PALLET: improving efficiency in the transport sector with a multi-agent approach

Floor Eigenhuis

MASTER'S THESIS ARTIFICIAL INTELLIGENCE

June 12, 2018

Daily supervisor: L. van der Vliet

Project supervisor: dr. T. B. Klos
Second examiner: prof. dr. mr. H. Prakken

Utrecht University

Abstract

In the transport sector, pallets are often not used efficiently. Especially for large companies, it is difficult to keep track of pallets, resulting in a pallet surplus at many locations where pallets are shipped to and from. Ahrma, a Dutch start-up, has built an infrastructure to track pallets. Based on this infrastructure, a Multi-Agent System named Pallet has been designed for improving efficiency in pallet allocation. Every location is assigned its own agent and transports between locations that do not carry a full truckload are assigned empty pallets to transport them from locations with a pallet surplus to locations with a potential pallet deficit. Through simulations, the efficiency improvement is measured. Efficiency is measured using three KPIs. Pallet improves efficiency on all three KPIs, but the outcomes generated by Pallet are not guaranteed to be optimal and the outcomes produced are hard to predict. More research is needed to conclude if Pallet will improve efficiency in practice.

Acknowledgements

I would first like to thank dr. Tomas Klos, for providing me with valuable input on my work and for the many discussions we had on how to proceed next. He was always available for questions and urged me to think about concepts I would not have though of myself. I would also like to thank prof. dr. mr. Henry Prakken for his feedback.

Next, I would like to thank everyone at DearNova and specifically Lars van der Vliet for their help and support. Every time I had a question about a subject, there was always a colleague who was willing to help me and I really felt part of the company during my internship.

Lastly, I would like to thank my family, boyfriend and cats for their support and distraction. My family because even though they probably didn't (and still don't) understand what I have been working on for the past 7 months, they put up with me when I was most stressed. A big thanks to my boyfriend who wanted to understand what I was working on, but also probably didn't. Nonetheless, he proofread some of my work for which I am very grateful. A last thank you goes out to my cats, who reminded me that even (or maybe especially) in the most stressful of times, cuddles are an absolute necessity.

Floor Eigenhuis, 12 June 2018

Contents

List of Figures v

List of Tables v

1 Introduction 1
- 1.1 Problem statement . . . 1
- 1.2 Pallet tracking . . . 3
- 1.3 Research question . . . 3

2 Approach 3

3 Multi-Agent Systems 5
- 3.1 Multi-Agent Planning Systems in the transport sector . . . 6
- 3.2 Frameworks for building Multi-Agent Systems . . . 7
  - 3.2.1 FIPA . . . 8
  - 3.2.2 The JADE framework . . . 8

4 Design of PALLET 10
- 4.1 Detailed interaction between the agents . . . 11
- 4.2 The planning algorithm . . . 13
  - 4.2.1 Existing planning problems . . . 14
  - 4.2.2 Iterative deepening-like planning . . . 15

5 Generating the Transport Schedule 20
- 5.1 Assumptions . . . 23
- 5.2 Generating the transport schedule . . . 24

6 Determining the efficiency improvement 26
- 6.1 Baseline . . . 27
- 6.2 Simulation . . . 27
  - 6.2.1 Technical details . . . 28

7 Results 28
- 7.1 KPI 1 . . . 29
- 7.2 KPI 2 . . . 31
- 7.3 KPI 3 . . . 32

8 Discussion 32
- 8.1 Using PALLET in practice . . . 34
- 8.2 Quality of solution . . . 34
- 8.3 Cost savings . . . 34

9 Conclusion 35
- 9.1 Future research and improvements . . . 35

Appendices **38**

A Pseudocode planning algorithm **38**

B Pseudocode transport schedule generator **41**

C Proportion of full trucks **41**

D Predictability of outcomes produced by PALLET **42**

List of Figures

1 JADE UML diagram 9
2 Sequence diagram 12
3 Visualization of the transport schedule of table 1 15
4 Total search tree that can be generated in this example. 16
5 Steps of the planning algorithm 18
6 Visualization of the transport data 21
7 Degree distribution in the transport data 22
8 Number of pallets per truck ride 26
9 Number of pallets added to locations 29
10 Pallet difference over time 30
11 Average number of pallets per location per day, ordered according to the ordering in figure 9 31
12 Average number of pallets per location per day 32
13 Proportion of full trucks 42

List of Tables

1 Transport schedule used in example 1 15
2 Transport schedule used in example 2 17
3 Free pallets at relevant locations used in example 2 17
4 Added transport in example 3 20
5 Format transport data 21
6 Number of pallets added 29
7 Transports with a full truckload 32

1 Introduction

As we arrive into a new and exciting age of technological advancements and Artificial Intelligence, it is hard to think that some things that seem simple now were once revolutionary in its industry. One such revolutionary invention at its time was the pallet. Now simply known as simple pieces of wood or plastic nailed together to transport goods and often discarded quickly, it revolutionised the transport industry in the 1920's (Barell n.d.). With the use of pallets and the mechanical forklift, goods could be transported faster than ever before. Since the 1920's, there has been little innovation in pallets, and this poses some distinct problems for the transport and logistics industry now.

1.1 Problem statement

In the transportation in logistics domain, there is still a lot to be gained from better planning. It often happens that trucks transporting goods are driving (part of) their trip (partially) empty; this is not only a waste from financial point of view, but also releases more greenhouse gases than necessary. Even though efficiency in the transportation sector has increased over the past years (McKinnon and Ge 2006), there is still room for further improvement. For example, the European Commission found that on any day nearly 25% of trucks drive empty in Europe, either in between loads or on the way home (European Commission 2014). There has been no research on partially filled trucks; however, the European Commission states that it is to be expected that these less-than-truckload practices are also common (European Commission 2014). There are, of course, also other means of transportation than through road traffic. Other modes of transportation include rail, air, water and pipeline (Davidsson et al. 2005). However, this thesis will only address road transport.

Goods in road transport are transported on pallets. There are several types of pallets, the standard in Europe being the Euro pallet. The Euro pallets are specified and standardised by the European Pallet Association, commonly known as EPAL (EPAL 2017a). These pallets are made of wood and measure 800 mm by 1200 mm. There are approximately 450 to 500 million of these pallets in circulation in Europe (EPAL 2017a). Traditionally, there are several ways in which pallets are managed (Harris and Worrell 2008; Ren et al. 2017):

- Transfer of ownership. In this pallet management approach, pallets are simply sold to the company that wants goods to be transported. If the pallet is reusable, the company will reuse it in its own facilities. If the pallet is not reusable, the pallet will be discarded.
- Pallet exchange. In this approach, every time goods are delivered on pallets, the receiving facility will give the delivering party the same number of empty pallets in return.
- Pallet pools. There are several types of pallet pools, most notably private pools and third-party owned pools. In the private pool, the pallet manufacturer owns

the pallets and manages the pallet pool; if pallets are lost or broken, the manufacturer must replace them. In a third-party owned pool, the pallets are owned by a third party that manages the pool.

The standardised Euro pallets are part of an open pallet pool (EPAL 2017b) as EPAL calls it. However, by the above definitions, the Euro pallets are part of a pallet exchange– intuitively, a pallet for a pallet. Pallets are exchanged immediately. When a truck delivers goods on a pallet somewhere, the truck driver will get empty pallets in return. A Euro pallet currently costs around €10. There are also options for companies to rent Euro pallets. Plastic pallets with the same dimensions of wooden Euro pallets are also used in Europe, though not as commonly.

The design of pallets has always been simple. There have been few technological advancements in the design of pallets because technology has never been cheap enough to put it in a disposable pallet. Increased globalisation and trade since the 1920's has resulted in enormous amounts of pallets being used and large companies rent or own thousands if not millions of pallets. Often, companies lose track of their pallets and hire people to count the pallets stacked in their warehouses. Ideally, a pallet arriving in a warehouse would be emptied only if the goods are needed at that location. If not, the pallet is ideally stored with the goods stacked on them waiting for the transport to the next destination. This is not the case; pallets are often not stored full but empty. Goods shipped from manufacturing facilities to distribution centres may need to be reassembled on different pallets before continuing their way to their next location, hence a pallet deficit at some locations and pallet surplus at others arises.

During my thesis internship at DearNova I encountered a client of theirs, Ahrma, a Dutch start-up that has decided to solve this problem. Ahrma has produced smart pallets equipped with a transponder that broadcast a periodic signal. The signal is received by gateways that are placed in manufacturing facilities, distribution centres, or trucks, and passed on to a dashboard. The dashboard then has a complete overview of the number of pallets at each location. Ahrma pallets are made out of a wood and plastic hybrid, making them more durable than Euro pallets. Ahrma rents out these pallets to companies. Companies renting the pallets receive the advantage of having more insight in their supply chain and knowing where their pallets are at any time. Tracking pallets also has other advantages; because it is known where the pallets are, pallets can be used more efficiently, and fewer pallets need to be used to reach the same goal. That is also one of the claims that Ahrma makes, however, logistic systems are complex and involve many factors. Even though Ahrma offers a system to track pallets, finding a way to use pallets more efficiently is not trivial. Given the fact that many locations have a pallet surplus and trucks often do not drive with a full truckload, this thesis aims to use Ahrma's pallet tracking infrastructure to fill partially empty trucks with empty pallets, and get these empty pallets to locations that need pallets in the near future. The ultimate goal is to improve efficiency in pallet allocation.

1.2 Pallet tracking

While the individual technologies that Ahrma proposes are not new, these technologies have never been used in tracking pallets. Because of the low costs of wooden pallets, it has never before been profitable to apply these technologies to pallets. Only now the pallets are more durable and the technology used has become considerably cheaper than ever before is it worthwhile to track pallets this way. No literature about efficiently using pallets or improving the efficiency of the supply chain by pallet management has been found. The only literature that resembles tracking pallets is a study performed for the United States military (Harris and Worrell 2008). The authors research the use of tracking pallets in the US military by the use of RFID tags (Harris and Worrell 2008). These RFID tags prevent the pallets from getting lost, as it can be seen where the pallets are at a certain moment. However, the study shows tracking pallets *can* be useful in the future, but there is no indication up to now that these traceable pallets are actually used. Next to the active RFID tags, there are also companies that produce plastic pallets with passive RFID tags and bar codes, which can be scanned to reveal supply chain information (*iGPS* n.d.). However, studies about using this information for pallet allocation or studies about pallet allocation in general are scarce, especially those who incorporate uncertainty (such as delays in transport because of traffic jams, pallets breaking, or sudden changes in transport schedules) (Ren et al. 2017).

1.3 Research question

The main research question is:

> *How can the efficiency in pallet allocation be improved with the use of pallet tracking?*

To answer this research question, several sub questions need to be answered first.

1. What is an appropriate approach for improving pallet allocation?
2. What are appropriate frameworks/programming languages to use for the approach chosen above?
3. Are there existing algorithms that will aid with the approach chosen above?
4. Does the chosen approach improve efficiency in pallet allocation?
5. Is it a realistic assumption that the resulting system can be used in the industry?

2 Approach

The start of the research lies in answering sub question 1. There are a couple of things to consider. First of all, it is important to note that optimisation of pallet allocation can be regarded as a *planning problem*. The goal is to utilise empty truck space and fill partially empty trucks with empty pallets to transport them to locations

where pallets are needed in the future, resulting in more efficient pallet allocation. This problem might include millions of pallets and thousands of locations, and thus a global and optimal solution can become hard to find in a short amount of time. Local, and potentially sub-optimal, solutions are preferred or a heuristic should be used to appropriate a global solution. Secondly, because the transportation domain is dynamic, and changes may occur every day, the resulting system must be flexible enough to deal with this. There are, globally, two ways in which a planning problem can be approached: with a *centralised* or *decentralised* approach (Bonisoli 2013). A centralised approach includes a central planner which oversees the whole process and tries to find the global optimum solution, or, in case computing the global optimum is not feasible, uses a heuristic to approximate the global optimum. Decentralized systems consist of different components, each trying to work towards a solution. The different components in distributed systems all have limited knowledge. While distributed systems can work faster for complicated problems, they might not reach a global optimum, which the centralised solution might. In this particular case of optimisation of pallet allocation, there are several arguments as to why to favour a distributed over a centralised solution:

- Finding a global solution may not be feasible. With thousands of locations that need pallets and millions of pallets in the system, computing a global optimum may get very difficult, fast. With a distributed solution, the knowledge is distributed over the system, and instead of finding a global optimum, several local optima may be found.

- The domain is already inherently distributed, so a distributed system follows naturally: the aim is to divide pallets over a network consisting out of different locations. So, the system is already distributed: all the locations indicate a separate facility where pallets are needed. It is only natural to model these locations as parts of the system itself. Moreover, the data in the system is inherently distributed: at the locations different data is available; information about incoming and outgoing transports and the number of pallet at the location.

- Distributed systems generally perform better than centralised systems when the domain is uncertain or complex (Wooldridge 2009).

A distributed system is preferred over a centralised system. There are different kinds of distributed systems. A subclass of distributed systems is the field of *Multi-Agent systems* (Wooldridge 2009). In Multi-Agent Systems (MAS), software agents are distributed throughout a system and have only access to local information. The agents then try to optimise their objectives by using their information and communicating with other agents (Máhr et al. 2010). MAS gained interest from the Artificial Intelligence community over the past decades because of its ability to model complex situations, especially when the agents in question can exhibit intelligent behaviour. MAS, in turn, has several advantages over 'normal' distributed systems (Wooldridge 2009).

- While in distributed systems (that are not MAS) the different components of a system can work independently of each other, they are not *autonomous* like the agents in MAS are. In a MAS, the agents are assumed to be autonomous in such a

way that they can make their own decisions to reach their own independent goals. Also, agents communicate with each other and synchronise actions at run-time instead of it being hard-wired as is the case with traditional distributed systems (Wooldridge 2009).

- The agents in a MAS all strive to reach their own goals, whereas in a distributed system all the different components work together to reach one common goal.

A MAS is appropriate for the problems posed in this thesis because the agents should, in a way, be self-centred. Even though they can be fairly cooperative (because they all have a goal of improving efficiency in pallet allocation), it is also important that the agents remain their autonomy. In this case, when using a MAS, different locations could be modelled as agents. Because the pallets will be distributed over a network possibly consisting out of different companies (with each their own location), these companies might want to keep their transport data as private as possible and only share the parts that directly relate to pallet allocation. In a 'normal' distributed system the different components have no option of not disclosing information to other parts of the system, while in MAS this is possible due to the autonomy of the agents. Thus, it was decided to make use of a MAS. The Multi-Agent System was named Pallet: Pallet Agents in Lean Logistics for Economical Transport, where lean logistics refers to a way of thinking to eliminate all wasteful activities from the supply chain (Jones et al. 1997). In the end, to verify if Pallet really improves efficiency, simulations will be run using a transport schedule from a large company as input.

