A Multi-Objective Optimization Model for the Buildup Allocation at KLM Cargo

Using a local seach heuristic in a data driven application

Thomas Hultermans

TU Delft

A Multi-Objective Optimization Model for the Buildup Allocation at KLM Cargo

Using a local seach heuristic in a data driven application

by

Thomas Hultermans

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on January 28th, 2021.

Student number:	4306597
Project duration:	18-11-2019 – 28-01-2021
Thesis committee:	Dr. J.M. Hoekstra
	Dr. P.C. Roling
	Dr. S.Fazi
	Dr. A. Bombelli

TU Delft

Acknowledgements

Hereby I present my thesis titled "A Multi-Objective Optimization Model for the Buildup Allocation at KLM Cargo". This work was written as final part of my graduation from the Master's programme in Air Transport Operations at Delft University of Technology.

Commissioned by KLM, a study was carried out during an internship period from November 2019 to November 2020 into the workforce allocation process for one of the cargo handling phases at Schiphol Airport. This thesis presents a scheduling model on the basis of the current practice at KLM Cargo.

The internship period of the past year has not been without its hurdles. The Covid-19 pandemic had, and continues to have to this day, a large impact on the airline industry. This impacted the communication during the first chaotic months of the crisis. Nevertheless, the internship was extended in order for this project to be completed. Although working from home due to the pandemic made writing this thesis a little more difficult and time-consuming than anticipated, I fully support the results of this research.

I would like to thank Paul Roling and Alessandro Bombelli of the Delft University of Technology for their guidance. I would also like to thank everyone at KLM who put in any effort to help me in the process of my research, in particular Bas Schmidt and Victor Schwarz.

Lastly, I hope you enjoy reading this thesis and gain a little more insight into the supply chain operations inside the cargo hub of one of the world's largest airline companies.

Sincerely, Thomas Hultermans

Delft, January 2021

Contents

List of Figures

List of Abbreviations

AKE	Container Pallet
AWB	Air Way Bill
BB	Bijbouwer ULD
CCC	Cargo Control Center
CHM	Cargo Handling Manual
EDF	Earliest Deadline First
IATA	International Air Transport Association
ICA	Intercontinental
LDP	Lower Deck Pallet
LF	Leeg Fust
M-ULD	Mixed ULD
MDP	Main Deck Pallet
MTD	Moving Truck Dock
PCHS	Pallet Container Handling System
PFM	Performance Management
RFC	Ready For Carriage
RFF	Ready For Flight
SME	Subject Matter Expert
T-ULD	Through ULD
ULD	Unit Load Device

Introduction

After completing an internship and successfully passing all courses, the Master's thesis is the final work before obtaining the degree of Master of Science at the Faculty of Aerospace Engineering at Delft University of Technology. This project has been undertaken in collaboration with KLM Cargo, who saw a need in their business for a research project to conceptualize an improvement in their cargo hub operations. After some discussion between all parties to set up a clear goal and framework, an internship contract was signed and the project was started.

The research project was carried out at the department of Performance Management (PFM) at KLM Cargo, with periodic visits to the university. The office of PFM is located in the cargo warehouse which made it possible to be closely situated to the ongoing operations inside the warehouse. At PFM the performance indicators of all the processes occurring at the KLM Cargo Hub at Schiphol Airport are monitored. Data is collected to support these indicators and the department identifies, with the help of data analytics tools, what can be improved on the basis of operational data of past events. The aim is however, to have a more predictive role. In that way, with the help of more advanced tools and access to real-time data, difficult situations can be tackled early on and preferably avoided.

This research project has been designed in an effort to work towards such a predictive role. The problem at hand is a bottleneck in the cargo buildup phase where shift leaders spend much time on setting up out a good work schedule for an upcoming shift. Based on the tasks, available personnel and flight departure times, shift leaders write up schedules in the best way possible to avoid delayed shipments. The current approach is not only a cumbersome endeavour, it takes lots of time as well for all three daily shifts. A solution needed to be found to make better buildup schedules in a shorter time with which difficult situations could be identified early on.

A scheduling model has been designed that not only aims at saving time and preventing delays, but also distributing the workload more evenly among personnel. The latter contributes to a happier workforce that can have their breaks on time as a result of better planning. Delivering shipments in time is important from a business perspective, but fairly distributing work according to everyone's abilities should be as equally important. The model presented here allows KLM Cargo to not only improve their schedule, but provide their employees with a fair distribution of work that prevents working overtime or last minute reallocation of tasks.

One of the questions that was very unclear to everyone who looked at this problem before, was the determination of work in one of the shifts. Out of all the shipments that move through the Hub, what constitutes as work, what is not work, and how can a better classification of work be made? There were some estimates and results from prior studies executed by KLM themselves, but the analysis in this research uses an entirely different approach. In addition, the allocation of tasks has never been studied before for any of the cargo handling phases. There has never been a team assigned to come up with an improved way of doing so. This project has tried to give an answer to both of these questions and in that sense, it was a very unique project.

From academic standpoint, the goal of the Master's thesis is to acquaint oneself with the process of academic research, recognize how new research can add to the existing collection of scientific literature and experience writing their own work in the from of a scientific paper. With the help of expert supervision from both the university and KLM, it has led to this final thesis document containing the process and findings of this research.

This thesis report is organized as follows : In Part I, the scientific paper is presented. Part II contains the study of relevant literature that supports the research. Finally, in Part III, some additional results are presented.

I

Scientific Paper

A Multi-Objective Optimization Model for the Buildup Allocation at KLM Cargo

Using a local search heuristic in a data driven application

Thomas Hultermans
Department of Air Transport & Operations, Delft University of Technology, The Netherlands

***Abstract* — The buildup phase at KLM Cargo requires cumbersome planning actions because of a slow data handover at the warehouse and an uncertain initial state of the buildup process at the beginning of the shift. This study presents a model to improve scheduling of future buildup shifts according to flight departure times of KLM. The most important objectives are to minimize the delays and equally distribute the workload. For this purpose, a MILP model has been created. The problem is modeled as a scheduling problem with time windows. Next to the branch-and-bound method, an adapted tabu search is implemented as an alternative optimization technique. Both methods show adequate performance in terms of computation time and model convergence. With the help of a real-time data architecture, scheduling can now be standardized, take less time, incorporate more detailed information and in addition, bottlenecks can be identified early.**

***Keywords* —** Operations Research; Scheduling Problem; Time Windows; Tabu Search;

1 Introduction

Workforce scheduling problems arise whenever a performance index is dependent on the allocation of personnel. This class of problems is common in many fields, including the service industry. KLM Cargo, situated at Amsterdam Schiphol Airport is experiencing such a problem in their cargo handling operations. Cargo handling is carried out from the arrival of shipments at the landside until the loading of the cargo hold of the aircraft on the airside. The logistic chain of events to get a shipment on the right aircraft involves sorting and building up shipments inside the Cargo warehouse. The buildup process is the subject of this research, where hours per day are spent on allocating the large number of personnel to a large number of buildup tasks. The Cargo department plays an important role in transporting shipments in time to consignees The allocation of personnel to tasks is subject to strict deadlines and the performance with respect to these deadlines has an effect on the relations with KLM Cargo's clients.

At the moment, the workforce scheduling is done manually by team coordinators and shift leaders while simultaneously considering multiple sources of information. A new allocation for the buildup phase needs to be made before the start of each of the three daily shifts. Due to cumbersome planning, shipments miss their flights. Flights are sometimes even forgotten to be allocated and information can change up until the final minutes before the shift. This makes scheduling teams to tasks difficult and time consuming. As indicated by KLM Cargo themselves, it can take 0.5 to 1.5 hrs per shift to allocate and adjust the teams to their tasks.

The goal of the scheduling process is to prevent delays of shipments. In addition, the work should be distributed equally because it requires additional coordination effort to reallocate teams to other workstations later on during the shift. When employees notice they have time to spare, they could slow down while others may be working faster. This is also the reason why a previous (internal, unpublished) study at KLM Cargo about productivity did not give accurate results when trying to track individual workers. The best solution is to give everyone a similar work package. A model needs to be developed that determines an optimized staff allocation based on the flight schedule. To give a proper recommendation, it is important to show the workings of the model on real data from KLM Cargo.

There is no scheduling model in earlier research that faithfully describes and incorporates all the constraints and features of this problem of interest, i.e. assigning teams to ULD workstations in the warehouse of an air cargo supply chain Hub. This research fills up that gap by developing such a scheduling model.

The work schedules will be made on the basis of flights. All flows of cargo destined for outbound international flights are considered. As will be clear later on, the only freight building subject to this research is freight building 3. A forecast of the work for future shifts is not part of this thesis. The work will be estimated by assuming a real-time database connection with the Cargo Hub.
The remainder of this thesis is organized as follows. The next section reviews relevant work on scheduling problems and linear optimization. Section 3 gives an overview of the business activities inside the Cargo Hub and the current process of assigning work to employees will be reviewed. Section 4 will explain the data behind many assumptions used in the model and how the inputs are retrieved from the data sources. The model will be defined in Section 5, together with a heuristic method. After that the parameters are set and the model is validated Section 7. The last section is the conclusion of this work.

2 Related Work

In this section the state-of-the-art of resource allocation and scheduling is reviewed. As stated by Ernst et al. (2004): "Task assignments are often required when working shifts have been determined but tasks have not been allocated to individuals." (p. 9).

Assignment Problems

Assignment problems are about finding optimal pairings of agents and tasks depending on a suitability function. Kuhn (1955) developed the Hungarian algorithm as the first practical solution method to solve the classic assignment problem (AP). Pentico (2007) provides an overview of the many variations of the assignment problem.

This project classifies as a GAP since there are more flights than there are teams. Ravindran and Ramaswami (1977) presents the bottleneck AP. This particular problem is not aimed at minimizing the total costs, but minimizing the most costly assignment. Duin and Volgenant (1991) studied two variations of the bottleneck AP. One is minimizing the difference between the least and most costly assignment. The other minimizes the difference between the maximum and average assignment cost. An example is given of minimizing the idle time of simultaneously operating machines. These bottleneck problems showed a good example for balancing the workloads in this research. Mulla et al. (2016) shows how in addition to a bottleneck formulation, hard constraints for workload, utilization and productivity can be used while taking into account awareness of fairness among the employees.

Geetha and Nair (1993) studied a bi-criteria bottleneck AP. Next to the assignment cost, their model assumes an additional supervisory cost as a bottleneck objective. Scarelli and Narula (2002) consider the problem of assigning referees to football matches with multiple independent selection criteria by using (in)compatibility and priority indices their method is able to include indifference, preference and veto thresholds. Equivalently so at KLM Cargo, different teams posses different skills. Shen et al. (2003) selects appropriate individuals for tasks based on a multi-criteria assessment of individual suitability, social relationships and existing tasks. Next to workload balancing, it focuses on qualitative impacts including specialization of labor and job enrichment.

Scheduling Problems

Graham et al. (1979) presents an overview and introduces a classification of different machine scheduling problems. Uniform parallel machine scheduling assumes a job as a single operation and the processing time depends on the speed factor of each machine. Oyetunji (2009) discuss common performance measures in scheduling problems, like total completion time and makespan. Janiak and Kovalyov (1996) looks at resource dependent processing times, which is an application of the critical path method where allocating more resources, or another form of extra cost, reduces processing time. The timing or sequencing of tasks in machine scheduling problems is often done by allocating a machine with a task to a time

slot. The advantage is that it is very easy and intuitive to obtain a schedule from the solution. The drawback however, is that the schedule works with minimal steps in time equal to the length of a time slot.

Cattrysse and van Wassenhove (1990) identify vehicle routing in distribution systems as applications of the GAP. Vehicle routing problems (VRP) are not modelled as bipartite graphs as done by Chang and Ho (1998) where edges represent an agent-task assignment, but graphs in which edges represent a relation between the nodes/tasks (e.g. distance). The VRP is a generalisation of the traveling salesman problem (TSP), which does not typically include capacity limits, but has a similar formulation. Laporte (1992) provides an overview of VRPs with different side conditions, among which are time windows and precedence constraints. The shipments belonging to a flight can only be build up when they have passed the breakdown phase and each flight has a deadline which would reflect an upper bound of the time window. Desrochers et al. (1990) introduce, like Graham et al. (1979) did for machine scheduling, a classification scheme for vehicle routing and machine scheduling problems together with examples. Variations discussed previously for the AP/GAP are also possible for the VRP, because it is an extension of the former. Bottleneck constraints and vehicle characteristics (e.g. team skills) are included in the problem described by Bell et al. (1983), which shows that the VRP can include such characteristics too.

A comparison is made by Beck et al. (2003) between a vehicle routing and job shop scheduling problem as a representation of a real world scheduling problem. Both problems can be reformulated to represent the other and both problems are solved in different ways. It is found that routing technology is superior when the problem involves optimizing the sum of total transition times, few precedence constraints and low resource specialization. This is in line with the project context. At KLM Cargo, there is low resource specialization because there will be multiple teams capable of executing the same task/flight and there are generally a few star flights which require precedence over other flights.

Optimization Techniques

When working with a linear problem containing binary and integer variables, commercial optimizers such as Gurobi will use a branch-and-bound or branch-and-cut technique to optimize relaxed subsets of these problems. Cordeau (2006) presents an example of valid inequalities for the dial-a-ride problem with time windows.

In turn, many heuristic methods have been developed tailored to specific problems in order to find good-solutions in a shorter time. Heuristics can be (non-)deterministic using local/global search procedures. Hillier and Lieberman (2015) explains that the tabu search is both deterministic and can be used as a local search method. An initial solution should set a good starting point. If a good initial solution can be generated, the optimal solution should be located somewhere in the neighbourhood. **?** generates an initial solution by ordering the nodes by the angle made with the depot in a Euclidean coordinate space by randomly testing new nodes. At KLM Cargo the travel time is less of a concern since its greatly exceeded by the service time. Focusing directly on the deadlines instead of distances could be more promising.

Cordeau et al. (2001) presents a unified tabu search heuristic for the VRP with time windows and the periodic and multi-depot VRP with time windows. The algorithm consists of a local search metaheuristic and a diversification mechanism to explore a broader portion of the solution space. The best feasible solution is post-optimized according to the specialized heuristic by Gendreau et al. (1998). This method thrives in its simplicity because it is based on a limited number of parameters and a simple scheme, compared to alternative heuristic approaches.

Modeling under uncertainty

The research done in Delgado et al. (2019) contains an example of dealing with uncertain parameters. The article presents the allocation of cargo to passenger flights as a multi-commodity flow problem with uncertainty in demand and capacity. The demand is assumed to be normally distributed and is converted into a deterministic chance constraint. The constraint is subject to change depending on the required confidence level requested by management or attainable confidence level from the input data.

Next to chance constraints, Hillier and Lieberman (2015) also describes that robust optimization can be applied in order to deal with uncertainties. The goal of robust optimization is to virtually guarantee a solution for all practically possible parameter values. For this a distinction is made between a soft constraint and a hard constraint. A hard constraint defines a limit that must be satisfied and a soft constraint can be violated a little bit without serious complications.

3 Business Analysis

This section presents the operations at the Hub of KLM Cargo at Schiphol Airport. First the business structure and key terms in the process within the scope of this research are defined. Next an overview of the buildup phase with its scheduling challenges is analyzed. Finally, the framework for optimization and the model inputs and parameters are derived from this section.

3.1 KLM Cargo and the Hub

Air France-KLM Group has 3 main business activities: Passenger Business, Cargo Business, and Engineering & Maintenance. KLM Cargo is part of the dedicated cargo business in the Air France-KLM Group. Freight from national and international destinations passes through this facility to be processed for further transportation. As part of the Skyteam, KLM Cargo not only handles its own cargo, but also that of Delta Airlines and Martinair. The fleet of passenger-, freighter-, and combination (combi) aircraft transports the cargo between airports around the world. The amount of space to store cargo depends on the aircraft type. The cargo facility is referred to as the Cargo Hub, since it is located at KLM's hub airport, Schiphol.

Transporting cargo in an aircraft requires a special carrier device called Unit Load Device (ULD). ULDs are crucial in transporting air cargo safely and maximize the use of the aircraft's cargo space. A ULD is either an aircraft pallet and pallet net combination, or an aircraft container. The International Air Transport Association (IATA) regulates the design of ULDs to ensure standardization and safety in the aviation industry. They come in many different shapes, but they can, for this research, be distinguished by three types:

- Main Deck Pallet (MDP): Main deck pallets are taller than lower deck pallets. The cargo is loaded onto a large plate and secured with a sail and net.
- Lower Deck Pallet (LDP): Lower deck pallets are constructed similarly to the MDP, but are built less tall.
- Container Pallet (AKE): A container fully encloses the shipments rather covering them with a tightly secured sail and net. They are lightweight structures with a base, and side- and roof panels.

MDPs cannot be carried by passenger aircraft, since they are built for taller cargo spaces in the main deck. Freighters and combi-aircraft carry all types of ULDs with the freighter, as the name suggests, being fully dedicated to cargo load instead of passengers.

The KLM cargo Hub consists of 3 freight buildings, shown in Figure 1. Freight building 1 is a stand alone building and takes care of all the cargo that includes mail, small packages, special shipments, pharmaceuticals and animals. It has its own handling process and is outside the scope of this project. Freight building 2, or Europort, is responsible for all incoming general cargo arriving by aircraft and headed for European destinations. Freight building 3, or World Port, is connected to building 2 and handles all of the general cargo, either arriving from trucks or from aircraft, which is destined for Intercontinental destinations.

