategy reduces all π^{local} to its initial value and resets π^{global} to the global pheromones $\pi^{\text{best,global}}$ of the best solution found so far. With regards to constraints (15) – (17), apart from the pheromones, there is no mechanism to ensure feasibility.

Initial runs on the test instances employing ACO-P produce unsatisfactory results on all scenarios. Knowing the RR heuristic performs better by utilizing a shortest path algorithm, we now defer from probabilistically choosing edges one after another and, instead, apply the DAG algorithm to obtain complete origin-destination paths for single commodities. Thus, equations (25) and (26) become obsolete, as the use of the DAG algorithm implies a path selection scheme as opposed to an edge selection scheme. The associated cost function is obtained by weighting equation (24) with the pheromone values and results in

$$c_{ij}^{\text{ACO}}(\text{k}) = \left(\frac{1}{\pi_{ij}^{k}}\right)^{\alpha} \left(\mu_{ij}^{k}\right)^{\beta}, \quad (31)$$

with

$$\mu_{ij}^{k} \coloneqq c_{ij}^{\text{RR}}(k). \quad (32)$$

Due to the utilization of a shortest path algorithm, we obtain a non-probabilistic ant system ACO-D. In this implementation, though, shuffling the commodities before the actual routing adds a random component.

5 Computational Results

Tests are executed with variations of the formulation discussed in section 3, i.e. different sets of constraint that are active. This is done to assure good performance on a wider range of problems. Twelve test instances are derived from four different real-world networks of DB AG, as given in Table 1. The name of the instance corresponds to the number of commodities that have to be routed. The letter following the name denotes the variant, i.e. the set of constraints and decision variables from the problem formulation in section 3. The four different scenarios comprise 338, 638, 1129 and 7334 commodities with $|\mathcal{V}| = 46$, 177, 177 and 177 nodes, respectively. The number of edges of each scenario, before preprocessing and the hierarchical transformation, is given by $|\mathcal{A}| = |\mathcal{V}|(|\mathcal{V}| - 1)$. Variant a is a moderately constrained NDP, i.e. constraints (1) – (7), (18), (20), (22) apply and $y_{ij} \in \mathbb{B}$. Variant b is the same model as described in section 3 but without the uniqueness constraints (13), (14) and (19). Finally, variant c comprises all constraints and decision variables.

The ratio of the commodities that are to be ripped is fixed to $\phi = 0.7$. The global pheromone ratio is chosen to be $\varrho = 0.5$. The exponent of the pheromones α and of the visibility β are both set to 1.0. The local and global pheromone evaporation rates take the values $\epsilon = 0.05$ and $\varepsilon = 0.65$, respectively. The parameter ε is chosen to be greater than ϵ, as to allow for bigger leaps in the search space once a new best solution is found, and not to get trapped in local optima. The initial value of the local pheromones is set to 0.0, and for the global pheromones it is set to 0.5. The threshold for triggering an exploration strategy is set to 100 iterations without objective improvement. The time limit for each run is set to eight hours (without preprocessing time for the hierarchical transformation) and the maximum number of iterations performed by any of the heuristics is set to 1000. All computations are performed on two cores of a XEON E5v2 processor on a system with 176 GB of RAM operating Windows with CPLEX 12.1.