3 Multi-Agent Systems

Multi-Agent Systems have been a research area since around 1980 and gained wide recognition and study in the following decades (Wooldridge 2009). As mentioned before, a Multi-Agent System is composed out of different entities, or agents, which can interact with each other to either reach their private goals or one common goal. Even though MAS have been studied rigorously in the past years, there is no universal definition of what an 'agent' is or entails. In the definition given by Wooldridge and Jennings (1995) (Wooldridge and Jennings 1995), an agent is a piece of software that displays the following properties: autonomy (able to have control over actions), social ability (able to communicate with other agents), reactivity (able to react to the environment) and pro-activeness (displaying goal-directed behaviour). On the other hand, Russell and Norvig (2011) (Russell and Norvig 2011) define an agent simply as a piece of software that can perceive and interact with its environment. A Multi-Agent System is a collection of agents in a system. The agents can either work together towards a common goal, which results in a cooperative MAS, or work selfishly toward their own goal, which results in a self-interested MAS (Jennings et al. 1998). Multi-Agent Systems are especially useful in applications where parts of the system only have access to incomplete information, multiple solutions are possible, or multiple entities that can make decisions are in charge of problem solving (Jennings et al. 1998).

Regarding problem solving, an important protocol for task allocation in MAS is the Contract Net protocol (Smith 1980). A contract net refers to a collection of nodes (or agents) in which each node can assume the role of a manager or a contractor. While the manager monitors how the task is executed, the contractor executes the task. The managers announce the task to be executed and the contractors bid on the task; the manager then chooses the appropriate contractor to execute the task. Roles to the nodes are assigned dynamically; a node might be both a contractor and a manager when there are multiple tasks to be executed in parallel (Smith 1980).

3.1 Multi-Agent Planning Systems in the transport sector

A classic problem in the transportation sector is the *vehicle routing problem* (VRP), which concerns finding minimal cost routes for trucks to deliver goods at certain locations (Dumas et al. 1991). A variant of this is the *pickup and delivery problem with time windows* (PDPTW), which similar to the VRP but then with pickup and delivery under time constraints (Dumas et al. 1991). Most of the Multi-Agent Planning Systems in the transportation sector are based on auctions (Bürckert et al. 2000; Máhr et al. 2010; Robu et al. 2011; Baykasoglu and Kaplanoglu 2011) and propose a solution to some variant of the VRP or PDPTW (Davidsson et al. 2005). Next, a couple of Multi-Agent Planning Systems applied to the transport sector will be discussed.

Among the first Multi-Agent Planning Systems applied to the transport sector was the MARS system (Fischer et al. 1995). The MARS system was the first to combine the Contract Net Protocol and an auction procedure to solve resource allocation problems. In a related paper, Bürckert et al. (2000) introduce a holonic Multi-Agent System (TeleTruck) for dynamic planning, fleet management and driver scheduling. A holonic agent is one that is composed of multiple agents that work together towards one common goal; the holonic agent looks like one agent to the outside environment. Agents in such a holon communicate through a special type of agent, called the head of the holon. An advantage of this approach is that because the agents have to commit themselves to join a holon, and might even have to hand over control, much closer collaboration is possible than with other forms of cooperation between agents. Agents can represent, among other things, trucks or trailers so that when a trailer is attached to a truck, they can intuitively form one holon.

Karageorgos et al. (2003) propose a Multi-Agent Planning System that combines information from different manufacturing sites and several agents to improve efficiency in a virtual enterprise. They use a nested form of the Contract Net protocol.

In research by Máhr et al. (2010), it was shown that agent-based approaches do especially well in domains with high uncertainty, such as the transportation sector. The study aimed to shorten the time a truck drives without a profit-generating load, to maximise profits for the logistics company (Máhr et al. 2010). The problem in this study can be described as a PDPTW. The authors show that the agent approach performed competitively to an online and central optimization, but the agent approach

also provided several distinct strengths in finding a solution (Máhr et al. 2010).

Baykasoglu and Kaplanoglu (2011) propose a Multi-Agent Based Load Consolidation System (MABLCS). Load consolidation combines different items, often produced at different times at different locations, into a single vehicle load to improve efficiency and avoid less-than-truckload (LTL) transportation. The resulting support system is meant for third-party logistics companies. Consolidation decisions for LTL offers are made by different types of software agents: here, the software agents represent orders (order agents) and system resources (truck agents) while the actual decisions are made by the support system itself.

Robu et al. (2011) propose a multi-agent platform for the allocation of loads in transportation. Data used to test this platform was from a real logistics company, and because the platform allowed for human schedulers to intervene with the decisions the agents made, it proved to be useful in real day-to-day operations, which is different from the other approaches discussed here. It proved to be successful in the sense that the human planners at the logistics company were eventually convinced of the system's usefulness.

A study by Chow et al. (2013) proposed an air freight planning system. Instead of making an agent out of the shipments or airplanes, they have multiple agents in this approach: a mobile agent which moves to different platforms an gathers information from different agent platforms. There is a client request agent, which receives the client-defined criteria, an airline agent which knows the available flights, a financial agent which calculates the cheapest path and a risk management agent which calculates the risk with a certain transport path by analysing weather and news websites (Chow et al. 2013). To find the best path for a certain freight, eventually the financial agent and the risk management combine their knowledge to find the best path.

In the analysis of the previous papers it stood out that only Robu et al. (2011) and Chow et al. (2007) proposed a system which was actually implemented and tested in a field study. Davidsson et al. (2005) also notes that little of the agent-based transport systems propose fielded experiments. This indicates that implementations of Multi-Agent Transport Systems in practice are rare.

3.2 Frameworks for building Multi-Agent Systems

A study by Kravari and Bassiliades (2015) provided a comprehensive survey of the existing Multi-Agent frameworks. There are a few requirements for a MAS framework used for this particular purpose: first of all, the framework is preferably written in the Java programming language. This is because the existing framework for the pallets is also written in Java and if the MAS framework is written in Java, this will make it easier to connect the MAS to the existing program. Secondly, the MAS framework must be open-source. Third of all, the MAS framework must still be up-to-date and have an active user community if any questions might arise, and lastly, the framework

must be scalable and be able to have thousands of agents running in it.

Several agent frameworks were considered, such as Jason, JADE, JAS and JACK. Of all known agent frameworks, JADE (Java Agents Development Environment) is the most widely used (Kravari and Bassiliades 2015) and was of all the candidates deemed most appropriate for this purpose. JADE poses several distinct advantages. One advantage is that it uses the agent abstraction over Java, therefore being fully compatible with other programs written in Java (Bellifemine et al. 2007). Another advantage of JADE is that it can be distributed across different machines, who do not even have to have the same operating system and can also be used on mobile devices (*JADE* n.d.). Also, it is fully FIPA compliant (see section 3.2.1), making interaction with other FIPA compliant platforms easy (Bellifemine et al. 2007). Chmiel et al. (2005) have tested the scalability of JADE. The authors conclude that 'JADE is a very efficient environment limited only by the standard limitations of Java programming language' (Chmiel et al. 2005). The authors show that JADE has linear scalability and that it can easily manage hundreds of messages and hundreds of thousands of messages. Since this research was conducted 15 years ago (in 2003) and since then not only computers have become more powerful but also the Java programming language and JADE have been improved, it is to be expected that the results described by Chmiel et al. (2005) can be easily surpassed.

3.2.1 FIPA

FIPA (Foundation for Intelligent, Physical Agents) (*FIPA* n.d.) is a non-profit association which aimed to develop standards for agent technology. Founded in 1996, it was dissolved in 2004 but in 2005 FIPA was incorporated in a standards committee of the IEEE (Institute of Electrical and Electronics Engineers) computer society (Bellifemine et al. 2007). FIPA is now named FIPA-IEEE, and continues to set standards in the areas of human-agent communication and mobile agents (Bellifemine et al. 2007). Although the standards in FIPA are detailed, they only outline the outer workings of an agent system. The way the system is implemented is left to the programmer. FIPA describes standards in the areas of agent communication and agent management and also has developed the widely used FIPA Agent Communication Language (FIPA-ACL) (Bellifemine et al. 2007). For a complete overview of the FIPA specifications, refer to *FIPA* (n.d.).

3.2.2 The JADE framework

As mentioned above, JADE is a framework for building Multi-Agent Systems. Figure 1 shows the main architectural components of JADE.
In JADE, agents live in a container which is part of a platform. A platform can have multiple containers. Every platform has a special container called the *main container* which is the first one to be launched, and all additional containers have to register to the main container. Containers (and consequently, platforms too) can be distributed over different hosts. The main container has amongst its responsibilities hosting the Agent Management System (AMS) and the Directory Facilitator (DF) (Bellifemine

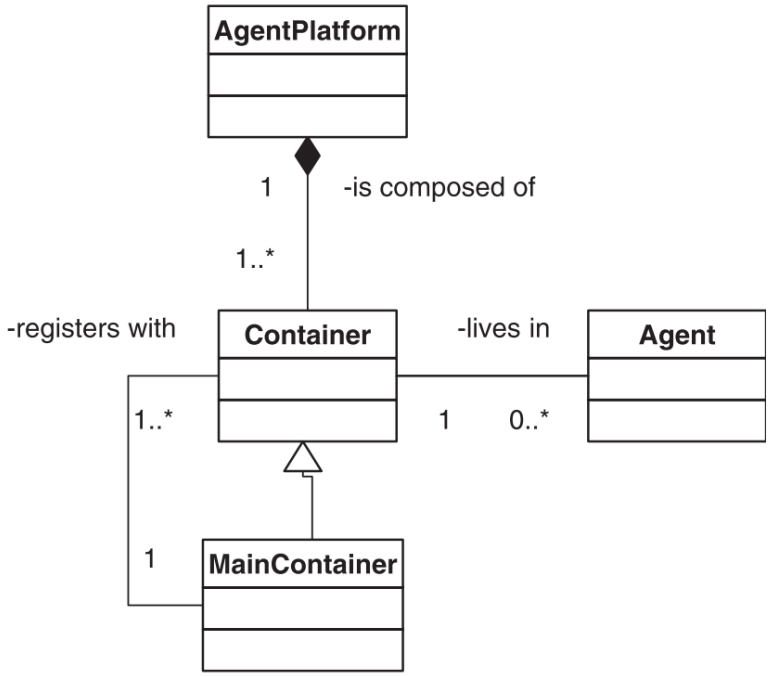


Figure 1: UML diagram displaying the main architectural elements of JADE. Picture copied from Bellifemine et al. (2007), pg. 33.

et al. 2007). The AMS contains a list of all the agents in the platform. It manages operations on the agent platform and is responsible for the creation and deletion of agents. The DF lists all the services the agents have to offer, and it the point of contact for any agent who wishes to deploy the services of another agent. Each agent runs in its own Java thread. Every JADE agent type has its own class in Java. Agent's behaviours are programmed as inner classes. There are different types of behaviours, such as `OneShotBehaviour` (which executes once and then stops), `TickerBehaviour` (which executes every tick) and `CyclicBehaviour` (which executes continuously). Agents *can* execute several behaviours concurrently, however, the scheduling of behaviours is not pre-emptive but cooperative, meaning that when a behaviour is performed, it takes over the agent's thread until the behaviour's `action()` method returns (Bellifemine et al. 2007). Behaviours can be added to the agent's behaviour queue in the agent's `setup()` method or from other methods. The agent does not start new behaviours directly but instead puts them in a queue and executes them one after another. The agents communicate with each other through sending messages. When an agent receives a message from another agent requesting some data or requesting to perform a behaviour, the receiving agent can always refuse the request, thereby keeping its autonomy. Messages are not immediately received; instead, a `CyclicBehaviour` is activated any time a message is received (when more messages are received they are put in the agent's message queue) and consequently only seen when the agent has come to that behaviour.

4 Design of Pallet

When designing a MAS, the first step is to identify the different roles that need to be fulfilled. Several components were identified: pallets, locations, and transports. Even though the pallets are the 'smart' components in Ahrma's system, the pallets are not modelled by assigning each pallet an agent. Instead, locations each get their agent, and pallets are modelled as an attribute of a Location Agent. This approach has been chosen because it does not matter which pallet is at a specific location, since they are all assumed to be the same in size and material. The locations, however, are a natural abstraction of agents: at the locations, there is information about the incoming and outgoing transports, the pallets at that location and how many pallets are needed to fulfil the outgoing transports in the future. Ideally, the Location Agents would themselves negotiate with other Location Agents to find out if they have pallets and assign empty pallets to transports. However, in practice this approach does not work as agents cannot read and respond to all the messages they receive fast enough, and the corresponding results are unpredictable or wrong (since an agent might work with information it got from another agent that might now be outdated). It was decided instead to move all these negotiations and calculations to another agent: the Calculate Pallet Agent. There is only one of each agent except for the Location Agents. Next to the Location Agents and Calculate Pallet Agent there are several other agents, each with their own distinct tasks, which mostly consist out of reading different databases and notifying other agents if any changes have occurred. It should be noted that Pallet is currently optimised for running simulations, and some agents might get different responsibilities or will be deleted if Pallet is used in real time. The agents and their detailed responsibilities are as follows:

- Agent Creator Agent
  This is an agent that is in charge of reading the database with all locations and creating a Location Agent for every location. This agent is needed because it needs to do a distinct task: reading the location database. Whereas the other agents can be easily created in a simple Main class, Location Agents need to be created with the location ID as their name.

- Central Transport Agent
  This agent has several duties. First of all, it checks the transport schedule regularly and informs locations that send or receive pallets of the planning. Secondly, it waits for messages from the Calculate Pallet Agent for empty pallets to be transported and updates the transport planning accordingly. It waits for a message from the Calculate Pallet Agent to change the date. After it has changed the date, it reads the transport planning for the new day.

- Location Agent
  Each location has its own agent. This is the most important type of agent in the system, and the only agent type that has multiple instances. The responsibilities of a Location Agent are to keep track of the upcoming incoming and outgoing transports, keep track of the number of pallets currently at the location and keep a

prediction for the number of pallets in the future based on the transport schedule it receives. Location Agents also communicate with the Calculate Pallet Agent to request pallets.