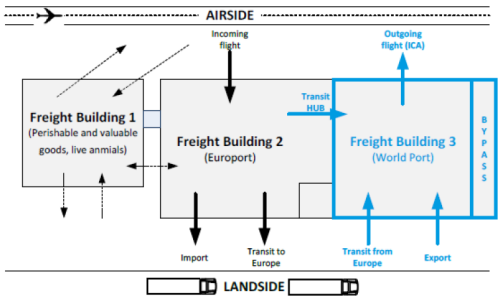


Figure 1: The freight buildings at the Cargo Hub at Schiphol Airport.

3.2 The Hub process

Looking at the flow of cargo on international flights, the process at freight building 3 can be separated into 5 stages: Acceptance, Storage, Breakdown, Buildup, and Transport.

Acceptance

The process at freight building 3 starts with the arrival of a trucks at the landside of the Hub. Documentation is reviewed and the Air Way Bill

(AWB) is stored in the operational management system, CHAIN. Freight can either be transit from outside The Netherlands, or export, from inside The Netherlands. Trucks are unloaded at the Moving Truck Dock (MTD), Ready For Carriage (RFC) checks are executed to confirm proper packing, completeness and no damages. Freight can be loose or stored on skids in which case it identifies as loose freight or on ULDs. The ULDs are unloaded and the ULD-category is established which determines further actions:

- ◇ Bijbouwer ULD (BB): A ULD of which not all space is used. There is room for other shipments with the same destination to be loaded on this ULD.
- ◇ Mixed ULD (M-ULD): This ULD contains shipments with different destinations. The shipments need to be separated for further transportation. If some shipments are left on the ULD, its status is changed to a BB.
- ◇ Through ULD (T-ULD): This ULD contains shipments with the same destination. It is usually immediately ready for flight, but sometimes there is space left, or it needs a check due to poor build quality. In those cases, the status is changed to a BB and is checked during buildup.

If a T-ULD is in order, it is moved to storage and needs no further work. Loose freight is sorted on non-mixed skids, grouped by destination. M-ULDs and BB-ULDs are moved to storage and are collected later on. Exceptions to the regular process of unloading freight at MTD or Export go through the bypass, located on the far side of freight building 3. It is used for special shipments, which have a short connection time like flowers, or have a large volume like a car.

Storage

Once the ULDs from the MTD satisfy the checks, they are moved to the Pallet Container Handling System (PCHS) by elevator or onto a dolly cart. The PCHS is an automatic storage system located on the upper floor of the Hub which can store up to 1200 ULDs. T-ULDs are brought down automatically a couple of hours before the scheduled departure time. M-ULDs come down earlier since they require additional handling. Loose freight can be moved to regular shelves, cooled storage, shelves for dangerous goods, and the buffer zones. A dolly cart is used if it contains very large cargo or if the cargo needs cooled storage. T-ULDs with very short connection times are sometimes also put on dolly carts and directed via the bypass to shorten the throughput time.

Breakdown

Two processes follow inside the Hub: Breakdown & Buildup. In short, these stand for separating shipments of different destinations from one ULD and putting them back together on different ULDs for the same destinations. At Breakdown, M-ULDs are automatically brought down from the PCHS, depending on the shipment with the earliest departure time. When the ULD is broken down completely, only skids and loose freight remain. Loose freight is placed on skids to enable transportation to the shelves or buffer zones. When a large volume shipment is left on the ULD, it is transported back to the PCHS and given the BB status. There can also be freight on the ULD heading for European destinations, which is then transported to freight building 2. So after Breakdown, cargo is stored in one of three ways: in the shelves, at the buffers, (back) in the PCHS system or transported to freight building 2. Shipments can also come from the breakdown in freight building 2, when they are on transfer from a European flight to an intercontinental flight.

Buildup

The buildup area consists of bays. Each bay is made up of buffer zones, buildup pits and transport trains. Figure 2 gives an overview of the layout. The buffer zones are marked out areas on the floor where shipments are placed, sorted by flight, that need to be loaded onto ULDs that are placed in the pits. The train is used to move the ULD once it is checked and Ready For Flight (RFF).

Shipments arrive at the buffer zone directly from breakdown or are taken from storage locations. Buildup implies putting a ULD onto the buildup pit, taking shipments from the buffer zones, placing them on the ULD, and carefully securing them for further transport. When considering the type of work for buildup there are 2 categories in which they can be divided:

- ◇ Leeg Fust (LF): An LF is an empty pallet on which all shipments are newly placed.
- ◇ Bijbouwer (BB): As mentioned in Section 3.1, when there is space for other shipments, a ULD is labled as BB. Two subcategories are distinguished in terms of work.
  - BB to build: When there is space left to place more shipments, it is labeled as BB to build in order to utilize the full capacity of the cargo hold and not waste cargo space in the aircraft.
  - BB to check: When a full T-ULD does not satisfy RFC requirements, it is labeled as BB to perform a check. This type of work does not imply loading shipments but rather checking nets, labels and build quality.

When the ULD has reached its maximum volume, it is covered with a net secured at the rim. It then gets pushed back onto a so-called train.

Transport

The trains move the ULD further down through the freight building into the Transportation area. Here the ULD is prepared for transport over the apron or stored back in the PCHS for later shipping.

3.3 Scheduling tasks for buildup

The KLM Cargo Hub is operating all day during three shifts. Rostering personnel for each shift starts months before the actual buildup shift and is done by a staff planner. On the day itself, the buildup tasks are scheduled for the next shift. The terms task and flight are considered interchangeable since a task encompasses the buildup of all ULDs for one flight. A buildup task involves the assignment of either entire bays or individual flights to teams. Each team has two employees and is then responsible for the buildup of all ULDs on its assigned bay or flights. Table 1 shows the characteristics of each shift. Notice the indication of a generally more resource intensive morning shift in relation to the evening shift.

Table 1: Shifts in the KLM Cargo Hub

Shift	Starting time	Ending time	Allocation by	Resources
Night	22:00	06:00	Bay	Low
Morning	06:00	14:00	Flight	High
Evening	14:00	22:00	Flight	Medium

The list of teams per shift contains permanent KLM employees and contractors (or flex-employees). A shift leader takes the list of teams from the staff planner and is then responsible for making a task allocation per team based on the current work progress, using: multiple attendance sheets, the operational database Chain to track Hub operations, a list of starflights set by the Cargo Control Center (CCC), and the list of skills and preferences per employee and team. Starflights are high yield or constrained flights which are more important to build in time than other flights. The flight schedules are made twice a year and so are the assignments of flights to specific buffer zones. An assignment of a team to a flight therefore automatically implies a work location in the Hub.

3.3.1 The inputs used for scheduling

Inputs for the scheduling problem can be divided into two groups: the set of tasks and the set of teams. They are combined to make a work schedule for the coming shift. Each set of inputs comes with its own list of characteristics.

The task input characteristics

In estimating the amount of work, the Hub assumes a nominal build time for MDPs, LDPs, and AKEs or 45, 40, and 25 minutes, respectively. This is for LF ULDs, but a BB generally take less time. Additionaly, work could have already started for some ULDs which reduces build time for the coming shift.

Each task is executed at a specific buffer-pit location and teams move from one buffer to the other. Tasks can start depending on the amount of shipments available at the buffers. Each task has a RFF time, or deadline. For KLM and Martinair cargo, this is 2.5 hours before flight departure and for Delta Airlines it is 3 hours before departure. Finally, the starflights are high yield or constrained flights and are more important to build in time than other flights.

The team input characteristics

Employees are sometimes not fully employable and are given a smaller workpackage. Teams also differ in skill. A Subject Matter Expert (SME) at buildup estimated that a team of two Flex employees works 10% slower than two regular KLM employees or a KLM/Flex combination. Furthermore, employees that are of age or in training are on the Golden Bay (bay 7). They are often given the same tasks and do not work on other bays.

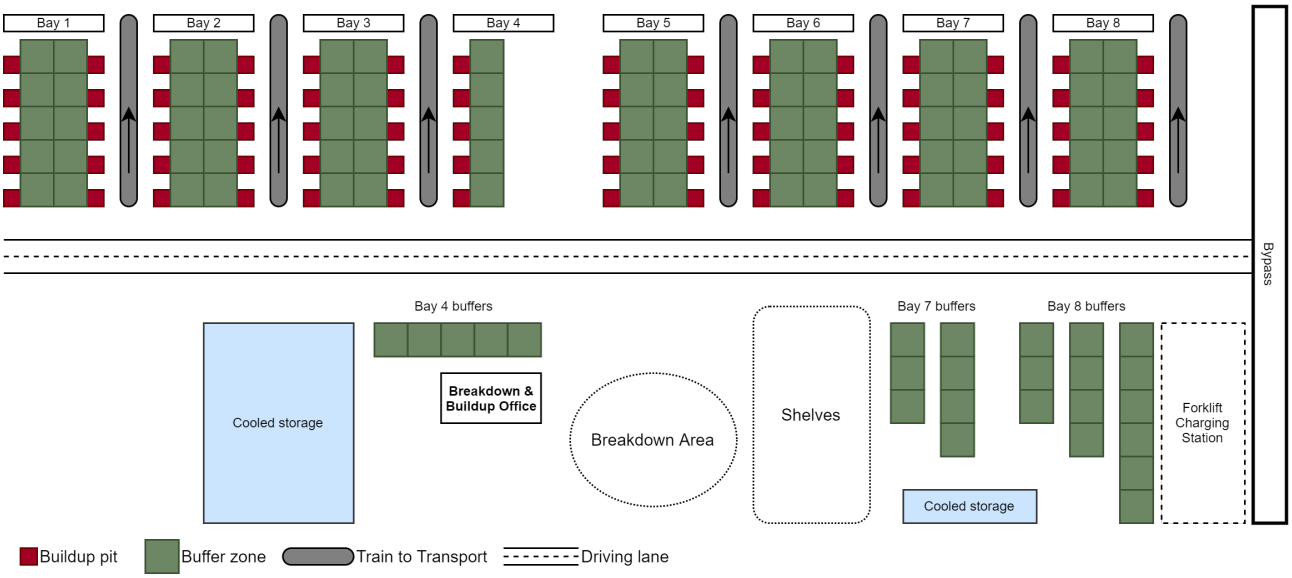


Figure 2: Layout of bays, buffer zones and buildup pits in the KLM Cargo Hub

3.4 Preliminary conclusions

Based on the evaluation of the scheduling process, there is a list of conclusions and assumptions within the scope of this research that should be considered in the modeling of the problem. The simplifications are as follows:

- ◇ A schedule is made for one shift at a time, either morning or evening.
- ◇ Schedules are made by allocating tasks by flight, not by ULD, to one or more teams during the shift.
- ◇ Multiple teams can work on the same flight.
- ◇ Each task has a RFF threshold, or deadline before which buildup should have finished.
- ◇ Deadlines for KLM and Martinair are set at 2.5 hours before departure and 3 hours for Delta Airlines.
- ◇ ULD types are assumed to be one of: MDP, LDP, or AKE.
- ◇ The nominal buildup time for an MDP, LDP and AKE is 45, 40, and 25 minutes, respectively.
- ◇ Teams on the Golden Bay do not work on other bays.
- ◇ It is assumed that a team of Flex/Flex works 10% slower in building up a pallet than a KLM/KLM or KLM/Flex team.
- ◇ Some employees can have a lower employability.
- ◇ Starflights are more important to finish in time and have priority over regular flights.
- ◇ It takes some time for a team to move from one buffer to another.
- ◇ Each team is always given a break during their shift.

4 Exploratory Data Analysis

In order to validate a model, it needs to be computed with real data. This section is dedicated to the collection and analysis of real operational input data from the Cargo Hub. The goal is to answer the question *"What is work?"*, in other words, define the amount of work per shift per flight to estimate schedule times. To achieve this, a couple of data related questions need answering:

1. Out of all ULDs that went on a specific flight, which of those generated work for buildup?
2. How can the release time of a flight be set for a specific flight?

As mentioned in Section 3.3, the input data consists of a set of teams and a set of flights. All the information regarding the teams alongside the (star)flights and corresponding workstations can be extracted from planning documents used during a shift. Apart from that there needs to be information on the flight departure times, the type and number of ULDs and their release times. By doing an exploratory data analysis, assumptions about these parameters can be made. The information about the amount of work for each flight should be more reliable and lead to better schedules instead of being determined based on experienced guesses with incomplete information.

After reviewing the data architecture in Section 4.1, Section 4.2 will show the findings of this analysis after which answers to the questions are given in Section 4.3.

4.1 Database architecture

The raw data is extracted from a data visualization tool called *Spotfire*. This application has a.o. a live data connection to *Teradata*, an Air France-KLM server located in Paris containing data from *Chain*, an operational database that logs actions in the KLM Cargo Hub. The information from Chain is uploaded once a day between 3 A.M. and 4 A.M. The collection of tables used in this project is called *HubTrack*, a temporary database used for this project that consolidates multiple data points in order to track the actions of each shipment in the Hub.

Hubtrack is a auto-refreshing database containing data from the past 8 weeks. The data subject to this analysis is compiled by extracting new data every couple of weeks and combining it into a dataset of 13 weeks from Monday June 22^{nd} to Sunday September 20^{th}.

Shipments are tracked by AWB number, i.e. a contract between carrier and shipper. Rows of data where an AWB was not associated with either an outgoing ULD, an outgoing flight, or flight departure date, were dropped. Additionally, when an outgoing ULD did not contain any timestamp information about its completion, its row was dropped. Furthermore, incorrect data types where restored and flight number and flight date information was enhanced. This created a clean dataset of 267,990 rows which was used in this analysis.

4.2 Buildup Work Analysis

The analysis is done in 3 steps. First, the ULD types need to be identified to assess the required time of work. Only ULDs that are passing the buildup phase need to be included, which is determined next. Finally, the data is examined to see what it can tell about the release times and buildup times.

Identifying ULD types

The identification of the 3 ULD types from Section 3.1 is done according to a simple scheme shown in Table 2 based on two data points. For a subset of name tags the height was ignored and assumed as an error in the data.

Table 2: Determination of ULD type

ULD Name Tag	ULD Height	
	< 165 cm	≥ 165 cm
AKE, AKH, RAP, RKN, DPE, DPL	AKE	-
AAP, PAJ, PAP, PLB, PLA	LDP	-
Other	LDP	MDP

The Categorization of ULDs from the data

Dismissing the T-ULDs that need no checks and go straight to storage or transport, there are two categories of ULDs that are encountered during the work: LF and BB ULDs. For BB the distinction is made between building and checking.

With the help of several data points, a decision tree was designed to filter the different categories of ULDs. The scheme resulted in the distribution of the different pallet categories and types shown in Table 3.

Table 3: Distribution of buildup ULDs on ICA flights

ULD Type	ULD Category				Total by type (%)
	T-ULD	LF	BB		
			Build	Check	
Main Deck Pallet (MDP)	153	821	257	118	4%
Lower Deck Pallet (LDP)	11280	14669	2555	2579	86%
Container (AKE)	964	2641	21	93	10%
Total by category (%)	34%	50%	50% 16%	50%	100%

Table 3 shows that more than half of all ULDs require buildup activity. 34% of all ULDs on ICA flights are T-ULDs and do not require work. The most common type of ULD is a LDP with 86%. Furthermore, most AKEs are T-ULDs, which is verifiable by the fact that these are fully enclosed and usually completely full from the outset.

Buildup ULDs summarized by flight and date

Depending on the aircraft type and amount of non-standard cargo, every flight carries a different amount of ULDs. Figure 3 shows that Martinair is represented in the lower range, which is because ULDs from Martinair passing through the the KLM Cargo Hub are all for buildup and KLM Cargo builds only part of Martinair's ULDs. Still most of the ULDs that move through the Hub are for KLM flights. The overall median amount of ULDs per flight is 4.

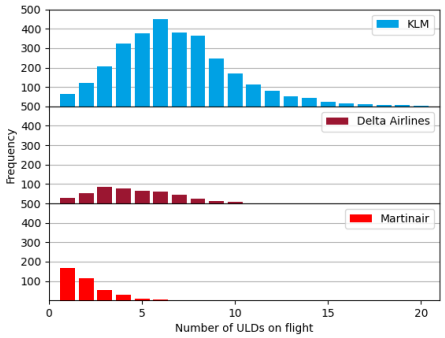


Figure 3: Frequency of ULD amounts per ICA flight per carrier, n=3990

Some days of the week are less demanding than others. Figure 4 shows the daily amount of ULDs that are transported on Intercontinental (ICA) flights. Demand for air transport is clearly influenced by weekday as can be seen by the periodic behavior of the graph. High demand occurs on Fridays and Saturdays, while low demand occurs on Mondays and Tuesdays. Many typical customers of the air transport service produce goods during the week and want these shipped over the weekend. Simultaneously, the consignees expect new shipments after the weekend, when a new business week starts. Therefore, the ULDs that are transported increases per day over the course of the week towards the weekend.

Determine release time and deadline of ULDs

A handover is defined as placing a shipment in storage or at the buffer, meaning it is available for buildup. The graph in fig. 5 shows the handover progress at the time of the first buildup in bins of 5%. About 85% of all ULDs have their first shipment placed when 96% -100% of shipments are handed over. This suggests that teams start buildup of a ULD, when there are enough shipments present to fill the entire ULD.