Table 1. Computational results of CPLEX vs. RR vs. ACO-D

	CPLEX				RR				ACO-D		
Name	Time (s)	Obj	Gap (%)	Δ (%)	Time (s)	Min	Avg	Δ (%)	Time (s)	Min	Avg
338a	1890	$^*\mathbf{5.95 \cdot 10^6}$	0.10	−3.72	42	$6.00 \cdot 10^6$	$6.08 \cdot 10^6$	−1.49	73	$6.03 \cdot 10^6$	$6.18 \cdot 10^6$
338b	28800	$^*\mathbf{1.69 \cdot 10^7}$	0.30	−13.28	13	$1.90 \cdot 10^7$	$1.96 \cdot 10^7$	0.27	31	$1.88 \cdot 10^7$	$1.95 \cdot 10^7$
338c	28800	$^*\mathbf{1.70 \cdot 10^7}$	0.86	−22.34	19	$1.95 \cdot 10^7$	$2.01 \cdot 10^7$	−8.06	43	$2.13 \cdot 10^7$	$2.19 \cdot 10^7$
638a	28800	$^*\mathbf{2.57 \cdot 10^6}$	10.38	−2.83	288	$2.61 \cdot 10^6$	$2.72 \cdot 10^6$	2.85	499	$2.61 \cdot 10^6$	$2.64 \cdot 10^6$
638b	28800	$2.93 \cdot 10^6$	21.15	8.64	323	$2.71 \cdot 10^6$	$2.79 \cdot 10^6$	3.75	559	$^*2.63 \cdot 10^6$	$\mathbf{2.69 \cdot 10^6}$
638c	28800	$3.25 \cdot 10^6$	32.25	18.70	1113	$^*2.61 \cdot 10^6$	$2.84 \cdot 10^6$	3.50	1501	$2.65 \cdot 10^6$	$\mathbf{2.74 \cdot 10^6}$
1129a	28800	$3.87 \cdot 10^6$	24.34	3.08	442	$3.70 \cdot 10^6$	$3.82 \cdot 10^6$	1.92	784	$^*3.60 \cdot 10^6$	$\mathbf{3.75 \cdot 10^6}$
1129b	28800	$3.81 \cdot 10^6$	24.51	3.33	420	$3.76 \cdot 10^6$	$3.84 \cdot 10^6$	3.98	704	$^*3.63 \cdot 10^6$	$\mathbf{3.69 \cdot 10^6}$
1129c	28800	$4.17 \cdot 10^6$	31.93	9.82	511	$3.77 \cdot 10^6$	$3.91 \cdot 10^6$	3.06	803	$^*3.66 \cdot 10^6$	$\mathbf{3.80 \cdot 10^6}$
7334a	28800	*7.92 · 10⁶*	–	–	1751	$1.70 \cdot 10^7$	$1.76 \cdot 10^7$	12.50	4978	$^*1.54 \cdot 10^7$	$\mathbf{1.57 \cdot 10^7}$
7334b	28800	*7.76 · 10⁶*	–	–	2382	$1.62 \cdot 10^7$	$1.72 \cdot 10^7$	8.43	5297	$^*1.55 \cdot 10^7$	$\mathbf{1.58 \cdot 10^7}$
7334c	28800	*7.33 · 10⁶*	–	–	7843	$1.77 \cdot 10^7$	$\mathbf{1.83 \cdot 10^7}$	−0.49	16402	$^*1.69 \cdot 10^7$	$1.84 \cdot 10^7$

CPLEX gets invoked once whereas the RR and ACO-D heuristics perform 20 runs on each variant, the results are given in Table 1. For the heuristics, the mean computational time until a stopping criterion is met is given, followed by the minimum (Min) and average (Avg) objective values. The percentage difference Δ of the objective is given, where the reference value is the average objective value of the ACO-D variant.

The CPLEX column exhibits the computational time of a single run until either the time limit of eight hours or a gap of 0.1% is reached. Entries where only the objective value is given represent the lower bound, as no feasible solution could be found within the given time limit. The lowest average objective value is set in bold, the lowest objective value overall is marked with an asterisk. The table reveals the dominance of ACO-D over the RR heuristic on almost all instances, though the ant system needs significantly longer computational time until it reaches an abort criterion. CPLEX outperforms both heuristics on small instances and is naturally less competitive on bigger ones, up to not being able to obtain a feasible solution for the biggest scenario, where ACO-D performs substantially better than the rest. The greater Δ values of the heuristics compared to the solver on instances 338b and 338c can be explained by the penalization of unrouted commodities in the objective function.

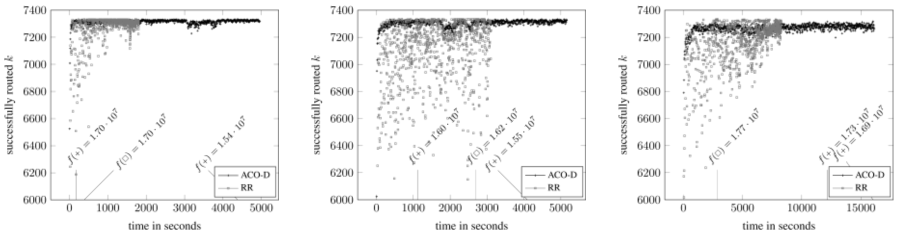

Figure 3. Computational results of scenario 7334, variants (f.l.t.r.) a, b and c: ACO-D vs. RR

Figure 3 compares the RR and ACO-D heuristic w.r.t. their performance and runtime, the best of the 20 runs of both heuristics on the largest scenario are depicted. Because the RR heuristic is designed to not evaluate the objective function in most iterations, the number of successfully routed commodities is plotted against the computational time. The best objective values of both heuristics are highlighted, as well as the point in time at which the inferior solution is surpassed by the competing heuristic. It takes marginally longer for ACO-D to find competitive results on small instances, but it then outpaces the RR heuristic on larger ones and better solution are obtained significantly faster.