- Calculate Pallet Agent
  This agent has more of a symbolic role than a real instantiation of some physical entity like the above entities. The Calculate Pallet Agent receives messages from the Location Agents requesting pallets for a certain day and tries to find empty pallets and transports with empty places to transport the pallets to the location according to the algorithm described in section 4.2. When it has found those pallets, it sends messages to the involved Location Agents, the Central Transport Agent and the Gateway Agent to notify them that the transport has changed. When the day in the simulation is over, it sends a message to the Central Transport Agent to change the date.

- Gateway Agent
  This agent is only in the system for simulation purposes. It provides the information usually the gateway would supply; the number of pallets at a location. The Gateway Agent checks the transport schedule and updates the pallet database accordingly. The pallet database contains the number of pallets currently at each location. It also receives messages from the Calculate Pallet Agent with updates of to the transport schedule.

- Pallet Agent
  This agent is responsible for checking the pallet database and pass on the information to the Location Agents. Even though the Location Agents have their own representation of the number of pallets at their location, they do not have direct access to the database. The Pallet Agent does, and sends the information on to the Location Agents so they can check if their internal representation is right and that nothing unexpected has happened.

4.1 Detailed interaction between the agents

To recap, the goal of Pallet is to fill up trucks, which are not filled with loaded pallets, with empty pallets. These empty pallets are transported to locations where they are needed for further transports. The *planning problem* is then defined as matching empty pallets with available transports to get the pallets to locations where the pallets are needed.

For this to be feasible, several conditions have to be met: there has to be room in the truck, the location where the transport leaves from has to have empty pallets and the transport carrying the empty pallets needs to arrive at the location that needs the pallets before the pallets are needed. Figure 2 illustrates how these agents work together and what the sequences of their actions are. All the agents in Pallet live in one container, the main container, on one agent platform, on one machine. When

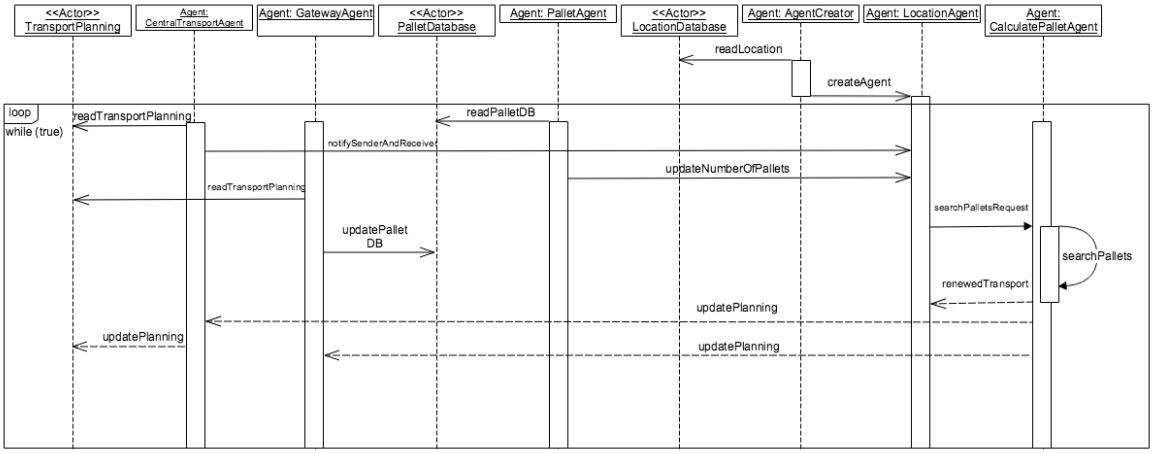


Figure 2: A UML sequence diagram of Pallet.

Pallet is started, all of the agents are initialised from the Main class except for the Location Agents. First, the Agent Creator Agent reads the database with all the location types and creates an agent for every location. Once the Agent Creator Agent is done reading the location database and creating the agents, it broadcasts a message to all the other agents to let them know that they can start their tasks.

The Central Transport Agent reads the transport planning for the day and then sends a message to the sender and receiver of the corresponding transport to notify them of the upcoming transport. The Pallet Agent reads the pallet database which registers the numbers of pallets on location and sends a message to each Location Agent to notify them of the number of pallets on location. Notice that only on the very first iteration this is 'news' for the Location Agent; since each Location Agent has an internal planning of the number of pallets on location based on the planning they received from the Central Transport Agent, after the first iteration this message from the Pallet Agent to the Location Agent is more of a check to see if the predictions were right.

After the Location Agents have processed the messages from the Central Transport Agent and the Pallet Agent, each Location Agent calculates if it has enough pallets to fulfil the future transports. Each pallet at a location is internally classified as a *reserved* or a *free* pallet. Pallets are classified as reserved when they need to be used for a transport that leaves within two days. These pallets cannot be taken by other Location Agents. Free pallets are pallets that are not meant to be transported in the next two days. Free pallets can be taken by other Location Agents.

When a Location Agent decides it does not have enough pallets to fulfil the coming transports, it first looks at its own transports to see if there are incoming transports that arrive before the pallets are needed and have room to carry empty pallets.

If there are no such transports, then no solution can be found and the next day, the Location Agent tries to find shipments again until the day the pallets are needed.

If there are such transports, the Location Agent sends a message to the Calculate Pallet Agent. The Calculate Pallet Agent then searches for new pallets according to the algorithm in section 4.2. If a total or partial (only part of the pallets have been found) solution has been found, the Calculate Pallet Agent sends a message to the involved Location Agents, the Central Transport Agent and the Gateway Agent notifying them of the updated transports. If no solution has been found, it sends a message to the Location Agent that originally sent the message and notifies it of the failure.

If still no pallets have been found the day before the pallets are needed, the pallets are added to the location and the number of pallets added is recorded (when used as in a simulation). When Pallet is used in practice and pallets cannot be added instantaneously, a message is sent to a human in charge notifying them of the upcoming pallet deficit at a location.

4.2 The planning algorithm

The following section assumes a basic understanding of planning problems and algorithms such as depth-first and iterative deepening search. For an overview of these, see Russell and Norvig (2011).

Regarding sub question 3, there are multiple approaches that are used in these kinds of problems. However, it should first be decided what kind of solutions are preferable over others. As mentioned before, a key point of a MAS is that the agents are autonomous. This means that agents cannot access private information of other agents directly. Instead, they have to request the information. The following requirements for solutions were set:

- It is preferred to use as few truck rides as possible.
- Locations that are closer (in terms of transports) are preferred above those that are further away. So, the solution that would need pallets from two locations that are each only a truck ride away is preferred above a solution where pallets need to be transported to an intermediate location to be transported to the goal location. This happens when there is no direct transport between a location that has free pallets and the goal location.

Next to the preferred solutions there are constraints to keep into consideration. The constraints that need to be considered are as follows:

- Transports that carry empty pallets need to arrive at the goal location one or two days before the day the empty pallets are needed. This is to ensure the pallets are there on time for the transport needing the pallets; also, this, in the real world, leaves enough time to prepare the pallet for transport (put the right goods on it). Also, the transport carrying empty pallets must leave after today's date.

- 33 loaded pallets fit in a truck. This is the number of Euro pallets that fit in a semi-trailer truck (*Types of trucks and trailers* n.d.). It is assumed there is only one type of pallet, and when a truck is not fully loaded, 15 empty pallets can fit in the place of 1 loaded pallet (since empty pallets can be stacked whereas loaded pallets cannot).
- As mentioned before, the only truck rides that are taken into consideration are the ones that already exist. The algorithm itself cannot create new truck rides.
- Only pallets that are labelled as *free* the day before the transport carrying the empty pallets leaves can be transported as empty pallets to other locations.

4.2.1 Existing planning problems

When considering all this, what remains is a planning problem *with constraints*. Where planning problems have long been studied in the fields of both Artificial Intelligence (AI) and more specifically Multi-Agent Systems, the usual approaches are not relevant to this particular planning problem. What is mostly meant by classical problems in AI and MAS is to plan actions to reach a *goal state* from a certain *start state* (Wooldridge 2009; Russell and Norvig 2011), or to start at a goal state and use backward reasoning to reach the start state.

The planning problem in this thesis has no actions to choose from. Instead, this planning problem has actions that are constrained and partially set. If transporting empty pallets from a sending location to a receiving location is seen as an action, then the date, the maximum number of pallets and the receiving and sending location are set. It is not possible to transport between these locations if there is no transport on that day, there is no room in the truck for empty pallets, or the sending location does not have free pallets. This makes this particular planning problem different from classical planning problems.

Although in section 3.1 several MAS in the transportation sector were discussed, these are also distinctly different from the problem posed here. Most of these concern some sort of variation of the Vehicle Routing Problem. The crucial parts of the Vehicle Routing Problem are that

1. there are goods that need to be transported from a certain location to a certain location;
2. the transports need to be planned to deliver the goods as efficiently as possible.

In this case, neither of those requirements is the case. In this planning problem, all the transports are known (albeit partly by different parts of the system) and the corresponding number of pallets need to be matched to available truck rides to deliver the pallets to locations where the pallets are needed. A different approach is warranted.

4.2.2 Iterative deepening-like planning

The planning problem can be visualised as a tree-like structure in which the locations are the nodes and the edges are transports that connect the nodes, with the location that needs pallets as the top node. For an example of this, see example 1.

Example 1: Visualization of locations and transports
Let table 1 contain the transport schedule.

Transport ID	Sender ID	Receiver ID	Loaded pallets	Empty pallets	Pick up by	Deliver by
0	A	F	32	0	08-08-2016	10-08-2016
1	A	D	30	0	09-08-2016	10-08-2016
2	B	D	31	0	09-08-2016	11-08-2016
3	C	E	28	0	10-08-2016	11-08-2016
4	D	F	32	0	12-08-2016	14-08-2016
5	E	F	32	0	12-08-2016	13-08-2016
6	F	G	33	0	15-08-2016	18-08-2016
7	D	H	32	0	15-08-2016	17-08-2016
8	B	E	32	0	16-08-2016	17-08-2016

Table 1: Transport schedule used in example 1.

If the locations are nodes and the transports are edges, the transport schedule can be represented as a graph. The resulting graph can be found in figure 3.

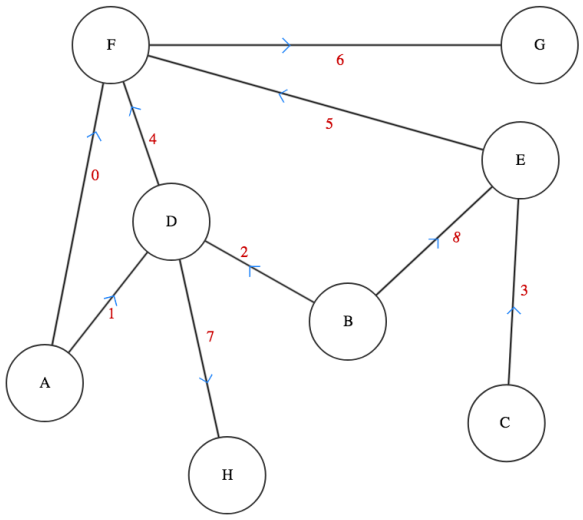


Figure 3: Graph of the transport schedule of table 1. The numbers next to the edges are the transport IDs. The arrow points in the direction of the transport.

Now assume Location Agent F notices it does not have enough pallets at its location to fulfil transport 6. Furthermore, let the variable *today* be 05-08-2016. Now the planning problem can be visualised as a search tree, with the location needing pallets (F) as the top node. Its children are the locations from which location F receives a transport one or two days before transport 6 leaves. The edges represent the

transports. The children of those children are generated as if that child was the top node. For an example of the search tree in this case, see figure 4.

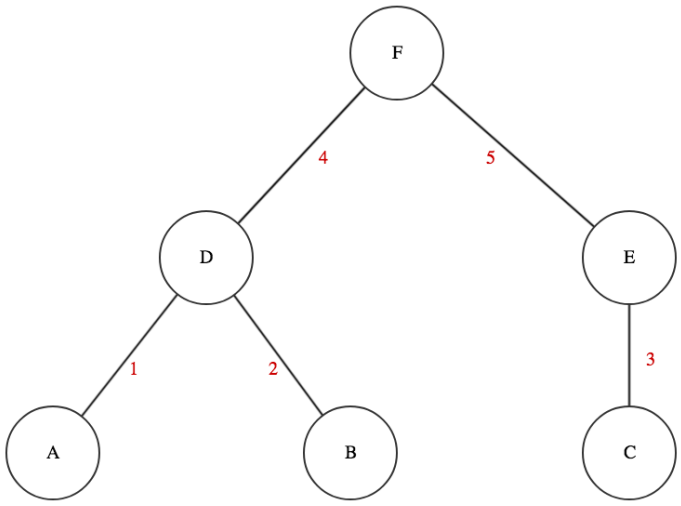


Figure 4: Total search tree that can be generated in this example.

In figure 4, the corresponding search tree is shown. The edges represent the transports to which empty pallets can be added- now it is the question of whether the nodes, the locations, have free pallets available. Note that not all the transports have been included in the search tree. Transport 1 was not included because it arrived at location F more than 2 days before transport 6 leaves. Transport 7 was not included because it is not relevant; location H is not connected to location F in the search graph. Transport 8 was not included because it arrives at location E later than transport 5, which connects location E to location F, leaves. Note that in the example given almost all of the transports had room left. If a transport already contains the maximum number of loaded pallets (33), that means that that transport cannot be part of the solution, so it is not included in the search tree.

The tree structure as described above is generated incrementally; only when a node is considered a 'goal location' are its children generated. This has the effect that the algorithm looks like *iterative deepening depth-first search.*

As mentioned in section 4, the planning algorithm starts with a goal location and all the appropriate transports to that goal location as input. The goal location is depicted as the top of the search tree, the nodes at the other end of these edges are depth 1. When not all the pallets that were needed were found at depth 1, the algorithm takes the first node at depth 1 as goal location and starts again. When no solutions have been found with that location as goal location, the second node at depth 1 is taken as goal location. When no solution has been found at depth 1, the algorithm takes the nodes at depth 2 as goal location and starts again. Nodes are not further explored when there are no more transports to that node or when truck rides further up in the tree are full.

For intuition, example 2 shows the detailed steps of the algorithm. The pseudocode of the algorithm can be found in appendix A.

Example 2: Steps of the planning algorithm

The relevant transports for this example, adapted from example 1, are depicted in table 2. Further on, let the variable *today* be 05-08-2016, so all the transports depicted below happen in the future.