It needs to be noted however, that it often occurs that shipments are all loaded onto ULDs first, and get scanned later, when more shipments have already been put on the ULD. Even with this information, there is still too little evidence suggesting many ULDs are built up when only a small part of the shipments are available (looking at 0% - 50%).

Since the work packages are made per flight, not per ULD, it is necessary to look at the buildup information of all shipments on a flight to see when

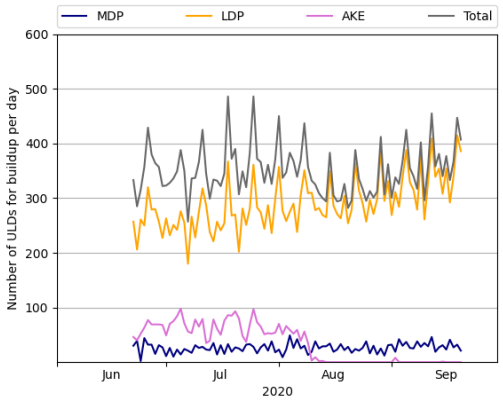


Figure 4: Buildup ULDs on ICA flights by date

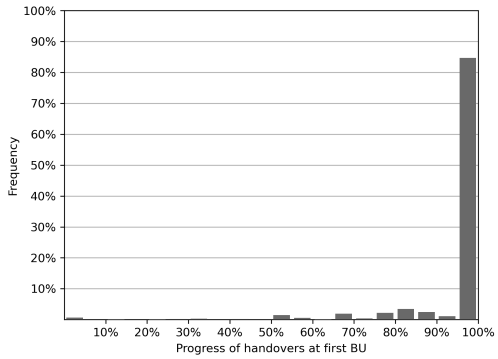


Figure 5: ULD handover progress in percentages of total shipments on a ULD.

a task can be started. Figure 6 shows the number of ULDs for which all shipments have been handed over at the moment of the first buildup for that flight, distributed by size of flight. A trend is visible that flights with a larger work package starts buildup when more ULDs of shipments have been handed over. At the same time, the spread is rather large. In consultation with an SME it is decided that buildup can start when just one ULD can be built, in order to not constrain the scheduling problem too much.

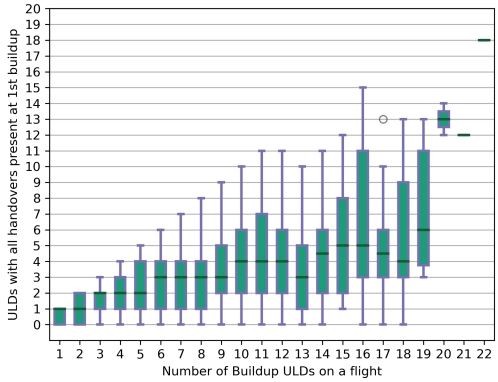


Figure 6: ULDs with all shipments handed over at the time of first buildup per flight as a function of the amount of buildup ULDs per flight.

Figure 7 shows the hours difference between the buildup of the 1st shipment of a ULD and the flight departure time, quantized by 30 minutes. The graph compares two airlines, KLM and Delta Airlines. It can be seen that some ULDs from KLM flights and a little less for Delta Airlines, have their first buildup more than 12 hours before departure, or even much earlier. A large part of ULDs start buildup more than 8 hours before departure, which confirms that ULDs are being built in earlier shifts.

On the other hand, the last buildup per ULD can be seen in Figure 8. The data is significantly more skewed than in Figure 7. What needs to be noted

is that for flights of KLM, the CCC set the RFF deadline 2 hours before departure, while for flights of Delta Airlines, this is set at 3 hours before departure. This difference is visible in the graph with some exceptions of late completions of ULDs.

The smaller spread in Figure 8 is explained by the fact that ULDs are sometimes partly built up, only to be finished in a later shift closer to departure.

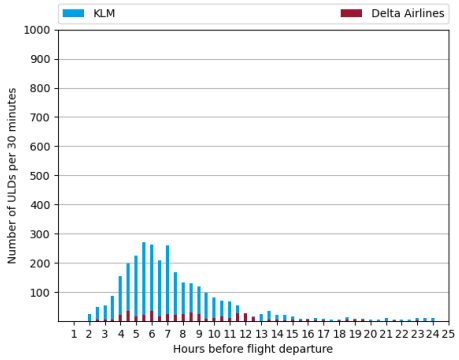


Figure 7: Frequency of times between first buildup and flight departure per ULD

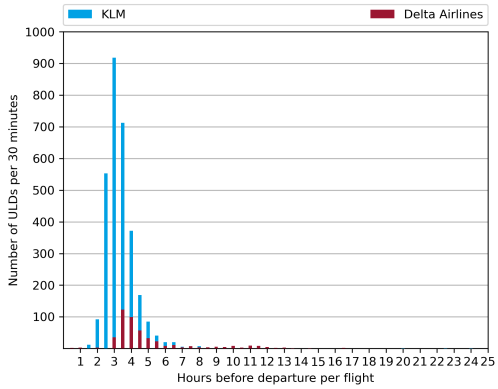


Figure 8: Frequency of times between last buildup and flight departure per ULD

The three types of ULDs take a different time to buildup. This is shown in fig. 9. What can be seen is that the median buildup times of LF ULDs are 37, 31, 12 for MDPs, LDPs, and AKEs, respectively. The median buildup times of BB-build ULDs are 34, 25, and 11 for MDPs, LDPs, and AKEs, respectively. The IQR of BB-build ULDs is on average 22% larger, which indicates more variability among BB ULDs. The data is consistent with the notion that an buildup times of an MDP $>$ LDP $>$ AKE for both LF and BB-build.

The times indicated by the SME are above the median, but they do fall within the IQR. The reason for this anomaly is that teams often scan the first couple of shipments in one go instead of individually when they are placed on the ULD. In consultation with the SME it is decided to leave the nominal buildup times as is and assume an 20% reduction in case of BB-build ULDs, rounded to whole numbers. Additionally, 15 minutes is assumed for BB-check ULDs, regardless of ULD type. These are recognized in the data when the ULD has been on a buildup pit, but there are no shipments built up.

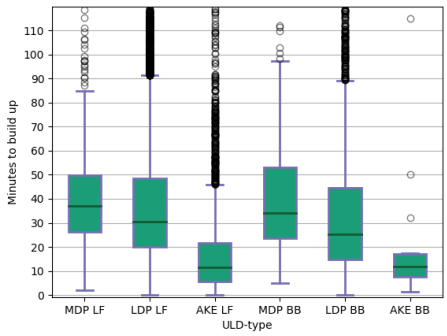


Figure 9: Timespan between begin and end of buildup for different categories and types of ULDs

Furthermore, ULDs could already have started their buildup in an earlier shift. In this case, it is assumed in consultation with a SME that their required work should be halved, rounded to whole numbers. Table 4 summarizes the buildup times for all categories and types of ULDs during buildup.

Table 4: Buildup times used for different types and categories of ULDs

	LF	$\frac{1}{2}\ LF$	BB_{buid}	$\frac{1}{2}\ BB_{build}$	BB_{check}
MDP	45	25	35	18	15
LDP	40	20	30	15	15
AKE	25	15	20	10	15

4.3 Preliminary Conclusions

Based on this exploratory data analysis, there is a list of conclusions and assumptions within the scope of this research that should be considered in the modeling of the problem. The simplifications are as follows:

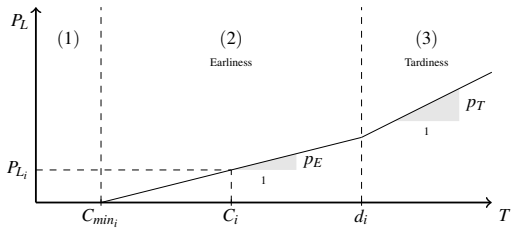
- ◇ The buildup times of different ULD cases are assumed as in Table 4.
- ◇ Release times of tasks are defined by the time when all shipments for a single ULD are handed over.

5 The Mixed-Integer Linear Multi-Objective Scheduling Problem

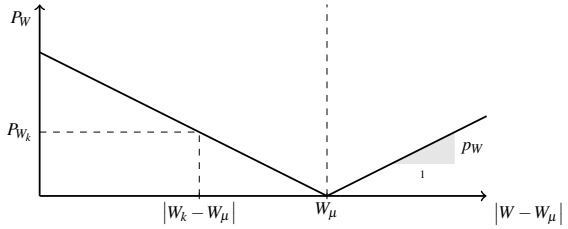
The scheduling problem consists of a set of n independent tasks $N = \{i, j\}_{i,j=1}^{n}$. Tasks are executed by a set of m teams $K = \{k\}_{k=1}^{m}$. Throughout the Hub there are different bays at which tasks are executed and it takes a certain time t_{ij} to transfer from one bay to the other. A time window $[r_i, d_i]$ is associated with each task $i \in (N, N_B)$, where r_i represent the release time at which service of task i may begin and d_i represents its deadline. Tasks have a nominal service which is subject to the skill attribute of a team, resulting in the team-dependent service time of a task s_i^k.

The measure of the scheduled margin between a task's completion time and its due time is defined as the lateness L_i. When a task is scheduled to be completed before its deadline, lateness is negative, also called scheduling with earliness. In the opposite case when a task is scheduled to be completed after its deadline, lateness is positive, also called scheduling with tardiness. A penalty P_{L_i} is assigned to the lateness of each task, starting from 0 at the earliest possible completion time $C_{i_{min}}$ (that is, in case it is executed by either one of the fastest working teams and started immediately at r_i). The penalty increases linearly for every minute a task is scheduled after $C_{i_{min}}$. Tardiness of a task is discouraged by increasing the rate. The relation between completion time, lateness and penalties is illustrated in Figure 10a, along with three distinguished regions.

Completion of task i cannot occur before $C_{i_{min}}$ in region (1). Completion in region (2) means completion is early and completion in region (3) means it is tardy. Each team's skill level affects its service time for a task. In addition, there are teams that will be working solely on the Golden Bay, i.e. bay 7. When a team is assigned to the Golden Bay, it cannot be assigned to flights on other bays. A team's employability, is used here as a team's workload capacity, indicated by cap_k. The workload is not penalized in absolute sense like the lateness. A workload penalty is incurred per absolute percentage point difference between the workload of a team and the average workload of all teams together. This is illustrated in Figure 10b. The notation of the input sets, parameters and decision variables of this model have been summarized in the list below.



(a) **Example** The lateness penalty P_L as a function of lateness L_i for task i. Task completion C_i cannot occur earlier than C_{min_i} and tardiness demands a heavier penalty than earliness.



(b) **Example** The workload penalty P_W as a function of the difference between workload W_k and the average workload W_μ for team k. Workloads cannot exceed 1.0.

Figure 10: Progression of penalties as a function of lateness and workload

Indices and sets	
i, j	tasks $i, j \in (0, N, N_B, n+1)$
$0, n+1$	the first and last task in the graph
k	teams $k \in K$
N	the set of tasks
K	the set of teams
N_G	the set of tasks on the Golden Bay, ($N_G = \{i \mid i \in N, i$ at Golden Bay$\}$ and $N_G \subseteq N$)
N_B	the set of tasks modeled as breaks, ($N_B = \{i_b^k \mid k \in K\}$ and $N_B \subseteq N$)
i_b^k	break tasks for each team
K_G	the set of teams working only at the Golden Bay, ($K_G \subset K$)
Parameters	
s_i^k	the service time of task i by team k
t_{ij}	the transfer time from task i to j
r_i	the release time of task i
d_i	the due time of task i
p_0	the penalty for completion at due time, i.e. $L_i = 0$.
p_E	the penalty for every minute of reduced earliness of a task
p_T	the penalty for every minute of increased tardiness of a task
p_W	the penalty for every minute of increased workload of a team
f_i	multiplier for the penalty of important tasks
M_{ij}^k	a sufficiently large positive number
α	the relative importance of lateness penalties
Decision Variables	
X_{ij}^k	assignment of team k to do task j after i
B_i	the beginning time of task i
C_i	the completion time of task i
L_i	the lateness of task i
P_{L_i}	the penalty assigned for lateness of task i
$P_{L_{max}}$	the maximum assigned lateness penalty among all tasks
W_k	the workload of team k
W_μ	the average workload among all teams $k \in K$
P_{W_i}	the penalty for the difference in workload of team k from W_μ
$P_{W_{max}}$	the maximum assigned workload penalty over all teams

With these definitions of inputs, parameters and decision variables, the proposed mixed-integer linear multi-objective scheduling problem (MILMSP) can be formulated as follows by choosing the X_{ij}^k and B_i so as to:

$$\text{Minimize} \quad \alpha \cdot P_{L_{max}} + (1-\alpha) \cdot P_{W_{max}} + \beta \cdot R \tag{1}$$

$$\text{where} \quad R = \left(\frac{1}{|N|} \sum_i^N B_i + P_{L_i} + \frac{1}{|K|} \sum_k^K P_{W_k} \right)$$

$$\beta \ll \alpha$$

Subject to

$$\sum_{j \neq i}^{N,N_B,n+1} \sum_k^K X_{ij}^k = 1 \qquad \forall\, i \in (N, N_B) \tag{2}$$

$$\sum_j^{N,N_B,n+1} X_{0,j}^k = 1 \qquad \forall\, k \in K \tag{3}$$

$$\sum_i^{0,N,N_B} X_{i,h}^k = \sum_{j \neq i}^{N,N_B,n+1} X_{h,j}^k \qquad \forall\, h \in (N, N_B),\ k \in K \tag{4}$$

$$\sum_i^{0,N,N_B} X_{i,n+1}^k = 1 \qquad \forall\, k \in K \tag{5}$$

$$\sum_{j \neq i}^{N,N_B,n+1} \sum_k^{K_G} X_{ij}^k = 0 \qquad \forall\, i \notin N_G \tag{6}$$

$$\sum_j^{N,n+1} X_{i_b^k,j}^k = 1 \qquad \forall\, k \in K \tag{7}$$

$$B_j \geq (B_i + s_i^k + t_{ij})\, X_{ij}^k \qquad \forall\, i, j \in (N, N_B),\ k \in K \tag{8}$$

$$B_i \geq r_i \qquad \forall\, i \in N \tag{9}$$

$$B_i \leq d_i \qquad \forall\, i \in N_B \tag{10}$$

$$C_i \geq B_i + \sum_j^{N,N_B,n+1} \sum_k^K s_i^k X_{ij}^k \qquad \forall\, i \in (N, N_B) \tag{11}$$

$$\begin{aligned} P_{L_i} &\geq f_i \cdot p_E \cdot (C_i - C_{min_i}) \\ P_{L_i} &\geq f_i \cdot p_T \cdot (C_i - d_i) + P_{L=0} \end{aligned} \qquad \forall\, i \in (N, N_B) \tag{12}$$

$$P_{L_{max}} \geq \max_{i \in N, N_B} \{P_{L_i}\} \tag{13}$$

$$W_k = \frac{1}{cap_k} \sum_i^{0,N} \sum_j^{N,n+1} s_i^k X_{ij}^k \qquad \forall\, k \in K \tag{14}$$

$$W_\mu = \frac{1}{|K|} \sum_k^K W_k \qquad \forall\, k \in K \tag{15}$$

$$P_{W_k} = p_W \cdot 100 \cdot \left| W_k - W_\mu \right| \qquad \forall\, k \in K \tag{16}$$

$$P_{W_{max}} \geq \max_{k \in K} \{P_{W_k}\} \tag{17}$$

$$X_{ij}^k \in \{0,1\} \qquad \forall i, j \in (0, N, N_B, n+1),\ k \in K)$$

$$B_i, C_i \in \mathbb{N} \qquad \forall\, i \in N,\ k \in K$$

$$W_k, P_{L_i}, P_{W_k}, W_\mu, P_{L_{max}}, P_{W_{max}} \in \mathbb{R}^+ \qquad \forall\, i \in N,\ k \in K$$

The first term in the objective function (1) minimizes the maximum lateness penalty among all tasks as suggested by Ravindran and Ramaswami (1977). The second term minimizes the difference between the maximum and average workload among all teams, introduced by Duin and Volgenant (1991). These minimax (or bottleneck) terms minimize the value of a worst case (most costly) scenario. The other tasks which are not the worst case can have any penalty as long as it is less than the maximum penalty. This leads to suboptimal results and does not aid in the convergence of the model. The third term therefore minimizes the aggregation of time- and penalty variables as a residual factor and counters the inherent behavior caused by the bottleneck terms. Because it only serves as a correction and not as the primary objective, β will be given a value many orders of magnitude smaller than the former terms, adjusted for the model size.

Constraint (2) is a degree constraint which ensures each task is visited exactly once. Constraints (3) – (5) guarantee that each team k starts at node (0) and ends at $(n+1)$ while maintaining flow continuity. Constraint (6) prohibits Golden Bay teams from leaving that bay. Constraint (7) makes sure that each team is at least assigned a break, as was stated in Section 3.4.

Consistency of time variables is enforced by constraint (8). A new task can be started after servicing and moving from the previous task. Each task

is imposed a minimum starting time by constraint (9), but only the break tasks are given an upper bound.

The completion time is defined by equality (11). The inequalities from (12) represent the convex piece-wise linear penalty functions shown in Figure 10a. The maximum lateness penalty is defined by equality (13).