6 Conclusion

In contrast to conventional ACO algorithms the path construction is not accomplished with a sequential and probabilistic choice of edges, instead, appropriate shortest path algorithms are utilized. The assessment of the quality of the chosen paths, w.r.t. the routing order and resulting objective value, is implicitly carried out via the pheromones. The proposed ACO-D system produces satisfactory results on a broad range of instances and outperforms the problem-specific RR heuristic and CPLEX on large, highly constrained instances. In principle, the studied ACO concept is not limited to the problems presented here, as we demonstrate applicability to differently sized problem instances and levels of restrictiveness. In order to further verify the applicability of the

ACO-D system to conventional NDPs and MCFPs more computational tests are needed, e.g. on available benchmark problems.

7 Acknowledgments

The author would like to thank everyone at DB Analytics (DS.VO) for the welcoming atmosphere in the office. Thanks to Prof. Udo Buscher for suggesting the organization, and to the department head of logistics Dr. Jörg Wolfner for making the internship possible. Special thanks go to Hanno Schülldorf and Dr. Christine Hayn for their supervision despite a crammed schedule. Thanks to Marc Johann for offering guidance through the extensive code base, and to Dr. Christian Ullerich for fruitful discussions with jokes on the side. A special token of appreciation goes to the fellow interns Marie Leonhardt and Tim Kipshagen for making the internship particularly enjoyable.

References

1. Assad, A. A.: Models for rail transportation. Transportation Research Part A: General, 14, 205–220 (1980)
2. Newton, H. N., Barnhart, C., Vance, P. H.: Constructing railroad blocking plans to minimize handling costs. Transportation Science, 32, 330–345 (1998)
3. Bodin, L. D., Golden, B. L., Schuster, A. D., Romig, W.: A model for the blocking of trains. Transportation Research Part B: Methodological, 14, 115–120 (1980)
4. Crainic, T. G., Laporte, G.: Planning models for freight transportation. European Journal of Operational Research, 97, pp. 409–438 (1997)
5. Homfeld, H.: Consolidating car routes in rail freight service by discrete optimization. Verlag Dr. Hut, München (2012)
6. Fügenschuh, A., Homfeld, H., Schülldorf, H.: Single-car routing in rail freight transport. Transportation Science, 49, 130–148 (2015)
7. Ahuja, R. K., Jha, K. C., Liu, J.: Solving real-life railroad blocking problems. Interfaces, 37, 404–419 (2007)
8. Costa, A. M.: A survey on benders decomposition applied to fixed-charge network design problems. Computers & Operations Research, 32, 1429–1450 (2005)
9. Gabrel, V., Knippel, A., Minoux, M.: A comparison of heuristics for the discrete cost multicommodity network optimization problem. Journal of Heuristics, 9, 429–445 (2003)
10. Yue, Y., Zhou, L., Yue, Q., Fan, Z.: Multi-route railroad blocking by improved model and ant colony algorithm in real world. Computers & Industrial Engineering, 60, 34–42 (2011)
11. Yaghini, M., Foroughi, A., Nadjari, B.: Solving railroad blocking problem using ant colony optimization algorithm. Applied Mathematical Modelling, 35, 5579–5591 (2011)
12. Schülldorf, H.: Optimization at Deutsche Bahn: aspects and examples. In: Dagstuhl Seminar Proceedings, pp. 1–6. Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2009)
13. Dorigo, M., Stützle, T.: Ant colony optimization: overview and recent advances. In: Gendreau, M., Potvin, J.-Y. (eds.) Handbook of Metaheuristics, vol. 146, 227–264. Springer, New York (2010)
14. Li, X., Aneja, Y. P., Baki, F.: Ant colony optimization metaheuristic for single-path multicommodity network flow problems. Journal of the Operational Research Society, 61, 1340–1355 (2010)