Transport ID	Sender ID	Receiver ID	Loaded pallets	Empty pallets	Pick up by	Deliver by
1	A	D	30	0	09-08-2016	10-08-2016
2	B	D	31	0	09-08-2016	11-08-2016
3	C	E	28	0	10-08-2016	11-08-2016
4	D	F	32	0	12-08-2016	14-08-2016
5	E	F	32	0	12-08-2016	13-08-2016
6	F	G	33	0	15-08-2016	18-08-2016

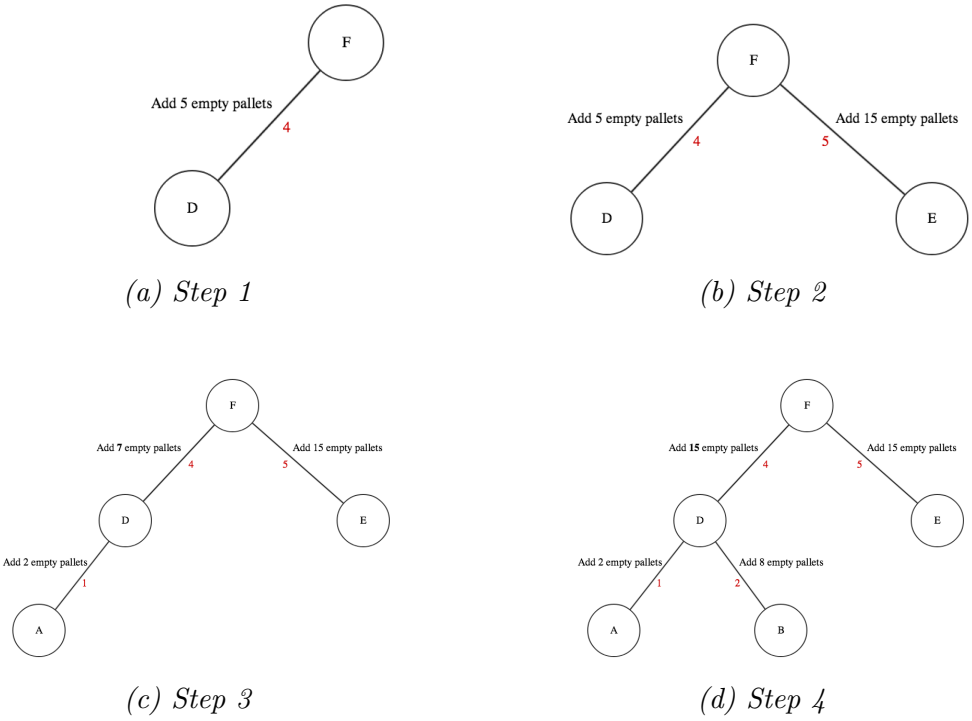
Table 2: Transport schedule used in example 2.

Assume that location F has just received information about transport 6, and it knows that the day before the shipment, August 14th, it only has 3 free pallets at location and thus does not have enough pallets to transport that day. Location F then sends a message to the Calculate Pallet Agent with the request for 30 pallets, to be arriving on either 14 August or 13 August, and its incoming shipments that still have room for empty pallets (transports 4 and 5). Table 3 denotes the free pallets at the locations talked about in this example.

Location ID	Date	Free pallets
A	08-08-2016	2
B	08-08-2016	15
D	11-08-2016	5
E	11-08-2016	30
F	14-08-2016	3

Table 3: Free pallets at relevant locations at the start of the search algorithm talked about in this example. Note that this table is only for illustrative purposes, no agent itself has a complete overview of the free pallets at other locations (they must always request it).

Figure 5 illustrates the different steps of the search algorithm in this scenario, where the nodes represent locations and the edges represent transports.



(a) Step 1 *(b) Step 2*

(c) Step 3 *(d) Step 4*

Figure 5: Visualization of the steps of the planning algorithm, using the information in provided in this example.

The Calculate Pallet Agent receives a message from agent F with the number of pallets requested (30), the date the pallets are needed (15-08-2016) and the incoming transports the two days before the transport leaves (transports 4 and 5). The transports are ordered by how much room they have left in the truck for empty pallets. In this case the transports already carry an equal number of pallets, so the one that is first in the list (ordered by transport ID) is first. So, first transport 4 is examined. The Calculate Pallet Agent sends a message to the Location Agent of the sending location requesting the number of free pallets the day before transport 4 leaves. Transport 4 leaves on 12-08-2016, so it sends a message requesting the free pallets at 11-08-2016. Agent D, as denoted in table 3, has 5 free pallets at its location on 11-08-2016. The transport is updated in the Calculate Pallet Agent (see figure 5a), and the total number of pallets still searching for is now 25. Transport 4 is stored for later examination; the truck is not full yet so if location D can get pallets from other locations it can transport more pallets to location F.

The Calculate Pallet Agent then examines the next transport, which is transport 5. It sends a message to agent E requesting the free pallets at 11-08-2016. Agent E has 30 free pallets at its location on that date. Only 15 pallets still fit in the truck, so these 15 pallets are added to the number of pallets transported (figure 5b) and now only 10 pallets are still needed.

Next, the Calculate Pallet agent requests the number of incoming shipments from location D arriving on 11-08-2016 or 10-08-2016 and starts the recursive step: it starts the algorithm again but then with new input parameters. The goal location is now location D, the transports are the new transports it has just received from location D, and the goal date is now 12-08-2016 (the day transport 4 leaves). The transports the Calculate Pallet Agent receives are transport 1 and 2.

Because transport 1 has the most room in the truck left, the Calculate Pallet Agent first sends a message to agent A requesting the number of free pallets at its location on 08-08-2016. Agent A has 2 free pallets on that day. These pallets are added (figure 5c) and now there's a search for 8 pallets left. Now the Calculate Pallet Agent sends a message to location B requesting the pallets at 08-08-2016. Agent B has 15 free pallets on the requested day; however, only 8 are needed. These 8 are added (figure 5d) and the algorithm returns since all the pallets have been found. The Calculate Pallet Agent sends messages to the Location Agents involved so they can update their internal representation and reserve pallets, and sends a message to the Calculate Pallet Agent to update the transport planning.

Note that because the Calculate Pallet Agent does not have a complete overview of the free pallets at locations, it might be that a sub-optimal solution is found, as we can see in this example. Even though it does find the requested number of pallets in the end, it might have been preferable that first location B would have been examined because then there is only one transport needed to deliver the needed pallets to location D, instead of two.

If there wouldn't have been at least 8 pallets at location B, the next step would have been to go further into recursion with location A as starting point. After that location A, location B would have been a starting point, and after that any of A's children in the tree. Note that even though location E has relevant incoming transports, the search tree has been pruned there because the last transport (transport 5) is already full. If no or only a partial solution would have been found, agent F would have sent the Calculate Pallet Agent a message each day until the day before the pallets are needed. When no solution would have been found, Location Agent F would have requested the number of pallets not found every day again until the day before the transport leaves. At that point the pallets are added to the system (in case of running a simulation) or an alert is send that the location has a pallet deficit (in case of using Pallet real time).

There are several crucial difference of this approach as compared to iterative deepening:

- Partial solutions can be found. This means that there can be multiple transports each carrying only a part of the needed pallets to the goal location, depending on the free pallets at the sending locations and the room available in the trucks.
- Nodes already explored are not explored again. In iterative deepening search, the search tree is generated from the top each time a new iteration starts. In this

case, the partial solution reached at that node is saved. This is an improvement from classical iterative deepening as nodes do not need to be generated again.

- If a certain solution cannot be reached from a node (location) anymore, the node is never considered as a starting point.

Note that actually, the search tree could be a *graph*, but is adjusted to make a *tree*. This is because cycles can occur. However, to avoid anomalies between what the Calculate Pallet Agent thinks it knows about the free pallets at a location and the actual free pallets at a location, it was decided that every location can only send to one other location. Once a transport has been explored, any transports that have the same sending location as any transport explored before are ignored. See example 3 for an example.

Example 3: Excluding transports
Suppose that next to the transport schedule depicted in example 2, the transport depicted in table 4 would be added.

Transport ID	Sender ID	Receiver ID	Loaded pallets	Empty pallets	Pick up by	Deliver by
3a	E	D	30	0	10-08-2016	11-08-2016

Table 4: Added transport in example 3.

This transport would introduce a cycle in the transport graph. It is not included in the search tree, because transport 5 has already been explored, so any transport with location E as sending location is disregarded.

5 Generating the Transport Schedule

Next, regarding sub question 4, a simulation using Pallet needs to be compared to an existing situation to say if Pallet improved efficiency. Ideally, real data gathered by Ahrma would be used for this. However, this data turned out not to be available because there are no large companies that have replaced all their pallets by Ahrma pallets yet; companies that have the Ahrma pallets have typically only replaced a part of their wooden pallets by Ahrma pallets.

Ahrma did supply a data set of a large company based in the United States that manually kept track of their pallet flows for over a year, from August 2016 until July 2017. This dataset will be used as a baseline and the corresponding transport schedule will be used as input for Pallet. The baseline and the simulation using Pallet will be compared to each other.

The dataset only contained pallet flows between locations on a monthly basis. For Pallet the information has to be much more detailed. As a result, a transport schedule based on the dataset had to be generated. The format of the data set as received is displayed in table 5.

Sender ID	Sender Address	Receiver ID	Receiver Address	August 2016	September 2016	October 2016	...
1	*Sender_address*	2	*Receiver_address*	200	150	120	...

Table 5: Example of the format of the data. The sender and receiver address columns contain the real street adress for the location, while the Sender and Receiver ID contain a unique ID per location. The months denote the total number of pallets shipped between the sender and receiver location in the given month.

The data represents a so-called hub-and-spoke model. Hub-and-spoke models are a distribution paradigm in which the hubs, the central locations, connect the spokes, which are the more outlying locations. The hubs serve as a central point for shipments to be collected and then distributed again. Hubs do not only send and receive shipments from spokes, but also from other hubs. Figure 6 contains a visualisation of the data set. The hub-and-spoke paradigm can be seen from this map. The nodes that are larger are the hubs while the smaller nodes are the spokes.

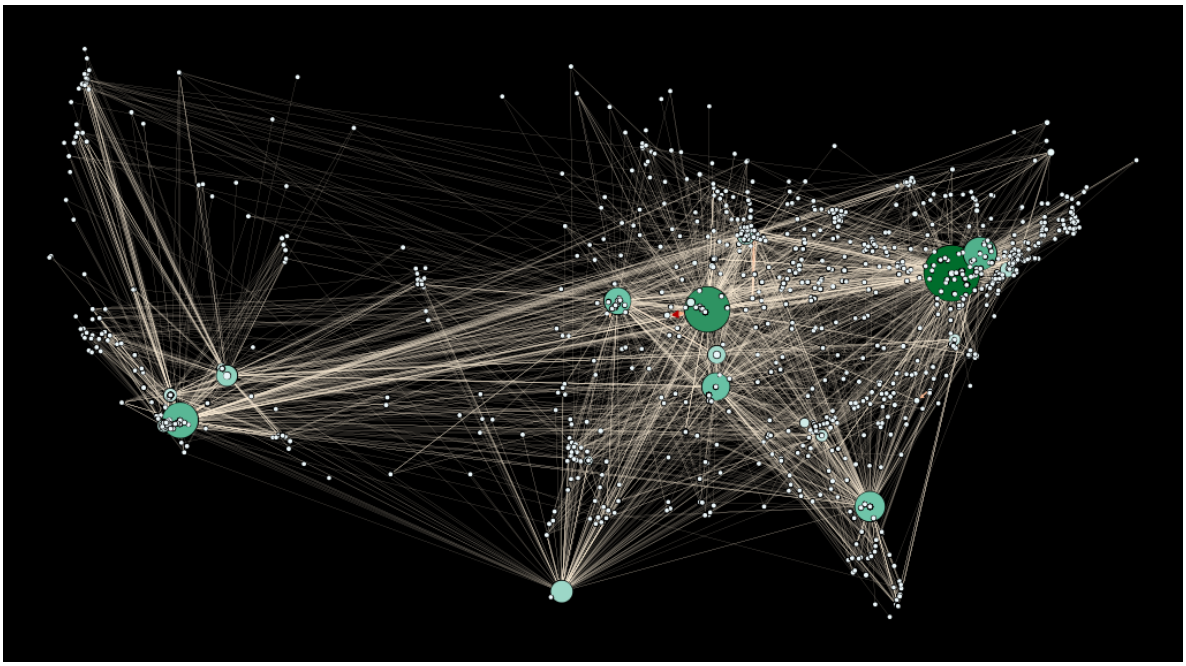


Figure 6: A visualisation of the transport dataset. Pictured is a directed graph of the transport schedule, where each node represents a location. The nodes are placed on the map according to the true geographical location of the location. The edges represent the transport flows, summed over a year. Darker and thicker edges mean more pallets were transported between the locations. Larger nodes represent a larger degree, while the colour of the node represents the out-degree of the node. A darker colour represents a larger out-degree.

The data set describes the movements of roughly 10,000,000 pallets. The data set is not complete. The spokes in the data can be described as being of two categories: manufacturing facilities or *factories* for short, and locations that are of a specific store, or *retailers* for short. The hubs can then roughly be described as *distribution centres*, as

they receive shipments from factories and other distribution centres and send shipments to retailers and distribution centres. The dataset does not contain information about where the factories get their pallets from and similarly it does not contain information about where the retailers leave their pallets they have received. The number of pallets transported between just the distribution centres is roughly 4,000,000. It is a reasonable assumption that at some point in the process the pallets at the retail locations will either be shipped back directly to the factories or to a central point where pallets are collected to be distributed again. However, to avoid making more assumptions than necessary, it was decided to create a transport schedule in which it is assumed that factories have an unlimited supply of pallets and retailers never ship the pallets they receive to another location. The factories and retailers are still in the final transport schedule, but factories only have outgoing shipments and retailers only have incoming shipments (as does the original data set). This means that the only places where the optimisation can happen are *between the distribution centres*.