The workload per team is determined by equality (14) as a fraction of a team's workload capacity. The mean workload is expressed in equality (15) which governs the assigned workload penalty from equality (16) as shown in Figure 10b. The maximum workload penalty is defined by equality (17).

Linearization

Constraint (8) makes this formulation non-linear, so it is necessary to reformulate it. Linearization can be done by introducing a sufficiently large constant M_{ij}^k (Hillier and Lieberman (2015)):

$$B_i + s_i^k + t_{ij} - M_{ij}^k(1 - X_{ij}^k) \leq B_j \qquad \forall\, i, j \in N, k \in K \tag{18}$$

The value of M_{ij}^k imposes large negative constants in the constraints that are not part of the optimal solution. To be more computationally efficient a minimum value of M_{ij}^k that is large enough is found by setting Cordeau (2006):

$$M_{ij}^k = \max\{0,\ d_i - r_i + s_i^k + t_{ij}\} \qquad \forall\, i, j \in N, k \in K \tag{19}$$

Soft constraints

As can be seen, there is only an upper bound on the B_i-variables for the breaks. It is uncertain whether all tasks can be completed before their deadlines. This complication can be worked around by robust optimization techniques described in Hillier and Lieberman (2015). Looking at Figure 10a the model allows time window (deadline) violation in exchange for a large penalty, imposed by constraint (12). This transforms the hard constraint of an upper bound into a soft constraint.

On the contrary for the workload, there are no defined upper bounds. Instead of penalizing a workload > 1, no penalty is given. A violation of workload is not seen as a complication as serious as the violation of task deadlines. Teams from the next shift or the same shift can take over the excess work. The most important part is balancing the workloads, not minimizing them.

Preprocessing

As a consequence of some constraints, some arcs can be excluded from the model formulation. There are several grounds on which to do so. 1) Teams can only visit their own break task and are thus not allowed to visit the break tasks of other teams. The model will ensure the assignment of a team to one break task according to constraints eq. (2) and (7). The variables $X_{b_{k'},j}^{k'}$ where $k' \neq k$ need therefore not be defined. In the same way it is not necessary to define arcs to 0 and from $n+1$. The variables $X_{i,0}^k$ and $X_{n+1,j}^k$ need therefore not be defined either.

Another consideration could have been to eliminate variables where $X_{ij}^k = 0\ \forall\, k \in K$ when $d_j < r_i$. However, this has not been done because of the soft upper bounds (d_i), which should allow the use of these arcs.

Valid inequalities

An additional constraint that does not eliminate feasible solutions is called a valid inequality. The lower bounds of the time variables (B_i) can be strengthened with knowledge about the order of the assigned tasks. This has been suggested by Desrochers and Laporte (1991) and is adapted for this problem in the following inequality:

$$B_j \geq r_i + \sum_{i \neq j}^{N,N_B} \sum_{k}^{K} \max\{0, r_i + s_i^k + t_{ij} - r_j\}\, X_{ij}^k \quad \forall j \in N, k \in K \tag{20}$$

6 The Tabu Search Method

A heuristic algorithm has been used during this project in light of the third research question. It is an attempt to solve the scheduling model quicker, speed up implementation of this model and to not be independent of commercial solver software. The initiation of the algorithm is described in Section 6.1 after which the heuristic algorithm defined in Section 6.1.

6.1 Algorithm for finding an Initial Feasible Solution

An initial solution is obtained through a greedy algorithm based on Jackson's Theorem for static single-machine scheduling (Jackson (1955)), which is an *Earliest Deadline First* (EDF) approach. The approach is adapted to fit this study and allocates tasks in the order of ascending deadlines. At every step it looks for the most cost-effective allocation for the unallocated task with the earliest deadline without backtracking, hence assigning each task to the earliest available team without correcting. Available means here that it considers the time of previous task completion and the *Golden Bay*.

An example of 5 ordered tasks (A-B-C-D-E) with different r_i (green) and d_i (red) is depicted in Figure 11 and allocated by the following reasoning:

1. $A \rightarrow t01$: All teams are available at 0 minutes. Start task A straight away with the first time in order.
2. $B \rightarrow t02$: t02 is earlier available than t01 and next in order.
3. $C \rightarrow t02$: t03 is next in order, but it stays on the Golden Bay and t02 is still earlier available than t01. C cannot be started straight away because of r_C.
4. $D \rightarrow t03$: t03 is a Golden Bay team and is available at the start of the shift, so t03 can do task D.
5. $E \rightarrow t01$: t01 is earlier available than t02 and task E can be started right after A.

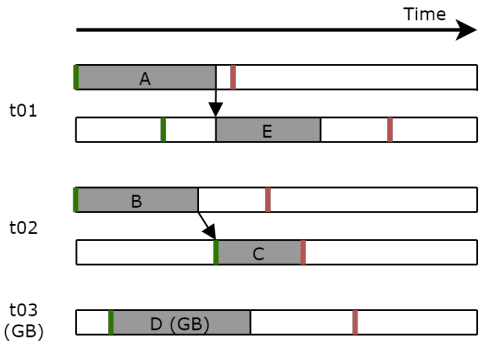


Figure 11: The initial task allocation according the EDF algorithm

This EDF algorithm only looks to allocate the task with the earliest deadline to a team, regardless of skill levels, transfer times, and workloads. The next step in this heuristic approach initiates from the schedule created by the EDF algorithm and aims to improve in all aspects of the model.

6.2 Tabu Search Algorithm

Because EDF provides a good initial solution when trying to minimize delays, it is assumed that the initial solution is not far off from an optimal solution.

A general outline of the tabu search algorithm

The basic concepts of a tabu search meta-heuristic has been covered in (Hillier and Lieberman, 2015, p. 625) and a brief explanation is repeated here. The application of tabu search includes a *local search procedure* tailored to the problem at hand. Note that a tabu search in itself is a meta-heuristic, which lays out the overall structure of the algorithm, but can be written such that it applies to many different problems.

Starting from an initial trial solution, it acts like a local improvement procedure trying to find the most *improving* solution at every iteration. Although is not required that every new solution should be better than the previous solution. After some iterations, it will logically arrive at a local optimum by continuously improving on the former solution, but then the key strategy kicks in. By allowing *deteriorating* solutions in the neighborhood of the local optimum, it moves away from the local optimum and arrives at a new point in the solution space where other better solutions can be found. This way, it is manages to escape local optima and eventually arrive at the global optimum.

The way it accepts deteriorating solutions is by means of a tabu list. This list records forbidden moves, otherwise called *tabu moves*. A tabu move is a move that cycles back to a local optimum and in order to escape this point in the solution space, such moves are be forbidden for a number of iterations. The only exception to a tabu move is when that move improves

on the best solution known so far during the search. This is also known as an aspiration condition.

A feature that can be incorporated is the concept of *diversification*. After some iterations, the tabu moves are removed from the tabu list and the algorithm can cycle back and forth between previously visited solutions. In order to discourage such cycling behavior, a method of diversification can force the search to unexplored areas of the solution space.

A local tabu search procedure for the scheduling problem

The algorithm is an adaptation of the tabu search method introduced by Cordeau et al. (2001). It is a local search heuristic that addresses *Vehicle Routing Problems with Time Windows* (VRPTW), which is very similar in its formulation to this scheduling problem. It will later be shown that this algorithm in fact does produce good results.

The method uses attributes to distinguish different solutions s. With each solution $s \in S$ is associated an attribute set $B(s)$=$\{(i,k)$: task i is visited by team $k\}$. An operator removes an attribute (i,k) from $B(s)$ and replaces it with an attribute (i,k'), where k != k'. The neighborhood $N(s)$ of solution s is defined as all possible attributes (i,k') for $B(s)$. This comes down to task i being removed from the schedule of k and inserted into the schedule of k'. At each iteration, the insertion is performed so as to minimize the value of the objective.

The objective is $f(s) = c(s) + \omega_1 t(s)$, where $c(s)$ is the original objective function of the MILMSP and $t(s)$ is the time window variation. A nonzero time window variation $t(s)$ implies an assignment of a task outside its time window ($B_i \leq r_i$ for any task or $B_i \geq d_i$ for break tasks). There are no workload variations included because there is no upper bound on the workload for teams.

Figure 12 shows the workings of the removal-insertion operator. The colors represent the teams as shown in the figure. In the example, $t01$ was assigned KLA-KLB, $t02$ was assigned KLC-KLD-KLE and $t03$ was assigned KLF, representing the current solution s (*Start* and *End* are omitted). The operator then removed (KLC, $t02$) from $B(s)$ and inserted it in $B(s')$, changing the assigned routes for $t01$ and $t02$, keeping $t03$ unchanged.

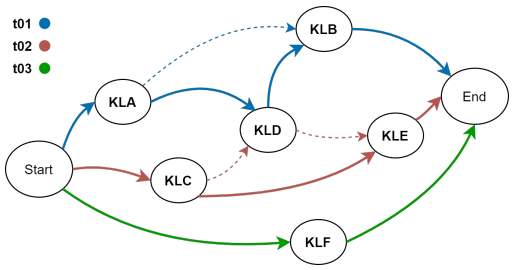


Figure 12: **Example** The removal-insertion operator generates a neighborhood solution. Attribute (KLD, $t02$) is removed and attribute (KLD, $t01$) is inserted.

Once the operator has generated a neighborhood solution s' with a minimal value for $f(s)$, the removed attribute (i,k) obtains a *tabu status*. The tabu status is defined as the reinsertion of attribute (i,k) for the next θ iterations.

In order to have a diversified search, any solution $s' \in N(s)$ such that $f(s') \geq f(s)$ is penalized by a factor proportional to the addition frequency of its attributes, and a scaling factor. Let ρ_{ik} be the number of times attribute (i,k) has been added to the solution during the procedure. Then, a penalty $p(s') = \lambda c(s')\sqrt{nm}\sum_{(i,k)\in B(s')}\rho_{ik}$ is added to every solution $f(s')$. The product $c(s')\sqrt{nm}$ serves as a correction factor proportional to the solution cost and the number of possible attributes in the entire problem. Additionally, the factor λ controls the intensity of diversification. The penalties drive the search process towards less explored regions of the search space. It is assumed that $p(s') = 0$ if $f(s') < f(s)$.

In short, the algorithm can be summarized as follows with the governing parameters listed in Table 5:

1. Set ω_1=1. If the initial solution s_{init} is feasible, set s^*=s and $c(s^*)$=$c(s)$; otherwise, set $c(s^*)$=∞.
2. For iteration = 1,...,η:
   - Choose a solution $s' \in N(s)$ which minimizes $f(s') + p(s')$ that is not tabu or that satisfies its aspiration criteria.
   - If s' is feasible and $c(s') \leq c(s^*)$, set s^*=s' and $c(s^*)$=$c(s')$.
   - Compute $t(s)$ and update ω_1 accordingly with δ.
   - Set s=s'.

This heuristic has been used in experiments by REF for a PVRP and MDPVRP, where by means of a sensitivity analysis it was determined that the best parameter settings were: $\delta = 0.5$, $\lambda = 0.015$, $\theta = [7.5 \cdot \log_{10} n]$. $[x]$ stands for the rounding of x to the nearest integer. These parameter settings were found again in Cordeau et al. (2001) and used in their experiments. Hence, in this study, these parameters are used as well. The number of iterations have been set at $\eta = 4 \cdot \theta$ in the verification process, as an estimated middle ground between number of tabu moves and iterations to reach the optimal solution.

Table 5: Tabu Search parameters

Parameter	Description
η	the number of iterations
θ	the number of iterations a move is tabu, i.e. the length of the tabu list
δ	the intensity of the change in weights a and b after each iteration
λ	the intensity of the diversification

7 Validation

The skill level of each team is based on the combination of regular KLM employees and contractors, or in other words, flex employees. There are 3 different skill combinations: *KLM/KLM*, *KLM/Flex*, *Flex/Flex*. In consultation with SME's at KLM, their skill multipliers have been set at: 1.0, 1.0, 0.9, respectively. In addition, there are a couple of teams that will be working on the Golden Bay, i.e. workstation 7.

7.1 Sensitivity Analysis

Setting model parameters

Before looking into the weighing of α, other parameters are set first. The model has 6 parameters that directly affect the objective function value. These are $\alpha, \beta, p_E, p_T, p_W$, and f_i. As a starting point, $p_E := 1$, meaning the penalty of a task increases by one, for every minute a task is finished late. To discourage tardiness of tasks, $p_T := 100$ and $f_i := 2$ to double the importance of starflights. To penalize workloads, $p_W := 5$. It has been reasoned that 1pp difference in workload on a 480 minute shift is 4.8 minutes of extra work/task delay, hence the value of 5. These parameters have been set in consultation with SME's at KLM Cargo and are used from the outset of this model validation.

Non-dimensionalizing the model objective

In a multi-objective optimization problem, single objective terms can best be compared when normalized to their ideal value. The ideal value of a term indicates the optimal value when only that term would be of interest. For a multi-objective minimization problem like the one formulated below:

$$\text{Minimize} \quad Z(\vec{x}) = \textstyle\sum_i a_i \cdot F_i(\vec{x}) \quad \forall\, \vec{x} \in S$$

where a_i stands for the weight of the $i^{th}-$term and S is the feasible solution space. Optimizing solely for F_1 by setting all $a_i = 0$ except for a_1, gives F_1*. For each model instance, the objective function can be non-dimensionalized by expressing each term as a fraction of its ideal value:

$$\alpha \cdot \frac{P_{L_{max}}}{P_{L_{max}}*} + (1-\alpha) \cdot \frac{P_{W_{max}}}{P_{W_{max}}*} + \beta \cdot \frac{R}{R^*} \tag{21}$$

As mentioned, the third term is a residual term and is only included to ensure proper convergence and output from the model. The purpose of β is to have a sufficiently small effect in order to not let R be the governing factor in the objective function. It is still desired to have $\beta \ll \alpha$, so the following approach was used, after identifying the maximum value for R for a single shift:

$$\begin{aligned} \max\{R\} &= \max_i\{B_i + P_{L_i}\} + \max_k\{P_{W_k}\} \\ &= 480 + 480 \cdot 100 \cdot 2 + 500 \\ &= 96980 \end{aligned} \tag{22}$$

Still this is a very conservative estimate since R is a sum of average values instead of maximum values. Formally, the first two terms in (21) are always ≥ 1. R^* is set equal to $\max\{R\}$ to never let the third term exceed 1.0. The upper bound of R is strongly affected by $P_{L_{max}}$ and therefore β is set to $1e^{-3}$ to ensure a marginal gain 3 orders of magnitude smaller than the other two terms. From here on, the first two fractions in (21) are referred to as F_1 and F_2, respectively.

It is up to the stakeholder to make a definite setting of weight α. However, some insight can be provided here. Typically in multi-objective optimization, there does not exist one feasible solution that minimizes all objectives simultaneously. The outcome of individual model objectives changes by its allotted weight, which can be shown on a Pareto plot. Solutions that cannot be improved in any of the objectives without deteriorating at least one other objective are called Pareto optimal solutions and are indicated by a Pareto front. The Pareto front is mathematically defined as the set of feasible solutions $\vec{x_1} \in S$ that dominates other solutions $\vec{x_2} \in S$ if both the following conditions are true:

1. $F_i(\vec{x_1}) \leq F_i(\vec{x_2}),\ \forall i \in \{1,2,...,p\}$
2. $\exists i \in \{1,2,...,p\}\ : F_i(\vec{x_1}) < F_i(\vec{x_2})$.

An example of a Pareto plot is given in Figure 13 for one model instance. The scheduling model actually has three model objectives which would create a 3-dimensional Pareto plane, but since the residual term is not of interest for the weight assignment, the two-dimensional projection of the Pareto plot is shown in Figure 13.

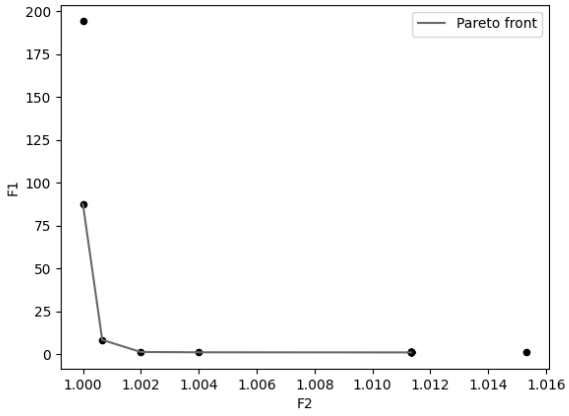


Figure 13: Pareto plot for scheduling of the evening shift of September 4th.

As shown, the two solutions on the outside are dominated by other solutions and are therefore not on the Pareto front. The point $[1.0113; 1.0]$ is actually the same for multiple solutions. Standing out from the Pareto plot is the large difference in range of values for F_1 and F_2. The small range of the workload penalty is attributed to it being a relative measure instead of an absolute measure, which is the case for the scheduling penalty. Additionally, the scheduling penalty increases much quicker once a task is delayed after its deadline. With the large value of p_T, $P_{L_{max}}$ can quickly increase to many times its optimal value if α is set to a low value. The small value of p_W and the fact that $P_{W_{max}}$ is measured relative to other penalties, explains in part why the change in F_2 is so subtle.