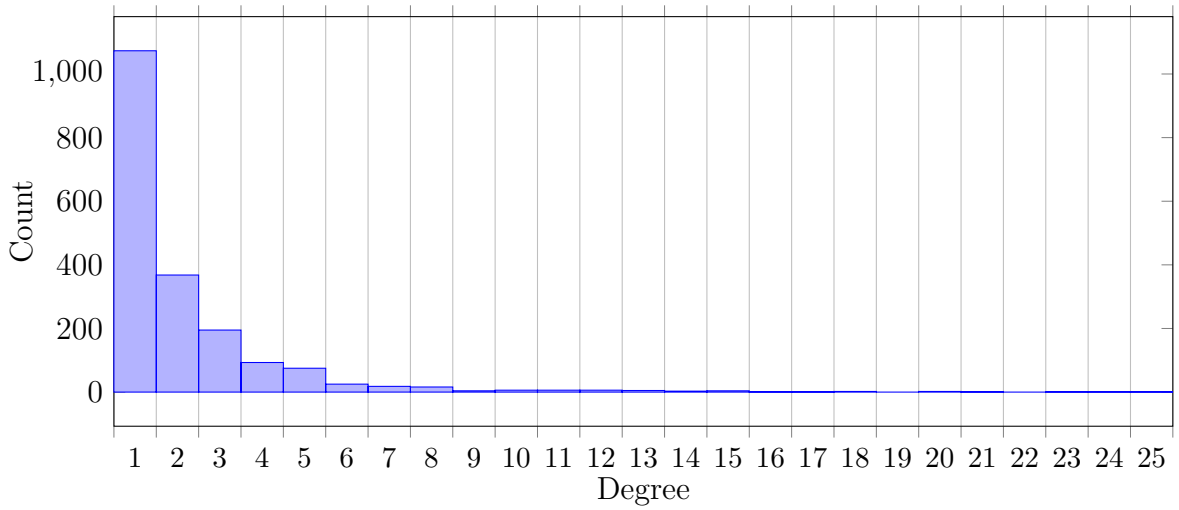


Figure 7: Histogram presenting the degree distribution in the data. There are 29 locations with a degree that is higher than 25; these have not been included in this graph for clarity. All of the locations with a degree larger than 25 are distribution centres.

Figure 7 shows the degree distribution in the data set, where the degree is defined as the number edges to a node. Incoming and outgoing edges each count as a degree, so a location that only sends shipments to one location and receives shipments from the same location has, in total, two degrees. The distribution centres generally have a larger degree than the other locations. There are 1947 locations in the data set, of which 130 (6.7%) are distribution centres. Of the spokes, 200 locations are classified as factories, and 1617 locations are classified as retailers. Even though the distribution centres generally have much more incoming and outgoing transports than the other locations, the relative small amount of distribution centres means the room for efficiency improvement is limited.

Because of this limitation, the actual initialisation for the location agents is slightly

different than what was discussed in section 4.1. All the location agents are still read from the location database, but now have an added type: retailer, factory or distribution. Only location agents of type distribution can send or receive empty pallets. Also, only Location Agents of type distribution send messages to the Calculate Pallet Agent requesting pallets. When a Location Agent sends the Calculate Pallet Agent a message, it only includes shipments it receives from other Location Agents of type distribution.

5.1 Assumptions

To generate the transport schedule, some assumptions had to be made. The assumptions are mentioned below.

- As mentioned in section 4.2, a truck can carry 33 loaded pallets. If a truck is not fully loaded, 15 empty pallets fit in the place of 1 loaded pallet (because empty pallets can be stacked while loaded pallets cannot). This would mean that a truck filled with only empty pallets can transport $15 * 33 = 495$ pallets.
- There is only one type of truck available; even if there are only 10 pallets delivered to a location in a month, it is still done with a full-size truck.
- Truck drivers can drive 700 kilometres in a day. So, if the road distance between locations is 701 kilometres, a transport will take two days.
- The route between locations is symmetric- it takes as much time to drive from location A to location B as it takes to drive from location B to location A.
- Trucks do not drive combined loads; trucks deliver goods to only one location. Also, the trucks do not drive return trips; it is assumed that trucks only ride one way.
- Truck rides are distributed evenly over the month. This means that if there are two truck rides in a month from location A to location B, one will happen at the first of the month while the other one happens around the 15th of the month. If there are three trips in a month, the trips will be scheduled around the first, 15th and 30th of the month. This has the disadvantage that more trips than necessary will depart at the first of every month.
- The pallets are divided evenly over the truck rides. This means that when 40 pallets need to be transported over a month, they are evenly distributed over the two truck rides; instead of one truck transporting 33 pallets and the other 7, both trucks will transport 20 pallets.
- Every day is treated the same; there are not more or fewer transports at weekends or on holidays.
- Even though several domain sources and conversations with domain experts suggest that on average, 40% to 70% do not drive with a full truckload, it was decided to generate the most efficient transport schedule possible that is consistent with

the data set that was supplied. The number of trucks that drive without a full truckload differs widely per industry, and no reliable data for this industry could be found. The transport schedule created is the worst case scenario (regarding room for optimisation).

5.2 Generating the transport schedule

To generate the transport schedule, next to the number of pallets it is also necessary to calculate the time it takes to drive between location. The dataset contained the entire sender and receiver address, which enabled looking up the actual road distance between two locations. The road distance was then divided by 700 and rounded up to get the number of days a transport takes. Then, the actual truck rides were generated by distributing the number of pallets evenly over the number of truck rides needed and then distributing the truck rides evenly over the given month. The pseudocode for the transport schedule generator is displayed in appendix B.
The steps of creating a transport schedule given the distance between locations and the number of pallets transported are as follows:

- Given the number of pallets transported in a month, calculate how many truck rides are needed in that month to transport all the pallets.
- The number of pallets per truck ride is calculated. Note that when the number of pallets is not divisible by 33, not all the truck rides will contain the same number of pallets.
- The truck rides are distributed evenly over the month. Note that when the number of truck rides is not divisible by the number of days in the month, not every day will have the same number of truck rides.
- The 'deliver by' date is the 'pickup by' date plus the number of days the transport takes.

For intuition, several examples are provided below.

Example 4: 450 pallets transported from A to B in January 2017

- Number of trips: $450/33 \approx 14$ (rounded up).
- Divisible number of pallets: $450 - (450 \mod 14) = 448$.
- Standard number of pallets per truck ride: $448/14 = 32$.
- There are 2 pallets left $(450 - 448)$, so that means in January 2017 there will be 2 trips with 33 pallets, and 12 trips with 32 pallets.
- There are 31 days in January, $31/14 \approx 2$ (rounded down). This means that the trips will be on the following days: 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27 January, with the two trucks that contain 33 pallets on the 1st and the 3rd of January.

Example 5: 3410 pallets transported from A to B in February 2017

- Number of trips: $3410/33 \approx 104$ (rounded up).
- Divisible number of pallets: $3410 - (3410 \mod 104) = 3328$.
- Standard number of pallets per truck ride: $3328/104 = 32$.
- There are 82 pallets left $(3410 - 3328)$, so that means in February 2017 there will be 82 trips with 33 pallets, and 22 trips with 32 pallets.
- There are 28 days in February 2017, which means that there are more trips than days. $104/28 \approx 3$ (rounded down).
- Truck rides left: $104 \mod 28 = 20$. The first 20 days of February there are 4 truck rides per day, and the rest of the days there are 3 truck rides per day. The transports containing 33 pallets will all depart in the first 5 days of February

Example 6: 1000 pallets transported from A to B in April 2017

- Number of trips: $1000/33 \approx 31$ (rounded up).
- Divisible number of pallets: $1000 - (1000 \mod 31) = 992$.
- Standard number of pallets per truck ride: $992/31 = 32$.
- There are 8 pallets left $(1000 - 992)$ so that means in April 2017 there will be 8 trips with 33 pallets and 23 trips with 32 pallets.
- There are 30 days in April, which means that there are more trips than days. $31/30 \approx 1$ (rounded down).
- Truck rides left: $31 \mod 30 = 1$. This means that on the 1st of April there are two trips and the rest of the days there is one trip.

The average number of pallets transported per month between locations is 989. The proportion of trucks in a month that transport 33 pallets grows as the total number of pallets transported in a month increases. When the number of pallets transported in a month is larger than 1024, there is no number of pallets for which all trucks in a month will drive partially empty. A more detailed explanation of this is provided in appendix C. As shown in the examples above, this way of planning results in many trucks filled: figure 8 illustrates this.

Approximately 69.16% of the trucks drive with a full truckload of 33 pallets, and over 80% of the pallets drive with at least 32 pallets. In total there are 443,870 transports in the generated transport schedule. There are 278,665 transports between distribution centres (which are the only transports on which empty pallets can be transported). Of these transports, 85.09% carry a full truckload (33 pallets) and 97.54% of transports between distribution centres carry at least 32 loaded pallets.

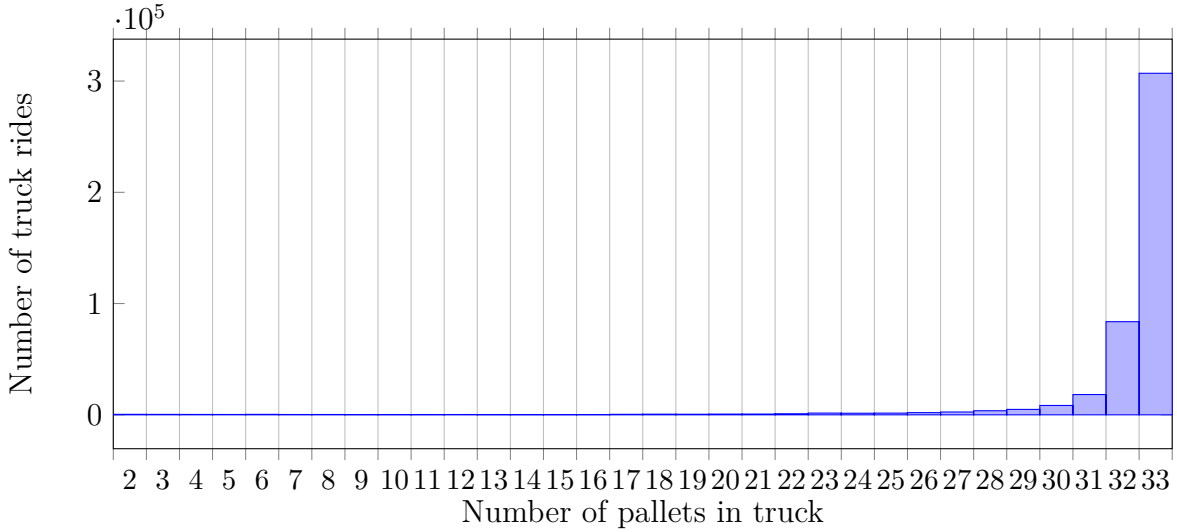


Figure 8: A graph depicting the number of pallets in a truck ride.

6 Determining the efficiency improvement

To determine if PALLET improves efficiency, first, a baseline is needed. For the baseline situation and the simulations, the same transport schedule was used. There are a few key performance indicators (KPIs) that will be used to determine if PALLET improves efficiency. The KPIs are as follows:

1. **The number of pallets needed to run through the transport schedule**. If this number goes down, the same goal is reached but then with fewer pallets.

2. **The average number of pallets at the locations per day**. This is calculated for every location separately. One of the original problems posed in section 1.1 was that sites have too many pallets. So, in line with the KPI above; having fewer pallets at the start of the simulation means that the average number of pallets per location per day goes down. However, the average number of pallets per day per individual location may differ between the baseline situation and the simulations. Some locations might have, on average, more pallets per day in the simulation than in the baseline. It is to be expected that there are more locations that have, on average, a lower number of pallets per day in the simulations as compared to the baseline situation.

3. **The percentage of trucks carrying a full truckload**. In line with the two KPIs above, the number of trucks carrying a full truckload goes up as the efficiency improves. Since only the transports between distribution centres can carry empty pallets, this number is the percentage of transports between distribution centres that carry a full truckload.

These KPIs will be referenced by their respective numbers in the following sections. Note that all the KPIs only concern locations that are distribution centres or transports between distribution centres.

6.1 Baseline

1. KPI 1 was calculated by going through the transport schedule and every time the number of pallets at a location would fall below zero (so the location had more outgoing than incoming shipments), that number of pallets would be added to the location. This is also the minimum number of pallets needed at the start to run through the whole simulation.

2. When the previous KPI was calculated, its result was the number of pallets that needed to be added to each location per day. Using this, an overview of the pallets at each location every day was created. Using this, the average number of pallets per location per day was calculated.

3. The percentage of full truck rides between distribution centres in the baseline is 85.09%.

6.2 Simulation

An important aspect that can influence the performance of Pallet is how far in advance transports are known. Transports between locations may take multiple days. When a Location Agent needs empty pallets, it has more options to get these empty pallets when transports are known further in advance. Unfortunately, there is no universal number on how far in advance transports are planned in practice. Both literary sources and domain experts are scarce on this subject. Moreover, this number differs across different domains. In the on-demand delivery domain, transports are often known a day in advance, while in other sectors transport is more regular and thus are known weeks in advance. To address this issue, how much in advance the transport schedule is known to the agents is modelled as a variable that changes across simulations. This variable can take three different values: transports can be known a week before the transport leaves, two weeks before the transport leaves and three weeks before the transport leaves. The KPIs as discussed in the previous section were calculated in the following ways:

1. KPI 1 is more difficult to calculate than in the baseline situation. When the simulation is started, every Location Agent has 0 pallets at location. When the agents cannot find available pallets from other locations, the pallets that have not been found are added to the location. After a complete simulation, there is an overview of how many pallets could not be found and were added to locations. However, this is not the number of pallets needed at the start of the simulation. Remember that Location Agents only search for pallets when they calculate that they have a deficit in the future. If pallets are added to locations in the beginning of the simulation instead of during the simulation, this causes the Location Agents to not start trying to get pallets from other locations until they need them, which is later than in the original simulation. The outcomes produced by Pallet are non deterministic and hard to predict. This has made finding the number of

pallets needed *at the start of the simulation* an extremely time-consuming process, because it can only be found through trial and error. Moreover, when a solution is found, it is not guaranteed to be optimal. For a complete discussion on this, see appendix D. It was decided to only use the number of pallets added during the simulation instead of the number of pallets needed at the start of the simulation. So, whereas KPI 1 for the baseline also signifies the number of pallets needed at the start to run through the transport schedule, for the simulations this is not the case.

2. Once KPI 1 is calculated, the pallets per day per location are generated. This is used to calculate the average number of pallets per location per day.

3. The updated transport schedule is used to calculate the extra number of pallets transported per truck and the new percentage of full trucks is calculated.

The expectation is that all the KPIs will improve when the transport schedule is known longer in advance. This is because the Agents can now see further in the future, and can use more transports to get empty pallets to them.

6.2.1 Technical details

When Pallet is used in practice, Pallet does not have to change the date itself. Then, it will receive a new transport planning every day and check the time of the machine to decide the date. However, when running a simulation, there is a necessity to change the date when the calculations are done. In Pallet because of the many moving parts, there is no easy way to find out if all the agents are done with their internal calculations. Because the Calculate Pallet is the agent that receives the most messages, it was decided to 'measure' activity of all the agents by the messages the Calculate Pallet Agent receives. When the Calculate Pallet Agent has not received a message for more than 10 seconds, it sends a message to the Central Transport Agent. The Central Transport Agent then changes the date to one day further. Because the transport schedule spans a whole year, this means that every complete simulation lasts around an hour.

The Central Transport Agent reads a new piece of transport planning at the beginning of every day. This has been implemented as a blank line in the transport schedule; every time the Central Transport Agent encounters a blank line in the schedule it stops reading, and at the beginning of the next day it starts reading from that point on again. This means that if the transport schedule is known a week before, the first day of the simulation (which is 01-08-2016), the Central Transport Agent reads all the transports that leave up until 07-08-2016. Every day after that a new day is read.

7 Results

In the following section, simulation 'one week before' will refer to the simulation in which the transport schedule was known one week in advance. Similarly, 'two weeks

before' will refer to the simulation in which the transport schedule was known two weeks in advance, and simulation 'three weeks before' will refer to the simulation in which the transport schedule was known three weeks in advance.

7.1 KPI 1

	Number of pallets added
Baseline	4,101,769
One week before	4,027,587
Two weeks before	4,022,809
Three weeks before	4,022,608

Table 6: Number of pallets added to all distribution locations (summed) in the baseline situation and the three different simulations.

In table 6, the results concerning KPI 1 are displayed. As can be seen in the table, the number of pallets added drops in all the simulations when compared to the baseline situation. When comparing the simulations themselves, however, the decline is not as sharp. The difference between the baseline and the simulation 'one week before' is 74,182, while the difference between simulation 'two weeks before' and 'three weeks before' is only 201 pallets.

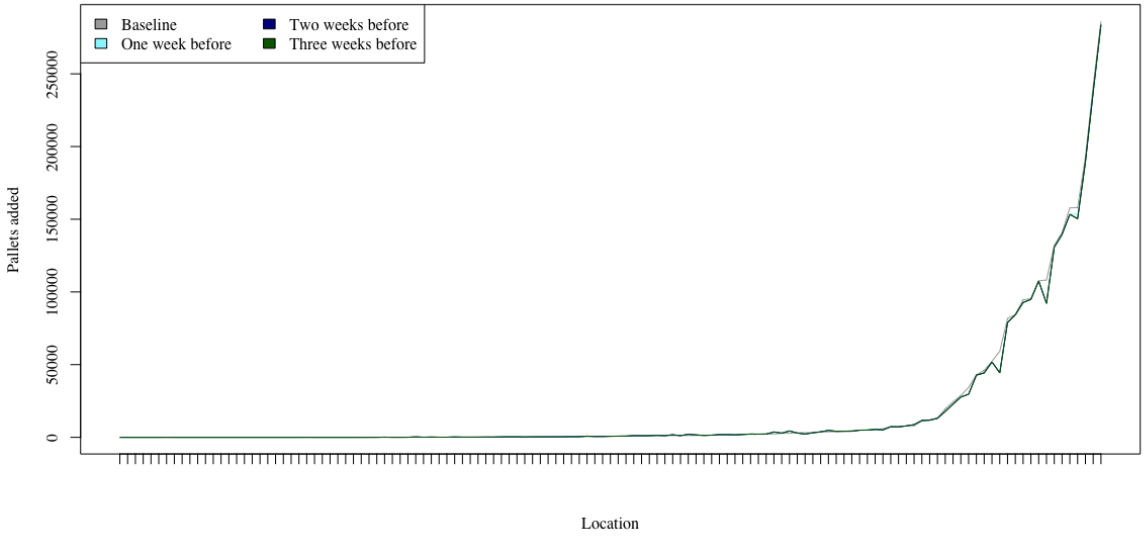


Figure 9: Number of pallets added to the individual distribution locations. Each tick on the x axis signifies a different location. The locations are ordered by the number of pallets added in the baseline situation. The 3 locations with the most pallets added have been omitted for the sake of illustration.

The decrease in pallets is not the same for every location when comparing the baseline situation to the simulations. Some locations even have more pallets added than in the baseline situation. Several explanations of this can be found in appendix D.

Figure 9 shows the number of pallets added per location. As shown, the different simulations barely differ from each other when it comes to numbers of pallets added. The largest difference in pallets added happens at the locations which have many pallets added in the baseline situation. Right before the curve up in figure 9, it shows that there are some locations at which the simulations add more pallets than the baseline situation, although the differences are small.

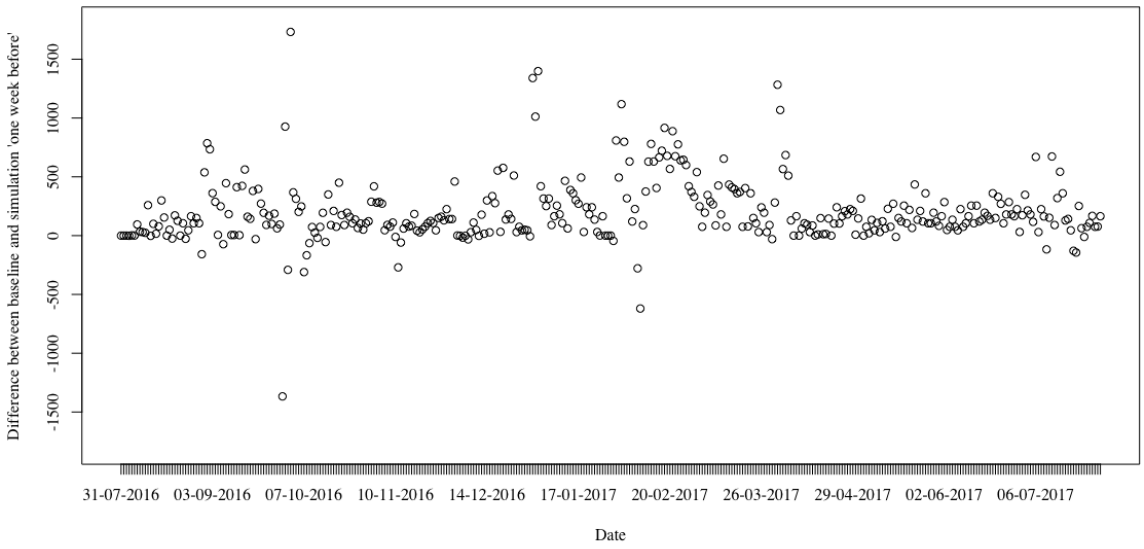


Figure 10: The difference per day between the number of pallets added in the baseline situation and simulation 'one week before', summed over all locations. A negative number means more pallets were added in the simulation than in the baseline situation, and a positive number means that there were fewer pallets added in the simulation than in the baseline situation.

Figure 10 compares the pallets added in total per day in the baseline situation and the simulation 'one week before'. It shows that in the simulation there are not always less pallets added than in the baseline situation. There are some days in which the simulation adds more pallets than the baseline situation, however, the days in which the baseline situation adds more pallets than the simulation are more prevalent.

7.2 KPI 2

Figure 11 shows the average number of pallets per day pe r location. The locations are ordered according the ordering in figure 9, to enable comparing the number of pallets added to a location with the average number of pallets per day at that location. It shows that the simulations follow mostly the same pattern, as the lines plotted for every simulation overlap. For some locations, the average number of pallets per day is higher than in the simulation, while for other locations, the average number of pallets per day is lower than in the simulation.

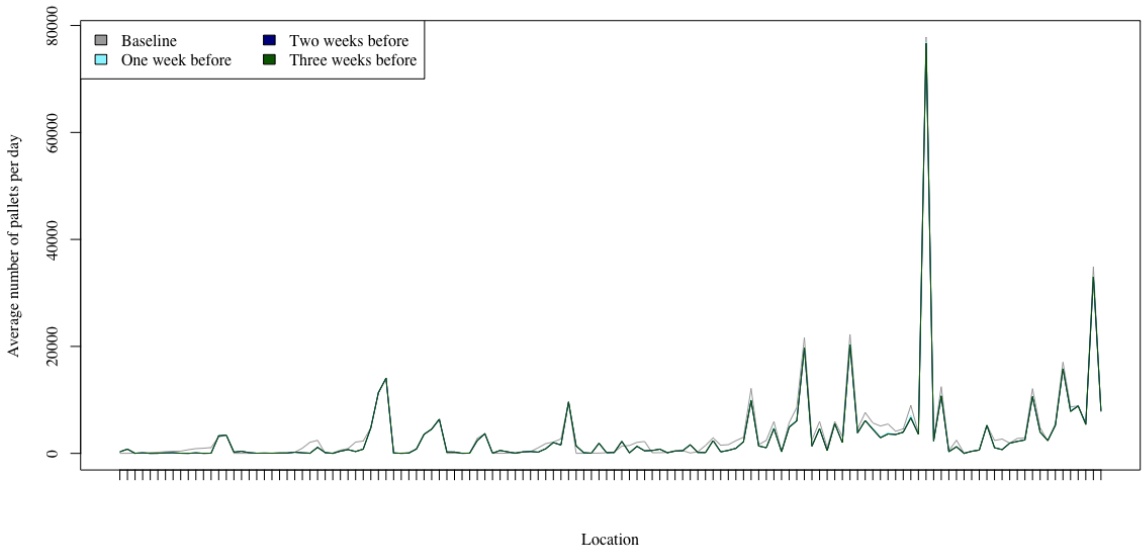


Figure 11: Average number of pallets per location per day. Every tick on the x-axis denotes a location. The locations are ordered according to the ordering in figure 9. The three locations with the most pallet added to it that were not included in figure 9 are included in this figure.

Figure 12 shows a clearer pattern than figure 11. In this figure 12, the locations are ordered according to the average number of pallets per day in the baseline simulation. It shows that for locations that had a low number of average pallets per day, the simulations might cause that number to be higher while for the locations that had a higher average pallets per day in the baseline situation the average number of pallets in the simulation is often lower.

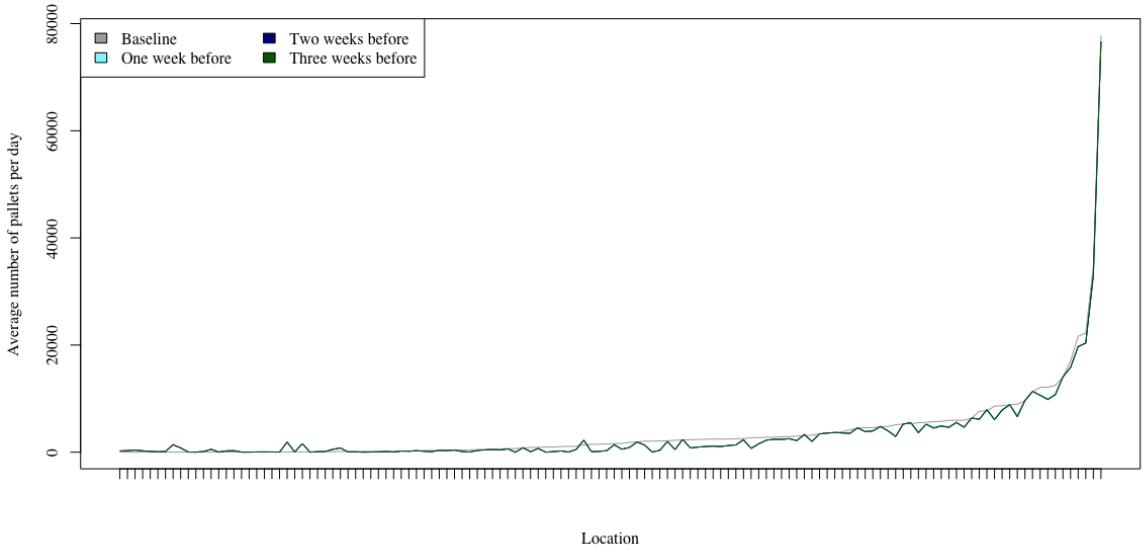


Figure 12: Average number of pallets per day per location per day, plotted for the baseline situation and every simulation. Every tick on the x-axis denotes a location. The locations have been ordered according to the average number of pallets per day in the baseline situation.

7.3 KPI 3

Table 7 shows the number and percentage of transports between distribution centres that are now full. As can be seen, even though there are 4395 more transports that are full when comparing simulation 'one week before' to the baseline situation, this results in an increase of full distribution transports of only 1.58 percentage points.

	Full dist. transports	**Percentage full dist. transports**
Baseline	237,103	85.09%
One week before	241,498	86.67%
Two weeks before	242,035	86.86%
Three weeks before	242,098	86.88%

Table 7: Number of transports with a full truckload. The second column denotes the total number of distribution transports driving with a full truckload, while the third column denotes the percentage of distribution transports that drive with a full truckload.

8 Discussion

The results show some striking patterns. First of all, PALLET does improve efficiency; on all three KPIs, there is an improvement. Secondly, PALLET is robust to changes

in its input. Where there is a slightly more significant difference in results between simulations 'one week before' and 'two weeks before', the difference in results between 'two weeks before' and 'three weeks before' is marginal. The differences between the simulations are mainly due to the long transport times in the transport schedule: transports commonly take three or four days. Receiving transport information only one week before the transport leaves, means that there is an insufficient number of transports available that arrive at the location in time when empty pallets are requested.

Compared to the baseline situation, simulation 'one week before' requires 1.81% pallets less, simulation 'two weeks before' requires 1.93% pallets less, and simulation 'three weeks before' also requires 1.93% pallets less (percentages rounded to two decimals). As shown in figure 10, on some days the pallets added is higher in the simulation than in the baseline situation; however, it much more often happens that the number of pallets added in the simulation is lower than in the baseline situation. On 29-09-2016 there appears a minimum in the graph, while almost immediately after that, on 02-10-2016, a maximum appears. A likely explanation for this is that there were some transports which had much room left, so the Location Agent that needed pallets took them from another agent. The other agent, in turn, had a pallet deficit in the future, causing the pallet deficit to happen first (for the Location Agent that had its pallets taken) and subsequently fewer pallets needed to be added to the Location Agent that took the pallets.