The changes should be interpreted relative to the range between ideal and worst values. Doing so, it can be seen that F_2 can be decreased by 86% relative to its range while only increasing the scheduling penalty by 0.13% relative to its range. So for a large collection of weights, F_1 can stay very close to 1.0. Because F_1 stays rather flat when approaching the ideal value of F_2 (two points at the top left) it suggests that the workload can be improved a lot starting from the worst case (bottom right). The reason can be explained as follows. In terms of the scheduling penalty, the optimal schedule assigns the tasks to be completed as early as possible. To do this, tasks cannot be given to one team, but need to be distributed as much as possible to start as early as possible. The more the tasks are distributed, the more equal the workload will be among teams. This explains why there are good solutions for both F_1 and F_2.

The Pareto plot also indicates that by decreasing the importance of workloads with 0.01%, changing $(1-\alpha)$ from 1.0 to 0.9999, $F1$ moves by 55% from its worst value while F_2 remains unchanged. Further decreasing the importance of workloads by 0.1%, changing $(1-\alpha)$ from 1.0 to 0.9990, achieves a value of F_1 of 96% lower than its worst value and an increase in F_2 by only 4.5% of its ideal value.

A method of selecting one of the Pareto Optimal solutions is formulated by Laplace's choice criterion in equation (23), described by Pereira et al. (2015). Laplace's criterion treats both objectives equally and in a minimization problem it looks for the Pareto Optimal solution with the lowest average cost. The fact that its an unbiased criterion is why its being used here. The results of the application on a subset of instances is given in Table 6.

$$\text{Laplace's criterion: } \min\{E(F1,F2) \mid \alpha = 0,...,1.0\} \tag{23}$$

Using the Laplace criterion results in different weights for different model instances. When taking both the median and mode of the α-column, it suggests a value of 0.1 for α. This value corresponds to the point $(1.002; 1.259)$ in Figure 13 and shows indeed a good consensus between both objectives.

Table 6: The Laplace criterion for multiple model instances

Model Instance	α	$(1-\alpha)$	F_1	F_2	Criterion
Sept 1st, evening	0.5	0.5	1.155	2.635	1.895
Sept 2nd, evening	0.1	0.9	1.000	1.03	1.014
Sept 3rd, evening	0.1	0.9	1.000	1.007	1.003
Sept 4th, evening	0.2	0.8	1.000	1.011	1.005
Sept 5th, evening	0.1	0.9	1.000	1.011	1.006
Sept 6th, evening	0.1	0.9	1.000	1.003	1.001
Sept 7th, evening	0.1	0.9	1.000	1.003	1.001

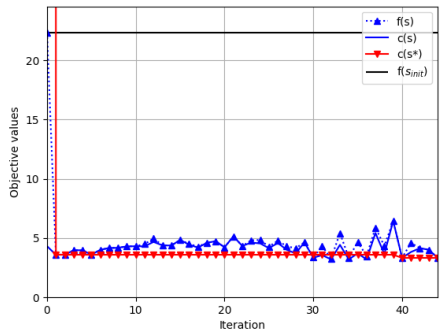
Computational Results

For the remainder of this section, the model results have been computed from a python script running on an Intel(R) Core(TM) i5-5200U CPU @ 2.20 Ghz with 4 GB of memory. The results for 28 different instances are shown in Table 8. It shows what kind of results can be expected and that the model results are in line with the weekdays, shifts and task- and team sizes. It can be seen that in general, there are more tasks in the evening shift than in the morning shift and the most demanding weekdays are Friday and Saturday. The delays are separated as early and tardy tasks. The early tasks are not completed at C_{min} but are done before the deadline. The tardy tasks are completed after the deadlines. As is reflected in the values of p_E and p_T, tardy tasks are penalized much more than early tasks. The average and standard deviation therefore only refers to the tardy tasks and shows the severeness of the delays exceeding the deadline.

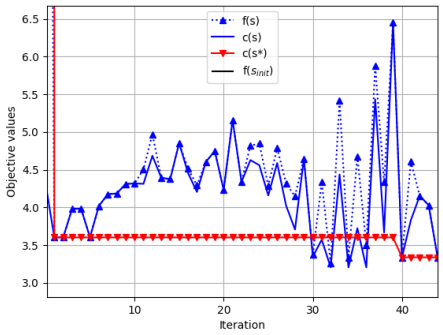
The shifts with the highest averages of delays are also the shifts with a high average workload. This shows that the busier the shift, the more difficult it is to schedule all tasks in time. Since scheduling is done on flight level and preemption is not allowed, complete tasks (<6 ULDs) cannot be separated as single ULDs in this formulation and poor results are inevitable. Furthermore, the delays are heavily dependent on the estimated release times r_i, team-dependent service times s_i^k, and task composition (ULD types and categories).

The occurrences of large standard deviations for workloads are, apart from the the Saturday morning shift on the 12th, the result of not being able to assign tasks to Golden Bay teams. This means that other teams take most of the work while the Golden Bay are only assigned to one or two tasks, greatly impacting the standard deviation. It should be noted that the shifts with few tasks on the Golden Bay are the evening shifts. When looking at the small standard deviations for workloads, it shows that there are plenty of tasks to be assigned to Golden Bay teams and there is much room for shifting around tasks to obtain a fair workload distribution.

Then the tabu search algorithm has been run on the model instance of the evening shift of September 4th. Figure 14 shows the convergence of the model towards the optimal solution. The generated schedules as initial solution and that of the final solution are shown in Figure 15.
What can be noted about Figure 14 is that the initial solution is not immediately feasible as can be verified quickly from the allocation of the break task (black) for team $t06$ in Figure 15a. The break task should be started before 20.30 o'clock, or 390 minutes into the shift. The time window variation is therefore nonzero for the initial solution and hence it is said to not be feasible. The first solution with no time window variation would be set as the first optimal solution since it would be the first feasible solution. Note that the difference between $f(s)$ and $c(s)$ is the time window variation of the solution. After the first iteration, a feasible solution is found and saved as the optimal solution $c(s^*)$. Note that better solutions seem to be found around iterations 30-40, but these are not feasible since $f(s) > c(s)$, hence they are not accepted as new optimal solutions. An improved optimal solution is only found at the 40^{th} iteration, which ultimately is the overall optimal solution.

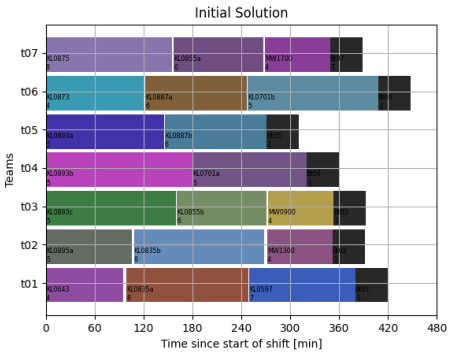


(a) Overall

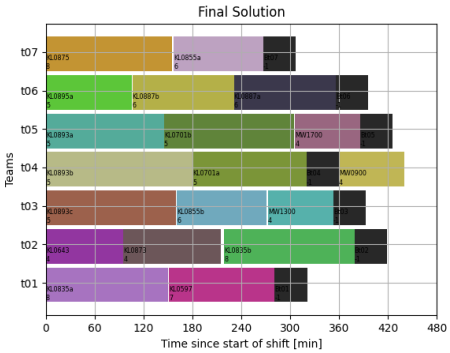


(b) Zoomed-in

Figure 14: Tabu search performance for scheduling the evening shift of September 4th. $\delta = 0.5$, $\lambda = 0.015$, $\theta = 11$, $\eta = 44$



(a) Initial Solution



(b) Final Solution

Figure 15: Schedules of the initial and final solution of the model when computed with the tabu search algorithm

The peaks in the graph are getting higher as the algorithm is trying to diversify its search by accepting other solutions after being penalized for repeating and non-improving moves. As a result, the model eventually moves from the worst solution (excluding $f(s_{init})$) to the overall optimal solution in a single iteration. This is the effect of δ and λ which penalize the value of $f(s)$ for repeating the same insertion moves. This example model instance shows how the tabu search goes from the initial solution to finding an optimal solution for a real shift. The effectiveness of the EDF is clear since the optimal solution found on the first iteration is not improved until the 40^{th} iteration.

To conclude, Table 7 shows the average computation times for the model instances from Table 8. Again it shows that due to a larger model size for morning shifts, it takes longer for the tabu search to compute each iteration. On average it takes almost half an hour to run all iterations of the morning shift and less than two minutes to run all iterations of the evening shift. When using the branch-and-bound method of the licensed solver, it takes much less time. However, this does not include the effectiveness of the EDF-algorithm of the tabu search which already provides good solutions on early iterations.

Table 7: Computation time of both solution methods for both sets of model instances

Solution method	Instance set	Avg. time per iteration	Avg. total time
Tabu Search	Morning shifts	29.31 sec	25 min 41 sec
	Evening shifts	2.86 sec	1 min 58 sec
Branch & Bound	Morning shifts	-	5 min 3 sec
	Evening shifts	-	19 sec

8 Conclusion

This paper has presented a model to improve on the workforce scheduling process of the buildup shift. This section will conclude this paper with answers to the research questions and recommendations for KLM and future research.

The risk of flight delays is quantified as a linear relation between completion time and severeness of delay. A piece-wise linear penalty function increments the penalty for every minute a task is completed later than its minimum completion time. The penalty increases substantially in case task completion is scheduled after the task deadline, hence a delay. The method of allowing a penalty instead of a hard schedule deadline is called using soft constraints and allows the model to always be solvable.

The workload distribution is quantified by a linear relation between difference from the mean and severeness of uneven workload distribution. A linear penalty function increments the penalty per team for every percentage point of workload difference from the mean. The slope is not increased for workloads over 100% because it is not the absolute workload that is of interest, but the difference in workload. There is no upper bound on the workload to allow the model to always be solvable.

For both performance metrics the model weighs the maximum penalty cases or bottlenecks. This means that the model will minimize the worst occurring cases among delays and workload differences.

The trade-off between both objectives is done by weight α. By looking at the 2-dimensional projection of the Pareto front, it is shown with the Laplace criterion that setting it to 0.1 gives a good average value to consider both objectives. It became clear that the scheduling penalty is much more sensitive to the weight than the workload penalty and that there is a strong relation between both objectives.

The tabu search proved to be an effective method in optimizing model instances when not working with licensed software. The EDF-method for finding initial solutions proved to be a good starting point from which the algorithm could progress. Although the branch-and-bound method of the commercial solver shows a faster convergence to an optimal solution, the tabu search showed that it can find an incumbent solution quickly and that incumbent solution could already be close to the overall optimal solution. Comparing the computation times of both solution methods with the current time required of 0.5 - 1.5 hours, it shows that this model is a faster way to schedule.

Not only does it take less time, from the moment it starts computing there is no input required from a user, which also frees up time. Once the model has provided a schedule, the shift leaders can start examining the optimal schedule in terms of scheduling risk and workload penalties and also the bottlenecks. The schedule can then be used as a starting point for the allocation of the next shift. Over time the model can be tuned and the

Table 8: Validation results

Model Instance					Delays				Workloads	
Day	Weekday	Shift	Tasks (gb)	Teams (gb)	Early	Tardy	Avg.	St. Dev	Avg.	St. Dev
01	Tue	Morning	31 (5)	14 (1)	12	2	48	43.4	50%	0.9
02	Tue	Evening	13 (2)	7 (1)	0	0	0	0	45%	9.7
02	Wed	Morning	35 (7)	16 (2)	8	0	0	0	53%	4.0
02	Wed	Evening	13 (0)	7 (1)	6	1	1	0	47%	19.3
03	Thu	Morning	33 (9)	14 (1)	6	0	0	0	50%	5.4
03	Thu	Evening	10 (0)	7 (1)	0	0	0	0	39%	16.8
04	Fri	Morning	36 (8)	17 (1)	7	5	66	72.2	47%	9.3
04	Fri	Evening	19 (1)	7 (1)	2	11	84	85.2	71%	37.2
05	Sat	Morning	41 (7)	15 (2)	5	13	92	49.5	67%	8.2
05	Sat	Evening	16 (0)	7 (1)	4	7	27	19.0	57%	36.3
06	Sun	Morning	32 (6)	14 (1)	7	0	0	0	55%	2.8
06	Sun	Evening	14 (0)	6 (1)	0	0	0	0	51%	23.5
07	Mon	Morning	32 (6)	15 (1)	10	0	0	0	48%	4.5
07	Mon	Evening	16 (0)	5 (1)	3	9	97	55.4	76%	38.0
08	Tue	Morning	32 (7)	15 (1)	4	0	0	0	48%	1.6
08	Tue	Evening	12 (1)	8 (1)	3	2	53	37.3	46%	21.0
09	Wed	Morning	44 (11)	17 (1)	9	9	198	129.5	65%	3.2
09	Wed	Evening	17 (0)	8 (2)	1	11	90	59.6	61%	47.5
10	Thu	Morning	33 (10)	18 (1)	5	0	0	0	43%	6.0
10	Thu	Evening	15 (0)	7 (1)	5	5	71	45.8	57%	37.0
11	Fri	Morning	41 (7)	17 (1)	6	13	134	86.6	59%	4.1
11	Fri	Evening	21 (1)	7 (1)	4	10	87	50.2	83%	20.3
12	Sat	Morning	55 (8)	17 (2)	7	27	241	128.4	80%	22.5
12	Sat	Evening	23 (0)	7 (1)	7	11	174	93.5	85%	54.6
13	Sun	Morning	44 (11)	17 (2)	10	13	80	77.9	66%	8.4
13	Sun	Evening	13 (0)	7 (1)	4	4	20	30.2	50%	32.2
14	Mon	Morning	45 (11)	16 (2)	16	14	127	85.3	58%	9.7
14	Mon	Evening	14 (0)	7 (1)	2	7	23	26.7	51%	32.9

recommended setting of weights can be altered. Tasks that are scheduled to be tardy shall be known in advance. Especially the indication of tardy starflights can spark additional action from early on. Some tasks are inevitably tardy due to their short time window. As a result, since deadlines are determine by flight departure, release times can be investigated to allow more time to build each task. This shows how the model can help uncover bottleneck situations.

Although the model also allows to distribute workloads evenly among teams, excessive workloads will occur on busy shifts. Teams that are likely to have more work than their employability allows can be identified. They can be aided by breaking down tasks further than has been done here (on ULD level) and distribute parts of their work to other teams when possible.

What should be remembered is that this model represents a simplified reality. Numerous assumptions have been made among which are levels of skill, build times, release times and transfer times, to name but a few. Moreover, even the assumption of a linear relation between the completion time and its effect on the severeness of delays and the linearly increasing effect on dissimilar workloads is a simplified estimation.

On the other hand, it must be stated that a scheduling model as it has been implemented now is not fully attainable. During validation the model has been used in back tests, assuming a real-time data connectivity with the operational database Chain at the time of scheduling. In addition, information has been used that could only be found from data points that would not yet be available at the time of scheduling. The distinction between a LF, BB Build, BB Check and T-ULD is made by looking at what has been logged up until flight departure. Since the categorization is needed for the work estimation, it is essential that this information is visible early before the shift or the work starts. A separate model using a learning algorithm on the data available at the moment of scheduling would be an interesting addition to this research. It has been considered during this project, but that would have extended the timeline considerably.

There are many different kinds of pallets and KLM makes a distinction between 3 types (MDP, LDP, AKE) to assume the required processing time of ULDs. No further distinction has been made in the types of ULDs, but the model took advantage of the available data and labeled ULDs that were already partly processed. If shift leaders want to take this into account, they need to look a few layers deeper in to Chain in order to find such information, if there is even time to do it, such that the ULDs in progress can be taken into account.

For future work it is recommended to use chance constraints to incorporate the stochastic nature of many input parameters in this model. These can be incorporated for release times and build up times. The restriction of Golden Bay teams not leaving that bay could be lifted and new results could be computed to more closely represent an actual shift. Initially, the teams do not move from the Golden Bay, but once help is needed on another bay to prevent delays, Golden Bay teams would help out as well. The sensitivity of the model has not been tested for changes in service time, team skills, and employability. It would add to the overall applicability and understanding of the sensitivity of this model to further research these aspects. With a more detailed model, incorporating the aforementioned subjects, the model could be used to make recommendations about the precise operations occurring during buildup. The model could be used in the assessment of the benefits of hiring more employees or the consideration of providing additional training to employees to raise their skill level.

The parameters of the tabu search have been copied from the extensive research done by Cordeau et al. (2001). It is however, an adaptation and in this project, the magnitudes of transportation times and processing times are the opposite of a traditional vehicle routing problem. Therefore, these parameters could be revised and another rule for stopping the tabu search could be tested. Right now it stops after $\eta = 4 \cdot \theta$ iterations as was determined during verification on smaller instances, but a more robust method can be developed.

References

Beck, J. C., Prosser, P., and Selensky, E. (2003). Vehicle Routing and Job Shop Scheduling: What's the Difference? *ICAPS*, pages 267–276.

Bell, W. J., Dalberto, L. M., Fisher, M. L., Greenfield, A. J., Jaikumar, R., Kedia, P., Mack, R. G., and Prutzman, P. J. (1983). Improving the Distribution of Industrial Gases With an On-line Computerized Routing and Scheduling Optimizer. *INFORMS Journal on Applied Analytics*, 13(6):4–23.

Cattrysse, D. G. and van Wassenhove, L. N. (1990). A Survey of Algorithms for the Generalized Assignment Problem. Technical report, Erasmus University Rotterdam.