According to KPI 2, the simulations also cause an improvement in the efficiency. As figure 12 shows, even though for some locations the average number of pallets per day is higher in the simulations than in the baseline situation, for more locations the average number of pallets per day is lower in the simulations than in the baseline situation. While it might be expected that there would be a relationship between the number of pallets added to a location and the average number of pallets per day on that location, comparing figure 9 to figure 11 shows no such relation. Calculating the correlation coefficients between average number of pallets per location per day and the number of pallets added per location shows a weak positive correlation coefficient of around 0.27. It can thus be assumed that the number of pallets added to a location and the average number of pallets per day of a location are not, or weakly, correlated.

According to KPI 3, the simulations also improve efficiency. The number of trucks with a full load has increased with 1.58 percentage points when compared to the baseline situation.

In figure 7 the degree distribution was shown for all locations. Analysis was done to see if either the number of pallets added or the average number of pallets per day per location correlated with the degree per location. No clear correlation was found between the degree per location and either of these KPIs.

8.1 Using Pallet in practice

Because of time constraints, Pallet was not integrated into Ahrma's system. Pallet is sound: for every valid input, every solution it outputs is a valid solution. However, its completeness is not guaranteed: there might be cases in which a solution exists, but none is found. Because Pallet has only been used for simulations, it is currently not optimized for receiving the input of the existing Ahrma infrastructure. However, Pallet is programmed in such a way that integrating it in the system should be relatively easy.

8.2 Quality of solution

In every simulation, the number of pallets added during the simulation is less than the number of empty pallets transported. This means that the solutions found are not the most efficient. In simulation 'one week before', the total number of pallets added to the system is 74,182 less than in the baseline situation. However, there are 107,774 empty pallets transported. This means that not all the empty pallets transported directly result in a reduction of the total number of pallets added to the locations. In fact, every empty pallet transported causes 0.69 pallets less to be added, in comparison with the baseline situation. This number would ideally lie closer to 1, meaning that every empty pallet transported causes one pallet fewer to be added to the locations. The cause for this discrepancy is that a Location Agent immediately requests empty pallets when predicts that it might not have enough for a transport in the future. This is, on the one hand, a pro, because it gives Pallet more time to look for free pallets at other locations. On the other hand, there may be other transports coming in on a later date that the Location Agent does not know about yet; when that happens, the empty pallets found are not needed anymore. Ideally, the Location Agent would release the empty pallets it has taken from other agents. However, this does not happen; it turned out to be difficult because the pallets might already be on their way.

8.3 Cost savings

Using fewer pallets also means a cost saving. Ahrma rents out its pallets for an average fee of €0.06 per day. If all the pallets added in the baseline situation would have been rented for the whole year, this would mean the pallet costs would be €89,828,741. If only rent is charged since the day the pallets are added to the locations, the total pallet costs would be €48,654,581. The same costs for the pallets in simulation 'one week before' are, respectively, €88,204,155 and €47,843,854. This means a respective saving of €1,624,86 and €810,727. The savings in the other simulations are only marginally higher. The cost of Ahrma pallets as compared to wooden Euro pallets is still considerably higher if a one-time pallet cost of €10 per wooden Euro pallet is assumed. However, keep in mind that Ahrma pallets offer multiple advantages next to pallet tracking, not the least of them being higher quality pallets (meaning less repair costs). The transponders in the pallets can also measure temperature, which would be beneficial when transporting goods that need to be transported and stored at a certain temperature.

9 Conclusion

The research question posed in section was as follows: *how can the efficiency in pallet allocation be improved with the use of pallet tracking?* To aid the pallet allocation, a Multi-Agent System named Pallet was built using JADE. Every location was assigned a Location Agent, and Location Agents request pallets when they need them to an agent in contact with other Location Agents named the Calculate Pallet Agent. The Calculate Pallet Agent uses an iterative-deepening like algorithm to find transports that have room to transport empty pallets and Location Agents which have a pallet surplus to transport empty pallets to Location Agents that need them for transport in the future.

Using real transport data from a large company in the United States a day-to-day transport schedule was generated. Using this, a baseline situation was calculated with the transport schedule as input. The baseline situation was compared to different simulations, which had the same transport schedule as input. The difference between the simulations was the time Pallet got the information about the transports before the transport left.

The results show that, according to the KPIs set to measure efficiency, Pallet does improve efficiency on all three KPIs. So, Palletis an answer to the research question.

9.1 Future research and improvements

Future research on Pallet can focus on a couple of improvements to the current system. First of all, as described in appendix D, Pallet's results are hard to predict and might not always produce an optimal outcome. Secondly, and this might be regarded in a flaw in the design of Pallet: scarcity is key. When Location Agents have enough pallets at their locations, Pallet will not produce an efficient outcome. Because of this fact, it was assumed during the simulations that when Location Agents cannot find a solution to their pallet deficit through Pallet they can get extra empty pallets from an unknown source at their location anytime. This way, scarcity was artificially introduced into the system, and the Location Agents were forced to look for pallets. However, this is not a realistic assumption, and thus when using Pallet in practice the efficiency improvement might be less. Lastly, Pallet was not introduced in practice. Only simulations with a generated transport schedule were run. Even though precautions were taken to make as little assumptions as possible, making assumptions was necessary and these might not be correct. It is assumed that Pallet can be integrated into Ahrma's system easily but again, this has not been tested.

Next to this, the following list provides some suggestion for new functionalities that could be introduced in Pallet.

- Pallets can of course break and then need to be repaired. A direction for further research could be that pallets break and less pallets are available than expected.

- Trucks can be delayed, in which case the planning needs to be readjusted.
- In line with the above points, Location Agent can use historical data of pallets breaking and trucks being delayed to make predictions taking that into account.
- PALLET now assumes that there is only one type of pallet and one type of truck. This could be adjusted to multiple types of pallets and trucks.
- Incorporating that not all pallets are emptied when arriving into a warehouse.
- Even though privacy has been build into PALLET because all the Location Agents keep their local information private and only share information upon request, this could be expanded. If, for example, multiple companies take part in the same pallet pool, they might need to reconsider how much information they are willing to share with other companies in the same pool. Also, they might want to keep the number of pallets they have private and save some pallets on location as a backup.

Overall, there is a lot to be gained from efficient pallet allocation. The first results using PALLET looks promising. Further research should be carried out to confirm if PALLET also improves efficiency when used in practice.

References

Barell, Sarah. *The history of pallets*. Accessed 01 May 2018. URL: `https://www.1001pallets.com/the-history-of-pallets/`.

Baykasoglu, Adil and Vahit Kaplanoglu (2011). "A multi-agent approach to load consolidation in transportation". In: *Advances in Engineering Software* 42.7, pp. 477–490.

Bellifemine, Fabio Luigi, Giovanni Caire, and Dominic Greenwood (2007). *Developing multi-agent systems with JADE*. Vol. 7. John Wiley & Sons.

Bonisoli, Andrea (2013). "Distributed and multi-agent planning: challenges and open issues." In: *DWAI@ AI* IA*, pp. 41–45.

Bürckert, Hans-Jürgen, Klaus Fischer, and Gero Vierke (2000). "Holonic transport scheduling with teletruck". In: *Applied Artificial Intelligence* 14.7, pp. 697–725.

Chmiel, Krzysztof, Maciej Gawinecki, Pawel Kaczmarek, Michal Szymczak, and Marcin Paprzycki (2005). "Efficiency of JADE agent platform". In: *Scientific Programming* 13.2, pp. 159–172.

Chow, Harry KH, King Lun Choy, and WB Lee (2007). "A dynamic logistics process knowledge-based system–An RFID multi-agent approach". In: *Knowledge-Based Systems* 20.4, pp. 357–372.

Chow, Harry KH, Winson Siu, Chi-Kong Chan, and Henry CB Chan (2013). "An argumentation-oriented multi-agent system for automating the freight planning process". In: *Expert Systems with Applications* 40.10, pp. 3858–3871.

Davidsson, Paul, Lawrence Henesey, Linda Ramstedt, Johanna Törnquist, and Fredrik Wernstedt (2005). "An analysis of agent-based approaches to transport logistics". In: *Transportation Research Part C: Emerging Technologies* 13.4, pp. 255–271.

Dumas, Yvan, Jacques Desrosiers, and Francois Soumis (1991). "The pickup and delivery problem with time windows". In: *European Journal of Operational Research* 54.1, pp. 7–22.

EPAL (2017a). *EPAL Euro pallet*. URL: https://www.epal-pallets.org/eu-en/load-carriers/epal-euro-pallet/. Accessed 6 December 2017.

— (2017b). *The exchange system that works worldwide*. URL: https://www.epal-pallets.org/eu-en/the-success-system/the-system/. Accessed 7 December 2017.

European Commission (2014). *Report from the commission to the European Parliament and the Council. On the state of the Union road transport market*. URL: https://ec.europa.eu/transport/sites/transport/files/modes/road/news/com%282014%29-222_en.pdf.

FIPA. Accessed 13 December 2017. URL: http://www.fipa.org/.

Fischer, Klaus, Jörg P Müller, Markus Pischel, and Darius Schier (1995). "A model for cooperative transportation scheduling." In: *ICMAS*, pp. 109–116.

Harris, Jeffrey S and Jeffrey S Worrell (2008). *Pallet management system: a study of the implementation of UID/RFID technology for tracking shipping materials within the department of defense distribution network*.

iGPS. Accessed 18 December 2017. URL: http://www.igps.net/.

JADE. Accessed 13 December 2017. URL: http://jade.tilab.com/.

Jennings, Nicholas R, Katia Sycara, and Michael Wooldridge (1998). "A roadmap of agent research and development". In: *Autonomous Agents and Multi-Agent Systems* 1.1, pp. 7–38.

Jones, Daniel T, Peter Hines, and Nick Rich (1997). "Lean logistics". In: *International Journal of Physical Distribution & Logistics Management* 27.3/4, pp. 153–173.

Karageorgos, Anthony, Nikolay Mehandjiev, Georg Weichhart, and Alexander Hämmerle (2003). "Agent-based optimisation of logistics and production planning". In: *Engineering Applications of Artificial Intelligence* 16.4, pp. 335–348.

Kravari, Kalliopi and Nick Bassiliades (2015). "A survey of agent platforms". In: *Journal of Artificial Societies and Social Simulation* 18.1, p. 11.

Máhr, Tamás, Jordan Srour, Mathijs de Weerdt, and Rob Zuidwijk (2010). "Can agents measure up? A comparative study of an agent-based and on-line optimization approach for a drayage problem with uncertainty". In: *Transportation Research Part C: Emerging Technologies* 18.1, pp. 99–119.

McKinnon, Alan C and Yongli Ge (2006). "The potential for reducing empty running by trucks: a retrospective analysis". In: *International Journal of Physical Distribution & Logistics Management* 36.5, pp. 391–410.

Ren, Jianwei, Bo Liu, and Zhengyu Wang (2017). "An optimization model for multi-type pallet allocation over a pallet pool". In: *Advances in Mechanical Engineering* 9.5.

Robu, Valentin, Han Noot, Han La Poutré, and Willem-Jan Van Schijndel (2011). "A multi-agent platform for auction-based allocation of loads in transportation logistics". In: *Expert Systems with Applications* 38.4, pp. 3483–3491.

Russell, Stuart and Peter Norvig (2011). *Artificial Intelligence: A Modern Approach, Global Edition*. CL. ISBN: 1292153962.

Smith, Reid G (1980). "The contract net protocol: high-level communication and control in a distributed problem solver". In: *IEEE Transactions on computers* 12, pp. 1104–1113.

Types of trucks and trailers. Accessed 29 May 2018. URL: http://bestshippingservice.co.uk/types-of-trucks-and-trailers/.

Wooldridge, Michael (2009). *An introduction to multiagent systems*. John Wiley & Sons.

Wooldridge, Michael and Nicholas R Jennings (1995). "Intelligent agents: Theory and practice". In: *The Knowledge Engineering Review* 10.2, pp. 115–152.

Appendices

A Pseudocode planning algorithm

Algorithm 1: Initial steps of the search algorithm in the private class `SearchPalletBehaviour` which takes as an argument a message from a location searching for pallets

Data: a message from a location requesting pallets for a certain day

```
/* Structure of message: {senderOfMessage, datePalletsNeeded,
   numberPalletsNeeded, transports} where transports is a list of transports
   the senderOfMessage receives before datePalletsNeeded that have room to
   transport empty pallets. Structure of the objects in
   transports:{senderLocation, receiverLocation, fullPallets, emptyPallets,
   pickUpBy, deliverBy}, where receiverLocation equals senderOfMessage in
   every object and deliverBy is always one or two days before
   datePalletsNeeded.                                                    */
1  totalPalletsStillNeeded ⟵ message.palletsNeeded;
2  goalLocation ⟵ message.senderOfMessage;
3  goalDate ⟵ message.datePalletsNeeded;
4  locationsExludedFromSearch.add(message.senderOfMessage);
   /* locationsExcludedFromSearch contains all the locations that cannot be
      included in the search anymore because they already transport pallets.  Also
      the sender of the message cannot be included because that would not be
      logical                                                               */
5  foundTotalSolution ⟵ RecursivePlan(goalLocation, goalDate, totalPalletsStillNeeded,
     message.transports);
6  if foundTotalSolution == false then
   |  /* the number of pallets for which no solution could be found is in the
   |     global variable totalPalletsStillNeeded                           */
7  |  Send message to the location originally requesting the pallets that (a part of) the pallets has
   |    not been found;
8  if transportsThatCarryExtraPallets != empty then
9  |  foreach transport ∈ transportsThatCarryExtraPallets do
10 |  |  send message to sender to reserve pallets and add to transport.numberOfEmptyPallets;
```

Algorithm 2: Search algorithm finding the available pallets and adds them to the transport

Input: a location id of the location that needs pallets, the date it is needed by, the number of pallets needed by that date and a list of transports.
`/* For an explanation of the objects in transports, see algorithm 1. */`
Output: **true** if all the pallets were found, **false** otherwise.

```
1  Method RecursivePlan(LocationID, dateNeeded, palletsNeeded, transports):
2      sort transports by how many pallets are transported in ascending order so the transports
         that have the most room are considered first
3      foreach transport ∈ transports do
4          Send message to transport.senderLocation requesting the number of free pallets at date
             transport.pickUpBy;
5          waitFor(response);
6          if response.getFreePallets() == 0 then
7              transportsThatCanCarryMorePallets.add(transport);
               /* Loction has no free pallets currently, but might be a candidate
                  for recursion                                                  */
8              continue;
9          locationsExcludedFromSearch.add(transport.getSender);
           /* Exclude all the locations already included in the search process   */
10         Calculate the maximum number of pallets that still fit in the truck
11         if response.getFreePallets >= maximumNumberOfPallets then
               /* There are more free pallets at the location than fit in the truck
                  */
12             availableFreePallets ⟵ maximumNumberOfPallets;
13             roomInTruckLeft ⟵ false;
14         else
15             availableFreePallets ⟵ response.freePallets;
16             roomInTruckLeft ⟵ true;
17         if palletsNeeded < availableFreePallets then
               /* There are actually more pallets available than needed          */
18             transportsThatCarryExtraPallets.put(transport, palletsNeeded);
19             Update all the potential transports that come after this one of the number of pallets
                 added
20             totalPalletsStillNeeded = totalPalletsStillNeeded - palletsNeeded;
21             palletsNeeded := 0;
22             if it exists, update bottleNeck of the receiverLocation to 0;
23             break;
24         else
               /* There are less available pallets than needed                   */
25             transportsThatCarryExtraPallets.put(transport, availableFreePallets);
26             Update all the potential transports that come after this one of the number of pallets
                 added
27             palletsNeeded := palletsNeeded - availableFreePallets;
28             totalPalletsStillNeeded := totalPalletsStillNeeded - availableFreePallets;
29             if it exists, subtract availableFreePallets from bottleNeck of receiverLocation;
30         if roomInTruckLeft == true then
31             transportsThatCanCarryMorePallets.add(transport);
```

Algorithm 3: Continuation of algorithm 2.

```
   /* This algorithm only shows the recursive step of algorithm 2, and is only
      shown here as a separate algorithm for clarity and illustrative purposes.
      */
33
34 if palletsNeeded > 0 and totalPalletsStillNeeded > 0 then
35     palletsNeeded ⟵ totalPalletsStillNeeded + palletsNeeded;
36     if transportsThatCanCarryMorePallets != null then
37         foreach transport ∈ transportsThatCanCarryMorePallets do
38             newTransports ⟵ new List();
39             see if there's a bottleneck on the transport, if the bottle neck is 0 it means
                 either this transport or any transports that come after is full, so delete
                 transport from transportsThatCanCarryMorePallets
40             send message to transport.senderLocation asking for shipments coming in one
                 or two days before transport.pickUpBy;
41             if response == null then
                   /* the location doesn't have shipments coming in                    */
42                 transportsThatCanCarryMorePallets.delete(transport);
43                 continue;
44             newTransports = response.transports;
45             foreach newTransport ∈ newTransports do
46                 if newTransport.senderLocation ∈ locationsExcludedFromSearch then
47                     remove the entry from newTransports;
48                 else
49                     add the transport.transportID to the entry from
                         newTransport.transportID in transportsThatComeAfter;
50             if newTransports == empty then
                   /* there are no incoming shipments that were not in
                      locationsExcludedFromSearch                                     */
51                 transportsThatCanCarryMorePallets.delete(transport);
52                 continue;
53             recursionLocationID = transport.senderID;
54             recursionDate = transport.pickUpBy;
55             break;
               /* found the transport to base the recursion on                        */
56     else
57         return false;
58     Calculate the max number of pallets that the new transport can carry and put them in
         the variable palletsForRecursion
59     RecursivePlan(recursionLocationID, recursionDate, palletsForRecursion,
         newTransports);
60 else
61     return true;
```

B Pseudocode transport schedule generator

Algorithm 4: The algorithm to generate the transport schedule.

Input: The number of pallets transported between locations, distance(in days) between the locations and the month the transport takes place in
Result: A transport schedule for the number of pallets transported between the two locations

```
1  Function GenerateTransportSchedule(pallets, distance, month, senderLocation,
     receiverLocation):
2      howManyRides = ceiling(pallets/33);
3      palletsLeft = pallets % howManyRides;
4      standardNumberOfPallets = (pallets - palletsLeft) / howManyRides;
5      startDate = firstDayOfMonth(month);
6      if howManyRides < daysInMonth(month) then
7          daysBetweenRides = floor(daysInMonth(month)/truckRides);
8          while pallets > 0 do
9              if palletsLeft > 0 then
10                 Add line to transportschedule with the senderLocation, receiverLocation,
                     standardNumberOfPallets + 1, startDate, startDate + distance;
11                 pallets = pallets - standardNumberOfPallets - 1;
12                 palletsLeft = palletsLeft -1;
13             else
14                 Add line to transportschedule with the senderLocation, receiverLocation,
                     standardNumberOfPallets, startDate, startDate + distance;
15                 pallets = pallets - standardNumberOfPallets;
16             startDate = startDate + daysBetweenRides;
17     else
18         truckRidesPerDay = floor(truckRides/daysInMonth(month));
19         truckRidesLeft = truckRides % daysInMonth(month);
20         while pallets > 0 do
21             if truckRidesLeft > 0 then
22                 truckRidesLoop = truckRidesPerDay + 1;
23             else
24                 truckRidesLoop = truckRidesPerDay;
25             for ride in truckRidesLoop do
26                 if palletsLeft > 0 then
27                     Add line to transportschedule with the senderLocation, receiverLocation,
                         standardNumberOfPallets + 1, startDate, startDate + distance;
28                     pallets = pallets - standardNumberOfPallets - 1;
29                     palletsLeft = palletsLeft -1;
30                 else
31                     Add line to transportschedule with the senderLocation, receiverLocation,
                         standardNumberOfPallets, startDate, startDate + distance;
32                     pallets = pallets - standardNumberOfPallets;
33             startDate = startDate + 1;
```

C Proportion of full trucks

Figure 13 shows the proportion of full trucks that drive between two locations in a month plotted against the number of pallets transported that month. As shown in the figure, there is a regularity. The peaks mean that the total number of pallets transported is divisible by 33 because then 100% of the trucks are full. The trend the

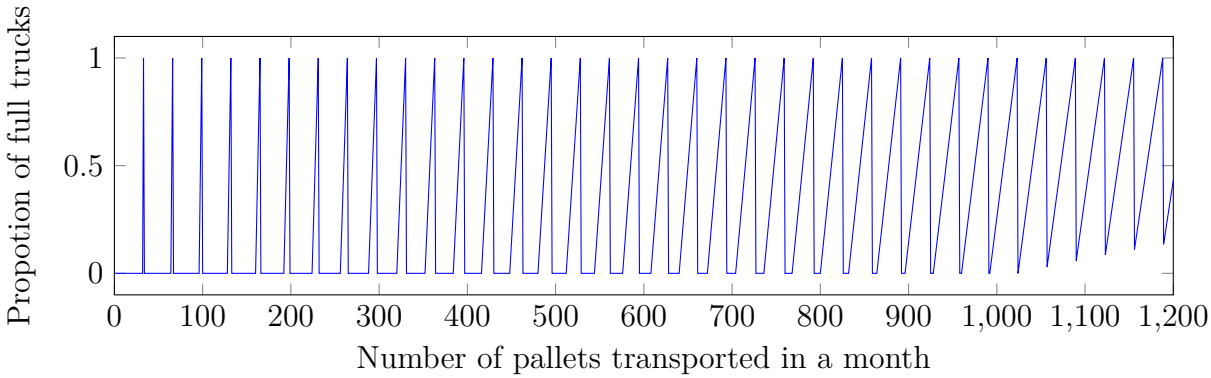


Figure 13: A graph depicting the proportion of trucks that are filled to capacity. The x-axis denotes the number of pallets transported in a certain month between location, and the y-axis denotes the number of trucks that are filled to capacity in that month.

graph shows that if the number of pallets transported in a month grows, the proportion of trucks that are filled to capacity rises.

Example 7: 300 pallets transported in a month
When 300 pallets are transported in a month between locations, there will be 10 trucks each transporting 30 pallets, so 0% of the trucks are filled to capacity.

Example 8: 650 pallets transported in a month
When there are 650 pallets transported in a month there are 10 truck rides carrying 33 pallets and 10 truck rides carrying 32 pallets, so $(10/(10+10) = 0.5)$ 50% of the truck rides are full.

As the graph shows, as the number of pallets transported in a month is larger, the broader the 'peaks' in the graph are. This means that pallets can be distributed better over the truck rides, and as a result from the 'magic number' 1024 $(32 * 32)$ on there is no number of pallets which results in 0% of the truck rides being full.

D Predictability of outcomes produced by PALLET

To make sure the simulation does not halt when no pallets are found, they are simply added to the location at which there is a pallet deficit. When starting the simulations, the initial goal was to find the minimum number of pallets needed at the start of the simulation to run through the simulation. This is mainly because it is not a realistic assumption that pallets can just be transported to locations when needed. However, this proved not to be feasible.

The way PALLET is designed poses a problem here. Because Location Agents only request pallets when they have none at their location, they will not search for pallets when they do have pallets at their location. So, when a Location Agent has pallets at

its location at the beginning of the simulation, it will start to search for pallets later than it would have when the Location Agent would not have had pallets.

Example 9: Needing pallets because of new transport schedule
The start date of the simulation is 01-08-2016. On 15-08-2016 Location Agent A has successfully requested and received 10 empty pallets from another Location Agent by sending a message to the Calculate Pallet Agent. On 20-08-2016, Location Agent A sends the Calculate Pallet Agent a new message for 10 pallets at 30-08-2016 because of a new piece of the transport schedule it has just received. There are no pallets available, so the Location Agent sends a message again the next day, and every day after that until 29-08-2016 but no pallets become available. The pallets need to be added to the location during the simulation.

The simulation is started again, but this time Location Agent A has 10 extra pallets at its location at the start of the simulation. This time, however, the Agent will not request 10 pallets at date 15-08-2016, because it has 10 extra pallets at its location as compared to the last simulation. This causes the Agent to use the 10 pallets added to the system for date 15-08-2016, and get into trouble again at 30-08-2016, resulting in again adding 10 pallets to the system again.

Next to receiving a new transport schedule, a Location Agent can also request pallets because another Location Agent has requested the free pallets at its location, resulting in a future deficit. Remember that Location Agents only reserve their pallets two days before a transport leaves. If another Location Agent requests those pallets before they are reserved, the Location Agent has to look for pallets, because it has reserved pallets it now does not have anymore.

Because of the way the planning algorithm is designed, the Calculate Pallet Agent prefers to take as many pallets as possible from a minimum number of Location Agents. It always asks the Location Agents for pallets in the same order; first, it asks the sending Location Agent of the transport with the potential for most empty pallets. When all transports have the same number of empty places left, it will ask the locations according to whose transport ID comes first. This poses a problem; if a certain Location Agent needs 10 pallets because its pallets have been requested by another agent, the other agent will request more pallets when it has more free pallets at location (until the number of pallets needed is reached or the truck is full).

Example 10: Needing pallets because of other location
Location Agent A needs 10 pallets at 30-08-2016 because Location Agent B has taken 10 of its empty pallets on 15-08-2016. Location Agent B needed more pallets, but Location Agent A had only 10 free pallets. These pallets would otherwise later have been used for transport leaving from location A.

Location Agent A sends a message to the Calculate Pallet Agent to request these 10 pallets, but they are not found. The pallets are added to Location Agent A.

The next time the simulation starts, Location Agent A has 10 extra pallets at its location. Location Agent B, however, needs more than 10 pallets and whereas in the first simulation it got those from another Location Agent, Location Agent B now takes those 10 extra pallets from Location Agent A (so, in total 20 pallets). This causes Location Agent A to still have a deficit of 10 pallets in the second simulation at 30-08-2016, even though 10 extra pallets were added from the beginning.

Another way in which adding pallets to the beginning of the simulation instead of adding pallets during the simulation can have an unexpected effect has to do with the process of reserving pallets. The Location Agents reserve pallets two days before a transport leaves, but if there are not enough pallets two days before, it will only reserve those that are available and then reserve the rest of the pallets only one day before a transport leaves. If extra pallets are added from the beginning because of another Location Agent taking the pallets from the Location Agent, and the Location Agent has not reserved enough pallets two days before a transport leaves, then adding more pallets will only result in adding more pallets to the reservation and still result in a pallet deficit for the Location Agent. For an example, see below.

Example 11: Adding extra pallets results in reserving more pallets
Location Agent A needs 10 pallets at 30-08-2016 because Location Agent B has taken 10 of its empty pallets at 15-08-2016. These pallets would have been reserved at 30-08-2016 for a transport that leaves at 02-09-2016. However, not all the pallets for the transport on 02-09-2016 have been reserved on 30-08-2016; there is a shipment coming in on 01-09-2016 which makes sure that all the pallets for the transport leaving on 02-09-2016 are reserved.

The simulation is started again, but now 10 extra pallets are added to Location Agent A from the beginning. Location Agent A reserves those 10 extra pallets on 30-08-2016, whereas in the previous simulation these 10 pallets were reserved on 01-09-2016. As a result, there are more free pallets at 02-09-2016. However, Location Agent B still takes the same number of (then still) free pallets at 15-08-2016, and Location Agent A still ends up with a deficit of 10 pallets. This is because Location Agent now reserves more pallets at 30-08-2016. When Location Agent B takes 10 pallets at 15-08-2016, Location Agent A has still reserved pallets at 30-08-2016 that were now taken by Location Agent B.

The problem described in the example above could be avoided by

1. reserving the pallets not two days before, but one;
2. by not searching for pallets that were originally reserved for transport leaving two days after the reservation;

The first potential solution was not adopted because it means that the Location Agents have almost no chance of finding other pallets if the pallets are taken two days before because of the long distances. The pallets might be taken from that Location Agent while the Location Agent taking the pallets might have done better by taking the pallets from another Location Agent. The second potential solution has proven to be

difficult: the most complicated part of PALLET is actually in the 'bookkeeping' of the Location Agents. Keeping track of all the pallets expected proved to be more difficult than expected and produced much code. While solution 1 could be implemented, this would produce an extra list and extra things to keep track of.

Because of the many transports and many agents, everything described above makes the outcomes of PALLET hard to predict and complicated to understand. This also ensures that there is no efficient algorithm to decide what the least number of pallets should be at the start of a simulation to fulfil the entire transport schedule. When using PALLET in real situations, there is no need to find the minimum number of start pallets. However, the above problems may also result in suboptimal solutions when used in practice.

There is also an aspect to PALLET that makes it non deterministic. As mentioned in section 2, a key aspect of Multi-Agent Systems is that the actions in the system are synchronised at runtime. This ensures greater flexibility, but also causes the outcome to differ among simulations with the same input because of the way messages are sent and received. When multiple Location Agents send a message to the Calculate Pallet Agent requesting pallets, the order in which the Calculate Pallet agent receives the messages might influence the outcomes.