Chang, G. J. and Ho, P.-H. (1998). The β-assignment problems. *European Journal of Operational Research*, 104(3):593–600.

Cordeau, J., Laporte, G., and Mercier, A. (2001). A unified tabu search algorithm for vehicle routing problems with soft time windows. *Journal of the Operational Research Society*, 52(8):928–936.

Cordeau, J. F. (2006). A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54(3):573–586.

Delgado, F., Trincado, R., and Pagnoncelli, B. K. (2019). A multistage stochastic programming model for the network air cargo allocation under capacity uncertainty. *Transportation Research Part E: Logistics and Transportation Review*, 131:292–307.

Desrochers, M. and Laporte, G. (1991). Improvements and extensions to the miller-tucker-zemlin subtour elimination constraints. *Operations Research Letters*, 10(1):27 – 36.

Desrochers, M., Lenstra, J. K., and Savelsbergh, M. (1990). A classification scheme for vehicle routing and scheduling problems. *European Journal of Operational Research*, 46(3):322–332.

Duin, C. W. and Volgenant, A. (1991). Minimum deviation and balanced optimization: A unified approach. *Operations Research Letters*, 10(1):43–48.

Ernst, A., Jiang, H., Krishnamoorthy, M., and Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153(1):3–27.

Geetha, S. and Nair, K. P. (1993). A variation of the assignment problem. *European Journal of Operational Research*, 68(3):422–426.

Gendreau, M., Hertz, A., Laporte, G., and Stan, M. (1998). A Generalized Insertion Heuristic for the Traveling Salesman Problem with Time Windows. *Operations Research*, 46(3):330–335.

Graham, R. L., Lawler, E. L., Lenstra, J. K., and Kan, A. H. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326.

Hillier, F. and Lieberman, G. (2015). *Introduction to operations research*. McGraw Hill.

Jackson, J. (1955). *Scheduling a production line to minimize maximum tardiness*. Research report. Office of Technical Services.

Janiak, A. and Kovalyov, M. Y. (1996). Single machine scheduling subject to deadlines and resource dependent processing times. *European Journal of Operational Research*, 94(2):284–291.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(2):231–247.

Mulla, A., Raravi, G., Bose, R. P. J. C., Rajasubramaniam, T., and Dasgupta, K. (2016). Efficient Task allocation in Services Delivery Organizations. *Proceedings - 2016 IEEE International Conference on Services Computing (SCC)*, pages 555–562.

Oyetunji, E. O. (2009). Some Common Performance Measures in Scheduling Problems: Review Article. *Research Journal of Applied Sciences, Engineering and Technology*, 1(2):6–9.

Pentico, D. W. (2007). Assignment Problems: A golden anniversary survey. *European Journal of Operational Research*, 176(2):774–793.

Pereira, J., Ekel, P., Palhares, R., and Parreiras, R. (2015). On multicriteria decision making under conditions of uncertainty. *Information Sciences*, 324:44 – 59.

Ravindran, A. and Ramaswami, V. (1977). On the Bottleneck Assignment Problem. *Journal of Optimization Theory and Applications*, 21(4):451–458.

Scarelli, A. and Narula, S. C. (2002). A Multicriteria Assignment Problem. *Journal of Multi-Criteria Decision Analysis*, 11(2):65–74.

Shen, M., Tzeng, G.-H., and Liu, D.-R. (2003). Multi-Criteria Task Assignment in Workflow Management Systems. *Proceedings of the 36th Annual Hawaii International Conference on System Sciences, 2003*, pages 9–pp.

II

Literature Study
previously graded under AE4020

1 Introduction

1.1. Background

This research project will be concerned with the buildup phase in freight building 3 of KLM Cargo. Specifically, it is the scheduling of work for the buildup phase that is subject of this research. During the buildup phase, teams are assigned to flights for which they need to build up ULDs that need to be on outgoing intercontinental flights. Flights are always prepared at a predetermined workstation. The teams are tasked with completing the buildup phase of all ULDs in time before flight departure.

At the moment, it is done manually by team coordinators and shift leaders while simultaneously looking at 4 different documents and an operational handling system dating back decades ago. A new schedule for the buildup phase needs to be made before the start of each of the three daily shifts (6 a.m. to 14 p.m., 14 p.m. to 10 p.m., and 10 p.m. to 6 a.m.). Due to inefficient planning, shipments miss their deadline and are not built up in time. Flights are sometimes even forgotten and up until the final 15 minutes before the shift it is not certain if everybody shows up. This makes last minute planning even more difficult and it often becomes work during the shift itself. On average it takes 0.5 to 1.5 hrs per shift to allocate and adjust the teams. The shift leaders have reached out to the department of Performance Management to look for a data driven solution. A model needs to be developed that determines an optimized schedule considering the distribution of workload and the flight departure times.

The project will be carried out as a Master's thesis assignment of a student of the TU Delft department of Air Transport Operations. This project will be supported by Paul Roling and Alessandro Bombelli from the deparment of Air Transport Operations at TU Delft and Bas Schmidt from the Performance Management department at KLM Cargo.

1.2. Research Objective and Context

> *The research objective is to build a model for the KLM Cargo Hub to schedule workforce at buildup aiming at reducing the ULDs missing their booked outgoing flights **by** writing a linear program that determines the optimal allocation of staff to workstations for the upcoming shift.*

The project is done at the department of Performance Management, where there is access to operational performance data and access to the work floor of the hub itself. Additionally, the project will focus solely on the algorithm and not on the implementation of it in the form of a computer application. The algorithm will not only contribute to the automation of the scheduling process itself. An automated system could potentially save hours of work spent before and during each shift and deliver a more consistent schedule where workloads are balanced and schedule risk is mitigated. To conclude, it can be stated that the research objective is indeed useful, realistic, feasible, clear and informative.

From an academic standpoint considering earlier research, there is no scheduling model that faithfully describes and incorporates all the constraints and features of this problem of interest, i.e. assigning teams to ULD workstations in the warehouse of an air cargo supply chain Hub. This research fills up that gap by developing a scheduling model.

1.3. Research Questions

The project context explored at KLM, led to two KPIs for the scheduling model: the number of missed ULDs and the equality of workload among teams. It leads to the following main research question and sub-questions:

To what degree can a workforce scheduling model for buildup improve scheduling performance?

1. How can the workforce scheduling model be formulated?
    - What team attributes and operational characteristics can be de-fined?
    - What are the costs associated with delays and how can they be modelled?
    - To what extent does additional workload impact the schedule and how can it be modelled?
2. How does the model perform on the defined KPIs for an upcoming shift?
    - How can risk of delay per flight be quantified?
    - How can workload distribution among teams be quantified?
    - What is the trade-off between delays and workloads in schedule performance?
3. How can the schedule be optimized fast enough such that it can be used on a normal day of operation?
    - What kind of simplifications must be applied?
    - What is the consequence of these simplifications?
    - What is an acceptable time span in which a schedule can be created?
4. How effective is the scheduling model compared to the current manual approach?
    - What are the advantages and disadvantages of the scheduling model?
    - What is the difference in schedule risk?
    - What is the difference in the workload distribution among teams?

1.4. Report Structure

Other combinatorial optimization problems will be reviewed to assess the applicability in this project context. Starting with assignment problems and thereafter scheduling problems. Then methods of optimization will be discussed. Heuristic methods can be created to not be reliant on commercial software and some interesting examples will be reviewed. Finally, a light will be shed on the stochastic nature of practical problems.

2

Literature Study: Assignment Problems

Assignment problems are about finding optimal pairings of agents and tasks depending on a suitability function. [20] developed the Hungarian algorithm as the first practical solution method to solve the classic assignment problem (AP). Originally, the AP deals with one-to-one assignments and involves assigning jobs to workers or machines. Allowing more assignments to one agent is known as the generalized assignment problem (GAP). [24] provides an overview of the many variations of the assignment problem and a few will be discussed here, including additional literature.

Even though this project context would immediately classify as a GAP since there are more flights or ULDs than there are teams, there have been interesting contributions made to APs. [4] formulates an AP with a binary qualification criterion per agent-task combination. An extension of this can be found in [3] where a binary indicator specifies which of 3 different skills belong to an agent or are required for a certain task.

[25] presents the bottleneck AP. This particular problem is not aimed at minimizing the total costs, but minimizing the most costly assignment. [12] studied two variations of the bottleneck AP.One is minimizing the difference between the least and most costly assignment. The other minimizes the difference between the maximum and average assignment cost. An example is given of minimizing the idle time of simultaneously operating machines. These bottleneck problems could be suitable for balancing the workloads. The aforementioned variations have also been applied to the GAP by [6], but the agent's capacity limits are not taken into account, which is not applicable at KLM. The model was written as a bipartite graph. [22] goes further and combines the aforementioned model characteristics and defines hard constraints in terms of upper and lower bounds on workload, utilization and productivity while being aware of fairness among the employees. Their ILP-based allocation is compared with the manually assigned transactions by counting violations of the baseline, which is the expected completion time of each transaction. A similar approach could be used by taking the flight deadlines or expected ULD completion times as baselines.

2.1. Assignment problem objectives

The objective can consist of multiple criteria. [13] studied a bi-criteria bottleneck AP. Next to the assignment cost, their model assumes an additional supervisory cost as a bottleneck objective, all expressed in one function. [26] consider the problem of assigning referees to football matches with multiple independent selection criteria. By using (in)compatibility and priority indices their method is able to include indifference, preference and veto thresholds. Equivalently so at the KLM, there are such qualitative factors that need to be taken into ac- count. [27] selects appropriate individuals for tasks based on a multi-criteria assessment of individual suitability, social relationships and existing tasks. Next to workload balancing, it focuses on qualitative impacts including specialization of labor and job enrichment. The concept of triangular fuzzy numbers is introduced which represents imprecise linguistic terms (e.g. "good" and "bad") as real numbers.

[29] shows a sequential approach in multi-objective problems based on the priority of the objective criteria. Each function that is optimized generates a set of optimal solutions which go through to the optimization of the next function, and so on, until one optimal solution remains.

2.2. Machine scheduling

[16] gives an overview and introduces a classification of different machine scheduling problems in a workshop. Unrelated and uniform parallel machines would not be suitable for this project but uniform parallel machine scheduling would. It assumes a job as a single operation and the processing time depends on the speed factor of each machine. Job shop, ow shop and open shop are problems in which a job consists of multiple operations and each machine is dedicated to one or more operations. This is not relevant for this project. Job preemption and precedence are reviewed as well.

[23] discuss common performance measures in scheduling problems, like total completion time and makespan. [19] looks at resource dependent processing times, which is an application of the critical path method where allocating more resources, or another form of extra cost, reduces processing time. The timing or sequencing of tasks in machine scheduling problems is often done by allocating a machine with a task to a time slot. The advantage is that it is very easy and intuitive to obtain a schedule from the solution. The drawback however, is that the schedule works with minimal steps in time equal to the length of a time slot.

3

Literature Study: Routing formulations in scheduling problems

[5] identify vehicle routing in distribution systems as applications of the GAP. Vehicle routing problems (VRP) are not modelled as bipartite graphs as done by [Chang and Ho, 1998] where edges represent an agent-task assignment, but graphs in which edges represent a relation between the nodes/tasks (e.g. distance). The VRP is a generalisation of the traveling salesman problem (TSP), which does not typically include capacity limits, but has a similar formulation. [21] provides an overview of VRPs with different side constraints, among which are time windows on nodes and precedence constraints, which arise in this project context as well. The shipments belonging to a flight or to a ULD can only be build up when they have passed the breakdown phase and each flight has a deadline which would reflect the upper bound of the time window. [11] introduce, like [16] did for machine scheduling, a classification scheme for vehicle routing and machine scheduling problems together with examples. Variations discussed previously for the AP/GAP are also possible for the VRP, because it is an extension of the former. Bottleneck constraints and vehicle characteristics (e.g. team skills) are included in the problem described by [2], which shows that the VRP can include such characteristics too.

A comparison is made by [1] between a vehicle routing and job shop scheduling as a representation of a real world scheduling problem. Both problems can be reformulated to represent the other and both problems are solved in different ways. It is found that routing technology is superior when the problem involves optimizing the sum of total transition times, few precedence constraints and low resource specialization. This is in line with the project context. At KLM, there is low resource specialization because there will be multiple teams capable of executing the same task/flight and there are generally a few starflights which require precedence over other flights.

By comparison with scheduling problems using time slots, vehicle routing problems are scheduling problems which use the sequence of visited nodes as their decision variables. The time at which each node is visited then fol- lows from the travel times between nodes and processing times at the nodes, which could be seconds, minutes. This allows for more flexible planning than when using time slots. When looking at the size of the models, 3-index ow-based routing/scheduling problems assign an edge to a vehicle that will traverse it (X_{ij}^{k}). The problem size will therefore scale by n x (n - 1) x k, n being the number of tasks and k the number of teams. Scheduling problems with time slots assign a team to a task at a time slot and the problem size scales by n x k x t, with t the number of time slots. With the number of nodes assumed equal, the size depends on the difference between n - 1 and t. Assuming that on a typical day at the KLM Cargo Hub, there are 60 flights or 500 ULDs to be scheduled. n - 1 is then 59 or 499 but when scheduling in terms of minutes, t becomes 1440, which is an order of magnitude difference. Therefore, for this project context and project size, the VRP formulation makes more sense.

4

Literature Study: Optimization Techniques

When working with a linear problem containing binary and integer variables, commercial optimizers such as Gurobi will use a branch and bound or branch- and-cut technique to optimize these problems. Some lessons can be learnt in terms of reducing model size in order to achieve a faster optimization. In turn, many heuristic methods have been developed tailored to specific problems in order to find good-solutions in a shorter time. Since computation time is of interest in practically-oriented projects like this one at KLM, some papers on heuristic methods will be discussed here.

4.1. Branch-and-bound algorithm

Branch and bound is a solution technique used for integer programming problems where the total set of solutions is partitioned into smaller subsets of solutions [18]. It begins by first solving the problem as a linear programming model without integer restrictions. This is called an LP relaxation of the problem, since the restrictions are relaxed. The optimal solution of the relaxed problem can therefore lay in between the rounded solutions of the integer problem. This solution designates the upper bound of the problem. From there, a diagram of nodes and branches is formed. A new node stands for a subset of the problem exploring a different part of the feasible region. For some nodes (regions) it can be determined that no improvement can be obtained from the best solution found up until that point. It is then decided to leave that node and continue on other branches until the optimal integer solution is obtained.

A variety to the branch-and-bound is the branch-and-cut approach. It involves a branch-and-bound and using the cutting planes method to refine the feasible region by means of linear inequalities. According to [18], the approach uses mainly 3 kinds of techniques. Automatic problem preprocessing, generation of cutting planes and branch-and-bound techniques. The preprocessing comprises of fixing variables at one of their possible values since the other values cannot possibly be part of the solution. Another category is eliminating redundant constraints which are automatically satisfied by other constraints. The last category is tightening constraints to reduce the feasible region for the LP-relaxation of the MILP problem. The latter is also an example of a cutting plane technique. A cutting-plane can also be generated by finding a minimum cover of a constraint, which is a set of binary variables that once one of the variables changes from 1 to 0, the constraint becomes satisfied. There are a myriad of cutting planes techniques, but fortunately commercial software can help.

A commercial optimizer such as Gurobi is able to apply presolve and cutting planes techniques automatically [17]. The former is used to reduce the size of the problem, similar to eliminating redundant constraints in the branch-and-cut approach. This is nice from a practical standpoint, however it is not desirable from a research standpoint, since the way of optimization is not entirely transparent to the user. Therefore it may be a good idea to incorporate user-defined-cuts or valid inequalities to enforce the cuts yourself without eliminating the feasible integer solutions.

[7] presents an example of valid inequalities for the dial-a-ride problem with time windows. It states that the lower bound of a time window can be increased depending on the previously visited node. There is an earliest time of arrival from the previous node depending on the lower bound of the time window of that previous node and its shortest possible service time. This means that an allocation of a vehicle to two nodes immedi-

ately affects the width of the time window and therefore tightens the constraints concerning the beginning time. A similar logic can be applied to the upper bound of the time window. There is a latest time of leaving a node depending on its service time and the upper bound of the time window at the previous node. Similarly, this can reduce the width of the time window, hence tightening the constraint.

Using a large constant to incorporate an if-else clause in the constraints is named as the big-M method in [18]. A large constant is also used by [7] in its constraints. It is emphasized in the paper to put a bound on those constants in order to avoid computational problems. It is therefore important to assign a value just small enough depending on the other parameters and coefficients in the functional constraint. Another valid inequality that is formulated are subtour elimination constraints. These are constraints that prohibit a vehicle to take a smaller tour that is part of a larger tour. In the VRP, this would imply that tours without starting and ending at the depot are not allowed, which is different from the classic TSP, where tours that do not visit all nodes are prohibited. However, this example of a subtour elimination constraint for the VRP is different than that for the dial-a-ride problem in [7] . The VRP subtours are a requirement in the original formulation and should not be used as a cutting-plane technique to reduce the feasible region of the relaxed problem in preprocessing. The paper also presents variable fixing, as described before by [18] and arc elimination as a way of preprocessing the model in the branch-and-cut algorithm. Arc elimination occurs when two nodes cannot be paired from the outset because of time windows, pairing and ride-time constraints and their arc is then eliminated. The most obvious example would be the arc between the origin hub node and the destination hub node. Other examples in this paper are more related to the dial-a-ride problem, but the time windows of flights and position of workstations can be considered in order to eliminate arcs and reduce the problem size.

4.2. Initial solutions

Finding an initial solution can speed up the optimization process for large problems considerably by setting a good starting point. Depending on the problem objective, one initial solution method is better than the other. One method for a VRP problem is proposed in [8] by ordering the nodes by the angle made with the depot in a Euclidean coordinate space. This attempt at a solution is focused on trying to limit the travel time between nodes. A random node is selected as the first node and the first arc is then made by adding a new node in the predefined order until load or duration constraints are violated for that route. Once that happens, a new route is initiated. The nodes/ flights in KLM's case could be ordered in that way, but the travel time is less of a concern since the service time at any node greatly exceeds the travel time between nodes. Therefore, another ordering technique or approach for an initial solution more tailored towards the situation in the KLM Hub would be better suited.

Based on the article of [8], a simple analogy can be made for KLM. An example of an important factor that can be considered for the initial solution is the deadline of each flight. By ordering all flights from earliest deadline to the latest, assign each flight to the team that has the earliest availability, and do the same for the next flight. This focuses on minimizing the initial completion time of each flight and with that the delays. Both this example and the aforementioned paper use a greedy method to choose the next node, which is a locally lowest cost choice at each stage.

4.3. Heuristics

Heuristics are methods of problem-solving that are commonly used to find good feasible solutions that are reasonably close to the optimal solution. A metaheuristic is a general method that provides a structure and strategy guideline for developing a heuristic for a particular type of problem. It is among the most important tools used in Operations Research. Two examples of metaheuristics will be discussed, the tabu search and large neighbourhood search. Other examples are simulated annealing and genetic algorithms. Based on the type of problem at hand one is more useful than the other.

[18] explains that, like many metaheuristics, simulated annealing and the genetic algorithm are derived from natural phenomenon. Simulated annealing is a local search method governed by a temperature variable that controls the selection of new solutions. It is incorporated in the paper on large neighbourhood search that will be presented later in this section. The genetic algorithm is particularly good at exploring various parts of the feasible region and finding solutions which are not local. This means that, if a good initial solution can be generated, the optimal solution should be located somewhere in the neighbourhood and thus genetic algorithms are not further explored in this review. The tabu search and large neighbourhood search will be

discussed here along with a post-optimization method that compliments the tabu search.

4.4. Tabu search

[8] present a unified tabu search heuristic for the VRP with time windows and the periodic and multi-depot VRP with time windows. The algorithm consists of a local search metaheuristic and a diversification mechanism to explore a broader portion of the solution space. The search includes solutions that only need to satisfy two constraints. These are that every route should start and end at the depot and each node is visited by only one vehicle. Infeasible regions may be therefore be included in the search since time window, load and duration constraints are not considered during the search and can be violated for some solutions. A cost function consisting of the original objective and the sum of the aforementioned violations is used to evaluate each new solution. Each violation has a positive multiplier that facilitates the exploration of the solution space, in particular for tightly constrained problems. The tabu search works with a set of attributes $(i;k)$ for each solution $B(s)$ that tracks if node i is visited by vehicle k or not. The neighborhood $N(s)$ of a solution is defined by applying a simple operator that removes an attribute from $B(s)$ and replaces it with $(i;k')$ where $k \neq k'$. This means that one node is removed from route k and inserted into route k'. Route k is then restored by connecting the predecessor and successor of node i.

The diversification is encouraged by prohibiting the reinsertion of i into route k for a number of iterations, which characterizes the tabu search. It is given a tabu status, which is typical for a tabu search method. An aspiration criterion can revoke the tabu status if that would allow the search to reach a solution of smaller cost. A solution in the neighborhood $N(s)$ of another solution with a larger cost function is penalized by a factor proportional to the addition frequency of its attributes and a scaling factor depending on the model size. This penalty also drives the search to less explored regions of the solution space whenever a local optimum is reached because the presence of attributes in that solution that have been inserted before increase the penalty of that solution. The intensity of the diversification can be controlled by a parameter.

Although only feasible solutions can be accepted as an optimal solution, the search can make a step towards an infeasible solution if that solution has the lowest cost in the neighbourhood. The scaling parameters of the violation of time window, load and duration constraints in case of infeasible solutions are adjusted after each iteration. This is repeated for a number of iterations and then the best feasible solution is post-optimized according to the specialized heuristic by [15].

This tabu search heuristic is proposed for the vehicle routing problem with time windows (VRPTW) and two generalizations: the periodic vehicle routing problem with time windows (PVRPTW) and the multi-depot vehicle routing problem with time windows (MDVRPTW). It is stated that this method is thrives in its simplicity because it is based on a limited number of parameters and a simple scheme, compared to alternative approaches. When tested on real-life instances from a German grocery distributor, using 800-1000 nodes and 30-40 vehicles in the different instances, it took 20000-25000 iterations to arrive at good solutions, which was also attributed to the post- optimization. It did take upwards of 2 hours to complete 10000 iterations, which shows the time-intensive effort of this algorithm. For KLM, the problem size will be much smaller and more comparable to earlier instances tested in this paper with 54 vehicles. The algorithm is also tested on the many instances of the VRPTW originally studied by [28]. The C1 and C2 problems contain clustered nodes instead of randomly distributed, which is comparable to KLM's case because several nodes are on the same work- station. In most of cases, with 100 nodes and 50-60 vehicles, good solutions were obtained after 105 iterations which took 20-60 minutes to compute.

4.5. Post-optimization of the tabu search

[15] extends the ideas of [14] by creating a post-optimization procedure for the traveling salesman problem with time windows. [14] introduces the GENIUS heuristic procedure that is developed for the traveling salesman problem. It is split up in to two steps: the generalized insertion (GENI) and the unstringing and stringing (US) procedure. In the GENI algorithm, an initial tour is created by selecting an arbitrary subset of 3 nodes. Then, each new arbitrary node is added to the tour by a least cost insertion considering one of two possible orientations, which are named as Type I insertions and Type II insertions. A p-neighbourhood is defined for every node as the p nodes on the tour closest to it. This p-neighbourhood defines the orientation of the nodes for both types of insertions. This is repeated for all nodes until the traveling salesman tour is complete. The amount of insertions are restricted by only considering the neighbourhood of the new node and this makes the method more efficient than considering all possible insertions. It is showed in the paper, that

for Euclidean problems, the size of the neighbourhood can cause a tour that crosses itself and eventually arrive at suboptimal tours as the best possible solution. By increasing p, the size of the neighbourhood, the complexity of the insertion increases, but the optimal tour can be reached.

The US algorithm is a post-optimization that consist of removing a node from a feasible tour and inserts it back. The removal of a node can again be done in two ways. These two possible removal types are the opposite of the insertion types in the GENI algorithm. By starting at an arbitrary node, this node is removed and reinserted according to the two types of unstringing and stringing. The lowest cost orientation is stored and in the next step, another node is tested. This process repeats until it results in a number of worse solutions in a row. This number is equal to the amount of nodes. Because now all nodes are part of the tour, a node that is to be removed is not necessarily located between two nodes belonging to its p-neighbourhood and therefore reinsertion in the same orientation may be forbidden. It was found that the solution quality of the GENIUS algorithm improves as p becomes larger, which is to be expected since more nodes are considered in the neighbourhood of the node. The algorithm has been tested against other alternative heuristic approaches for the TSP. It turned out that it is advantageous in terms of solution quality and computation times, especially as the number of nodes grow beyond 200. The presented routines can be extended to a number of VRPs and other types of minimum cost permutation problems, which has been done by [15] for a TSPTW.

[15] provides an approximate algorithm to handle TSPTWs with large time windows. It builds on the GENIUS procedure introduced by [14]. It is stated that it is focused on finding good quality solutions for problems with large time windows and as a post-optimization step in VRPTWs. The principles of GENIUS can be translated, but care must be taken since paths are directed because of the time windows. The insertion procedure works as follows. Both distance and time dimensions are considered in the definition of node neighbourhoods since only looking at distance does not work well with time windows. Instead of defining a p-neighbourhood, a p^+-neighbourhood and p^-neighbourhood are defined per node. The p^+-neighbourhood are a number of closest successors of a node on the tour in terms of distance and a number of closest successors on the tour in terms of the proximity of the time windows. The p^--neighbourhood is defined the opposite way. The origin depot is always part of the predecessors and the destination depot is always part of the successors. It also keeps track of the latest time at which it is feasible to leave a node and enter a node. The order in which nodes are inserted is not random and depends on a measure of difficulty of insertion, which can be window width, earliest arrival times or earliest window closing time. The first option turned out to be the most satisfactory. All nodes are then added to the routes. If a node cannot be added, a backtracking procedure is used in which each routed node is tried to be replaced by an unrouted node. The nodes that are removed and replaced are being tracked in a list and if some node reaches an upper limit of removals, the process stops with an infeasible solution. This method looks a lot like the tabu search method. A tabu search keeps a tabu list of forbidden moves and here no moves are tabu, but there is an upper limit on the amount of times a certain move can be executed.

The next phase is the post-optimization. This can be applied when the insertion procedure produces a feasible solution. It works the same way as the unstringing and stringing in [14], although now a lo- cal rearrangement is only considered of it preserves time window feasibility. This is repeated from the origin depot to the destination depot. If the solution has not improved after a full pass of all nodes, the post-optimization procedure stops. The results of this heuristic indicate that for problems with large time windows, this method consistently produces feasible solutions in shorter times than other exact approaches with which it is compared. The results are attributed to the GENIUS method, neighbourhood definitions and backtracking procedures when no feasible insertions are possible.

4.6. Adaptive large neighborhood search

A lot of research has been done on local search heuristics. The tabu search is a local improvement method where marginal changes to the solution are tested in each iteration. This means that only solutions in the close neighborhood are explored for improvements. A genetic search randomly explores different parts of the solution space, but since good initial solutions can be created, it is expected that a controlled search over a larger solution region is more beneficial for this research. Compared to the tabu search method, large neighbourhood search, as the name implies, changes large parts of the solution.

[10] presents an adaptive large neighbourhood search (ALNS) for the pollution-routing problem (PRP), which is an extension of the classical VRPTW. The problem is formulated as an integer programming problem which

minimizes the pollution levels and driver wages as a consequence of the routing schedule. The heuristic works in two stages. In the first stage, it involves solving a VRPTW using the ALNS heuristic to find optimal routes, and the second stage is optimizing the speed on those routes to minimize fuel consumption and driver wages. The removal and insertion involves removing not one, but several customers (nodes) and reinserting them. It is stated that the initial solution quality is not so important since the algorithm can easily recover from a poor initial solution. Twelve removal and five insertion operators are assigned a score and a probability of selection during the optimization. Three of the removal operators and one of the insertion operators is newly introduced in this paper. The score increases if the solution improves and decreases if the solution deteriorates. A higher score has a positive impact on the probability of it being applied in the next iteration. The removal starts with selecting an operator that removes some nodes and stores them in a removal list. This is done a number of iterations and a partially destroyed solution is left. Insertion operators are used to re- pair the partially destroyed solutions by inserting the nodes from the removal list. The nodes can only be inserted when feasibility with respect to capacity and time windows can be maintained. A simulated annealing framework was borrowed for the ALNS algorithm to assess the acceptance of a new solution. There is a chance of accepting a new solution based on a temperature variable that decreases gradually, making it more likely to accept new solutions over the course of more iterations. Once the routes have been found, the speed optimization algorithm then solves a non-linear objective where the speed on a route is optimized per vehicle.

Three groups of parameters were distinguished, which defined: (I) the selection criteria for the removal and insertion procedures, (II) the calibration of the simulated annealing framework, and (III) the workings of the removal and insertion operators, determining a.o. the removable nodes and candidate node weights. Also in this paper, results were generated for [Solomon, 1987] benchmark VRPTW instances, not testing the speed optimization algorithm. It turned out that the ALNS with the new operators performs very well, in particular also in the clustered C1 and C2 instances where the optimal solutions were found within several minutes op computation time. The speed optimization algorithm shows that for practical 50 to 75-node sized instances (comparable to KLM), the ALNS heuristic is able to improve the optimal solution compared to CPLEX, a commercial optimizer, by 2-6% while taking just a fraction of the time. In addition, the algorithm shows that it is highly effective in finding good-quality solutions in a short time on instances with up to 200 nodes as compared to CPLEX.

This ALNS heuristic is much more intricate than the formerly described tabu search algorithm. Instead of one operator used in [8], this paper uses 17 in total for removal and insertion. That means that programming and implementing the ALNS heuristic, also meaning finding the right parameter setting, will take significantly longer than implementing the tabu search. However, it shows similar results as the tabu search method but the computation time is much shorter, which is a positive aspect for operating it at KLM. Although it must be mentioned that available computing power plays an important role in the computation time.

5

Literature Study: Modeling under uncertainty

The data stored from hub operations at KLM has shown not to be entirely reliable. The time at which pallets can start to be build up, in other words the lower bound of the time window of a to be prepared flight, has not been tracked before. This means that an attempt at a novel analysis needs to be made in order to get input values based on real data. Even though many data points are being tracked, not all data points have been verified and the booked shipments and pallets are often not equal to actual amounts. Assumptions or forecasts need to be made in order to acquire useful data from the system to be used as input data into the model when applying the model to real data. The nature of the input data means that uncertainty can be present in capacity constraints.

The research done in [9] contains an example of dealing with uncertain parameters. The article presents the allocation of cargo to passenger flights as a multi-commodity ow problem with uncertainty in demand and capacity. Omitting the multistage scenario tree to determine capacity constraints, an easier to understand and implement approach is the determination of demand. The demand is assumed to be normally distributed and is converted into a deterministic chance constraint. The constraint will change depending on the required confidence level requested by management or attainable confidence level from the input data.

Next to chance constraints, [18] also describes that robust optimization can be applied in order to deal with uncertainties. The goal of robust optimization is to virtually guarantee a solution for all practically possible parameter values. For this a distinction is made between a soft constraint and a hard constraint. A hard constraint defines a limit that must be satisfied and a soft constraint can be violated a little bit without serious complications. To determine the allowed violation of soft constraints, a range of uncertainty is defined for an independent parameters surrounding its estimated value. By assigning the most conservative value to the parameter, either the minimum or the maximum depending on whether the constraint is in the $\leq$ or $\geq$ form, will guarantee a feasible solution regardless of the values taken on by these parameters. In other words, if the model has a feasible solution in the worst case, it will have a feasible solution in a slightly better case.

III

Supporting work

1

Appendix 1

In this part, some supporting figures will be presented which are not included in the scientific paper. It serves as an additional explanation of the process undertaken during the research or to clarify some concepts mentioned in the paper.

1.1. Workforce planning

In the business analysis it is mentioned that schedules are made a few hours before each shift. Available teams are assigned to buildup tasks. The tasks are, as shown in the paper, a consolidation of a set of ULDs from the booked shipments for each flight. But how are the teams determined?

The teams are formed as a result of a long planning process. It starts months in advance on the planning department. A forecast of the work for a certain shift is made by a rough estimation of the required number of employees. Note that this estimate does not look at booking data and number of ULDs so it is a different approach of estimating work than presented in the paper. From this estimate follows a preliminary roster based on the availability of personnel. Over time the estimates are revised and a definite roster is made up for regular KLM employees a couple weeks in advance. Because bookings can change up until the last hours before a shift, there is a need for additional manpower that can be called up on a late notice. Also, employees may not be able to show up because of illness. For these reasons there is a possibility to work with contractors, or flex employees in order to have a sufficient workforce at the start of the shift.

All available employees for a shift are put together in teams based on their skills, experience and their contract (regular/flex). These are made to the best of the abilities of the staff planner. All the aforementioned steps occur in a planning tool called Novulo, as is shown in 1.1. Finally, a couple hours before the start of the shift, the shift leader consolidates the team information with other inputs like attendance sheets, booking data in the operational database and their own experience.

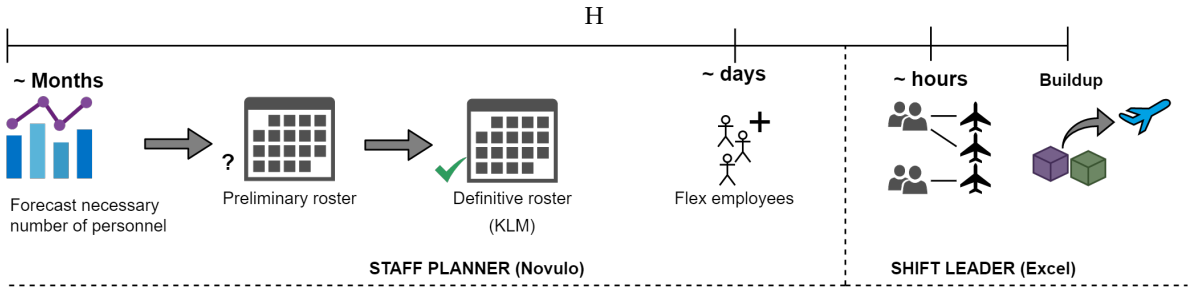


Figure 1.1: The planning of personnel starts far ahead of the buildup shift

1.2. Data entities

The paper mentions a database in the form of a collection of tables with the name *HubTrack*. It is a temporary database that consolidates multiple data points in order to track the actions of each shipment in the Hub from the past 8 weeks. Shipments are tracked by AWB number, i.e. a contract between carrier and shipper. The Hubtrack data can be viewed as a consolidation of information from multiple entities as shown in Figure 1.3.

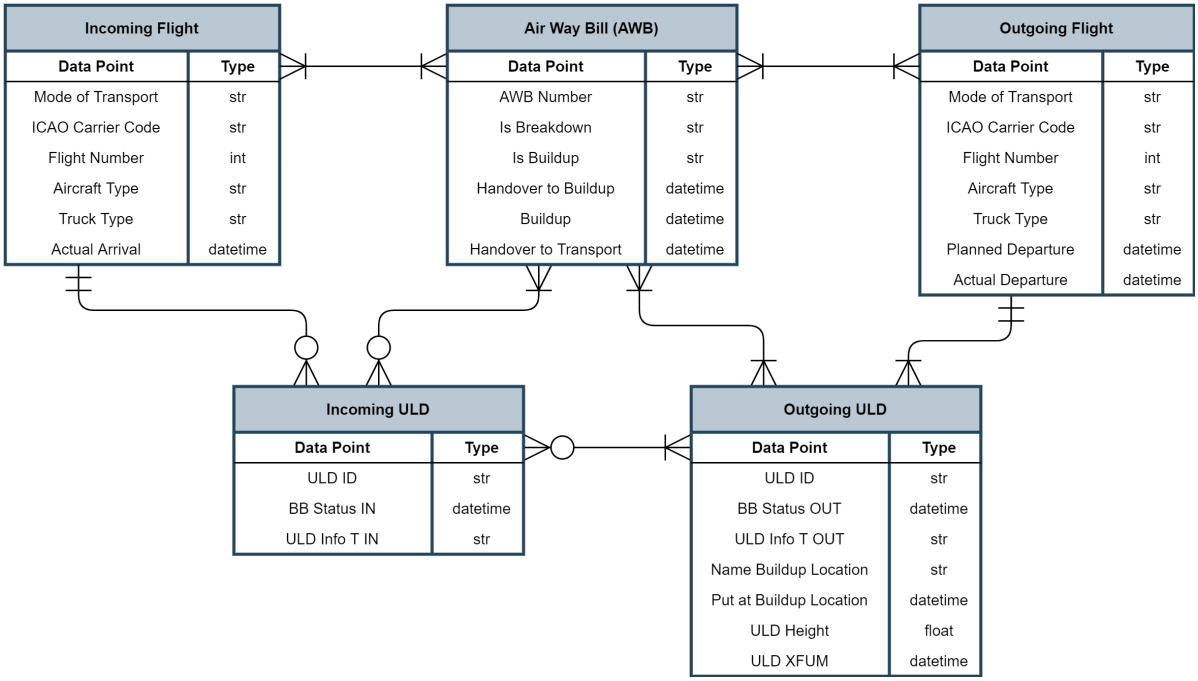


Figure 1.2: Entity Relations Diagram of HubTrack

The diagram shows that AWBs (shipments) can be on zero or multiple incoming ULDs, while always leaving on at least one outgoing ULD. An outgoing ULD could therefore not have any incoming ULD associated with it, meaning it arrived as loose freight. Incoming flights can also be truck rides as shown by the data point *Mode of Transport*. The BB status is given twice. For outgoing ULDs this is done when a ULD is only partly build and then stored for a later shift.

Some data points are more useful than others and there were many missing entries in the data, which hindered the exact ULD classification and release times. Often lines had to be dropped and therefore many data could not be used. Wrong entries occurred too, where for example a ULD was tagged and moved to transport before the last shipment was built onto it. These entries could not be included in the data.

1.3. ULD categorization

During the exploratory data analysis, a classification scheme has been developed in order to separate the different categories of ULDs from the data. The data entities and their data points like timestamps and information tags are used throughout this scheme. It is shown in 1.3 and starts at the top left with an uncategorized outgoing ULD.

For each outgoing ULD in the data, all rows are extracted containing corresponding AWBs and incoming and outgoing modes of transport. The data points are compared in a decision tree. Once enough decisions can be made in favor of a certain ULD category, it assigned that category.

An LF ULD is an empty ULD which requires 100% of work, and BB-ULDs are ULDs that are partly built and need additional shipments (BB-build) or checking (BB-check). T-ULDs are through ULDs and if they are correctly delivered they can immediately go through to the transportation department to be put on a flight. In case something is not in order, it needs to be checked.

As can be seen from the figure, T-ULDs that need checking are converted to the same category as BB-ULDs for checking. There is also only one combination of outcomes that would specify a T-ULD, meaning no work for the buildup phase.

Data from before, during and after buildup had to be used in order to properly categorize the ULDs. This can be seen as a flaw in this approach since it is desired to know this beforehand. This is discussed in the conclusion of the paper.

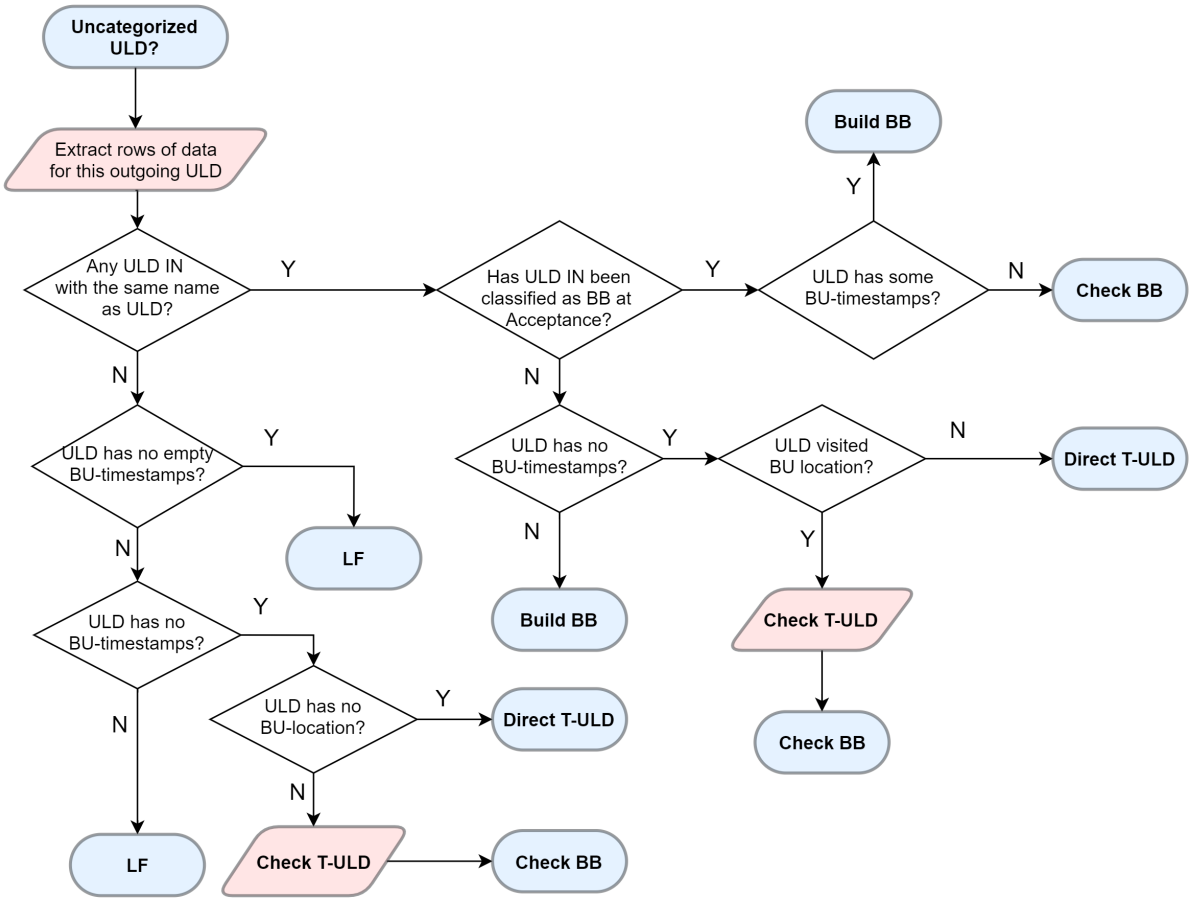


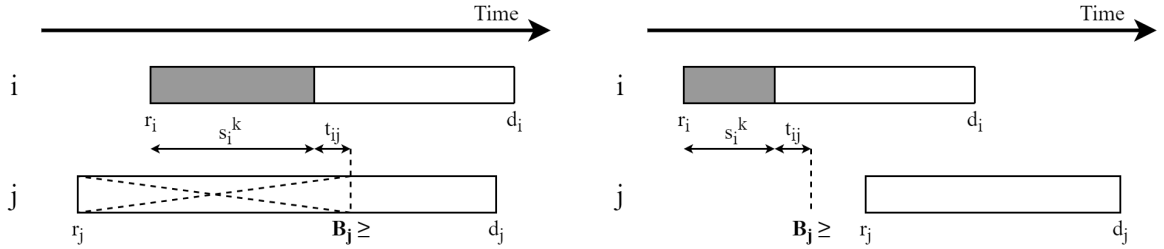
Figure 1.3: The ULD categorization scheme used during the exploratory data analysis

1.4. Valid inequalities in the model formulation

The model formulation makes use of so called valid inequalities. These are additional constraints that do not eliminate feasible solutions. It is used to strengthen the lower bounds of the time variables (B_i) with knowledge about the order of assigned tasks from the X_{ij}^k-variables. This is illustrated in 1.4. The equation is repeated here for clarification:

$$B_j \geq r_i + \sum_{i \neq j}^{N,N_B} \sum_{k}^{K} \max\{0, r_i + s_i^k + t_{ij} - r_j\} X_{ij}^k \qquad \forall j \in N, k \in K$$

In 1.4(a) the situation is shown when $r_j \leq r_i + s_i^k + t_{ij}$. The lower bound of B_j increases since task j cannot start at r_j. In 1.4(b) the situation is shown when $r_j > r_i + s_i^k + t_{ij}$. In this case, the *'max()'*-function prevents a miscalculated definition of the lower bound. The lower bound of B_j remains the same ($B_j \geq r_j$) since task j cannot start at the B_j when it is before r_j.



(a) An increase of the lower bound for the timing variables from the valid inequality.

(b) The purpose of the 'max()'-function in the valid inequality.

Figure 1.4: Illustration of the strengthening the lower bounds of B_i-variables with valid inequalities

Bibliography

[1] J. Christopher Beck, Patrick Prosser, and Evgeny Selensky. Vehicle Routing and Job Shop Scheduling: What's the Difference? *ICAPS*, pages 267–276, 2003.

[2] Walter J. Bell, Louis M. Dalberto, Marshall L. Fisher, Arnold J. Greenfield, R. Jaikumar, Pradeep Kedia, Robert G. Mack, and Paul J. Prutzman. Improving the Distribution of Industrial Gases With an On-line Computerized Routing and Scheduling Optimizer. *INFORMS Journal on Applied Analytics*, 13(6):4–23, 1983. ISSN 0092-2102. doi: 10.1287/inte.13.6.4.

[3] Adrian Brezulianu, Lucian Fira, and Monica Fira. A genetic algorithm approach for scheduling of resources in well-services companies. *International Journal of Advanced Research in Artificial Intelligence (IJARAI)*, 1(5):1–6, 2012. ISSN 1098-6596. doi: 10.1017/CBO9781107415324.004.

[4] Gaétan Caron, Pierri Hansen, and Brigitte Jaumard. The Assignment Problem with Seniority and Job Priority Constraints. *Operations Research*, 47(3):449–453, 1999. ISSN 19450699. doi: 10.1063/1.3061193.

[5] Dirk G. Cattrysse and Luk N. van Wassenhove. A Survey of Algorithms for the Generalized Assignment Problem. Technical report, Erasmus University Rotterdam, 1990.

[6] Gerard J. Chang and Pei-Hsin Ho. The β-assignment problems. *European Journal of Operational Research*, 104(3):593–600, 1998.

[7] Jean François Cordeau. A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54(3):573–586, 2006. ISSN 0030364X. doi: 10.1287/opre.1060.0283.

[8] J.F. Cordeau, G. Laporte, and A. Mercier. A unified tabu search algorithm for vehicle routing problems with soft time windows. *Journal of the Operational Research Society*, 52(8):928–936, 2001. ISSN 14769360. doi: 10.1057/palgrave.jors.2602371.

[9] Felipe Delgado, Ricardo Trincado, and Bernardo K. Pagnoncelli. A multistage stochastic programming model for the network air cargo allocation under capacity uncertainty. *Transportation Research Part E: Logistics and Transportation Review*, 131:292–307, 2019. ISSN 13665545. doi: 10.1016/j.tre.2019.09.011. URL https://doi.org/10.1016/j.tre.2019.09.011.

[10] Emrah Demir, Tolga Bekta, and Gilbert Laporte. An adaptive large neighborhood search heuristic for the Pollution-Routing Problem. *European Journal of Operational Research*, 223(2):346–359, 2012. ISSN 03772217. doi: 10.1016/j.ejor.2012.06.044.

[11] Martin Desrochers and Gilbert Laporte. Improvements and extensions to the miller-tucker-zemlin subtour elimination constraints. *Operations Research Letters*, 10(1):27 – 36, 1991. ISSN 0167-6377. doi: https://doi.org/10.1016/0167-6377(91)90083-2. URL http://www.sciencedirect.com/science/article/pii/0167637791900832.

[12] C. W. Duin and A. Volgenant. Minimum deviation and balanced optimization: A unified approach. *Operations Research Letters*, 10(1):43–48, 1991. ISSN 01676377. doi: 10.1016/0167-6377(91)90085-4.

[13] S. Geetha and K. P.K. Nair. A variation of the assignment problem. *European Journal of Operational Research*, 68(3):422–426, 1993. ISSN 03772217. doi: 10.1016/0377-2217(93)90198-V.

[14] Michel Gendreau, Alain Hertz, and Gilbert Laporte. New Insertion and Postoptimization Procedures for the Traveling Salesman Problem. *Operations Research*, 40(6):1086–1094, 1992. ISSN 0030-364X. doi: 10.1287/opre.40.6.1086.

[15] Michel Gendreau, Alain Hertz, Gilbert Laporte, and Mihnea Stan. A Generalized Insertion Heuristic for the Traveling Salesman Problem with Time Windows. *Operations Research*, 46(3):330–335, 1998.

[16] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H.G.Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326, 1979. ISSN 01675060. doi: 10.1016/S0167-5060(08)70356-X.

[17] Gurobi. *Mixed-Integer Programming (MIP) A Primer on the Basics*, accessed July 2, 2020. https://www.gurobi.com/resource/mip-basics.

[18] Frederick Hillier and Gerald Lieberman. *Introduction to operations research*. McGraw Hill, 01 2015. ISBN 9780073523453.

[19] Adam Janiak and Mikhail Y. Kovalyov. Single machine scheduling subject to deadlines and resource dependent processing times. *European Journal of Operational Research*, 94(2):284–291, 1996. ISSN 03772217. doi: 10.1016/0377-2217(96)00129-4.

[20] Harrold W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

[21] Gilbert Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(2):231–247, 1992. ISSN 03772217. doi: 10.1016/0377-2217(92)90138-Y.

[22] Ameer Mulla, Gurulingesh Raravi, R. P. Jagadeesh Chandra Bose, Thangaraj Rajasubramaniam, and Koustuv Dasgupta. Efficient Task allocation in Services Delivery Organizations. *Proceedings - 2016 IEEE International Conference on Services Computing (SCC)*, pages 555–562, 2016. doi: 10.1109/SCC.2016.78.

[23] E. O. Oyetunji. Some Common Performance Measures in Scheduling Problems: Review Article. *Research Journal of Applied Sciences, Engineering and Technology*, 1(2):6–9, 2009. ISSN 20407459.

[24] David W. Pentico. Assignment Problems: A golden anniversary survey. *European Journal of Operational Research*, 176(2):774–793, 2007. ISSN 03772217. doi: 10.1016/j.ejor.2005.09.014.

[25] A. Ravindran and V. Ramaswami. On the Bottleneck Assignment Problem. *Journal of Optimization Theory and Applications*, 21(4):451–458, 1977. ISSN 00223239. doi: 10.1007/BF00933089.

[26] Antonino Scarelli and Subhash C. Narula. A Multicriteria Assignment Problem. *Journal of Multi-Criteria Decision Analysis*, 11(2):65–74, 2002.

[27] Minxin Shen, Gwo-Hshiung Tzeng, and Duen-Ren Liu. Multi-Criteria Task Assignment in Workflow Management Systems. *Proceedings of the 36th Annual Hawaii International Conference on System Sciences, 2003*, pages 9–pp, 2003. doi: 10.1109/HICSS.2003.1174458.

[28] Marius M. Solomon. Algorithms for the Vehicle Routing and Scheduling Problems With Time Window Constraints. *Operations Research*, 35(2):254–265, 1987. ISSN 0030364X. doi: 10.1287/opre.35.2.254.

[29] A. Volgenant. Solving some lexicographic multi-objective combinatorial problems. *European Journal of Operational Research*, 139(3):578–584, 2002. ISSN 03772217. doi: 10.1016/S0377-2217(01)00214-4.