Scheduling of Intelligent and Autonomous Vehicles under Pairing/unpairing Collaboration Strategy in Container Terminals

Shahin Gelareh
Ecole Polytechnique de Lille, Avenue Paul Langevin, 59655 Villeneuve d'Ascq, France

Rochdi Merzouk
Ecole Polytechnique de Lille, Langevin, 59655 Villeneuve d'Ascq, France

Kay McGinley
Technological University Dublin, kay.mcginley@tudublin.ie

See next page for additional authors

Lagrangian Relaxation for Scheduling of Intelligent and Autonomous Vehicles in Container Terminals Under Pairing/Unpairing Strategy

Shahin Gelareh[a,*], Rochdi Merzouki[b], Kay McGinley[c], Roisin Murray[c]

[a]*LAGIS FRE CNRS 3303, Ecole Polytechnique de Lille, Avenue Paul Langevin, 59655 Villeneuve d'Ascq, France*
[b]*Ecole Polytechnique de Lille, Avenue Paul Langevin, 59655 Villeneuve d'Ascq, France*
[c]*Department of Transport Engineering, Dublin Institute of Technology, Bolton St., Dublin 1. Ireland*

Abstract

A new class of Intelligent and Autonomous Vehicles (IAVs) has been designed in the framework of Intelligent Transportation for Dynamic Environment (InTraDe) project funded by European Commission. These vehicles which are technologically superior to the existing Automated Guided Vehicles (AGV) in different technical aspects offer more flexibility and intelligence in manoeuver in the area where the logistics operations take place. This includes the ability of pairing/unpairing enabling a pair of 1-TEU (Twenty-foot Equivalent Unit) IAVs join and transport any size between a 1-TEU and a 1-FFE (Forty-foot Equivalent) containers. To accommodate this feature, in this article, we extend the classical mixed integer programming model of AGV scheduling in order to minimize the makespan of operations to transport a set of containers of different size between quay cranes and yard cranes. In particular, a case study on Dublin Ferryport Terminal is carried out. In order to cope with the complexity of the scheduling model, we design a Lagrangian decomposition approach utilized with variable fixing procedure and a primal solution heuristics to obtain high quality solution of instances of the problem.

Keywords:
Intelligent Autonomous Vehicle, Automated Guided Vehicle, Mixed Integer Programming, Scheduling, Lagrangian Relaxation

1. Introduction

The ever increasing volume of international tarde which is up to 90 percent fully containerized has demanded for appropriate solutions for several other issues arising across the entire logistics operations and supply chain.
This circulation of huge number of containers is mainly taking place on three main routes: 1) Asia-Europe, 2) Trans-Pacific, and 3) Trans-Atlantic. In total more than 500 ports and tens of liner shipping companies are involved in the global maritime logistics.
Due to the intensive interaction and global-wide spatial distribution of components of the containerized transport system, the inefficiencies in individual parts of the system (both from liner service providers and port authorities points of view) will propagate its negative impact spatially and temporally across the network of systems.

From the liner shipping industry point of view, larger vessels are needed to help the service providers benefit from the economy of scale in transport of the growing volumes. Deploying such vessels is very expensive (often more than several millions dollars per day) and such vessels are generating profit for the owner only during the sailing time and the part of voyage time spent at ports which is referred to as *turn-around time* of a complete voyage at different ports are actually the unprofitable parts. Therefore, the economy of scale may not be exploited unless in long-haul transport. This means that the Liner Service Providers (LSPs) usually do not find it profitable to call too many ports along a service rotation (the so called '*string*'). Consequently, some major transhipment hubs came into play which

*Corresponding author (Tel: +45 4525 3386)
Email address: `shahin.gelareh@polytech-lille.fr;shahin.gelareh@gmail.com` (Shahin Gelareh)

are consolidation and distribution ports and in turn are proxying service to other smaller ports in their region. The selection of such ports depends on several aspects —one of the most important is the efficiency and infrastructure in terms of turn-around time (of course assuming that ports has potential to be a hub port, i.e., it has enough draft for admitting larger vessels and a lot more eco-political aspects).

From the port (equivalently terminal) operators point of view, their port competitiveness is essentially dependent on that they must be able to minimize the turn around time of vessels while maximizing the throughput. In this way they can compete with other neighboring ports and survive in such a highly competitive market. Due to the dynamism of the competition, the goal is not reached unless some strategic, tactical and operational decisions at the terminals being constantly reviewed and addressed in better ways. This includes three main issues: i) the layout and equipment, ii) routing decisions and, iii) scheduling of operations.

Since introduction of containerized transport, several different type of port equipments has been developed which includes several type of cranes (for quay side and yard side) and carriers (e.g. with and without the ability to lift, single and multiple carriers) with several different physical and mechanical characteristics. Alongside with the enhancements in Information and Communication Technologies (ICT) the concept of automated and semi-automated infrastructures came into play and the port authorities started to gradually incorporate ICT in order to improve their efficiencies. Nowadays, the European Container Terminal (ECT), Ports of PSA in Singapore, Kaoshiung in Taiwan, Pusan in Korea, Kawasaki and Kajima in Japan, Thamesport in UK, Bremerhafen and Hamburg in Germany, and Antwerp in Belgium are among the most automated container terminal in the world.

The same trends is still alive and all around the globe several projects are funded to develop new technologies for ports. From among such facilities, the vehicles with a certain degree of intelligence and also autonomy —at the same time— exploiting ICT and equipped with several kind of sensors and Geographical Information System (GIS) tools (see InTraDe[1] as an example) are of special interest.

Although the hub ports are in a breath-taking competition for improving their efficiencies but such a competition is not limited only to them. The smaller ports (e.g the Dublin in our case) are concerned about the efficiency of operation under the current trend of import/export volume while there is almost no chance for any expansion of the ports and terminals due to the land-use and infrastructures in the neighborhood. This suggest investing on technological enhancements eventually leading to more efficient facilities (both transporters and stacking/unstacking). Intelligent Autonomous Vehicles (IAVs) (see Figure 1) are a new class of transporters designed in the framework of InTraDE which generalize the performance of AGVs. A few facts regarding IAVs follows:

- in contrast to AGVs, IAVs do not need to follow signed segment of roads to reach destination and do not have to follow a particular itineraries. Rather, they are MIMO (Multi-Input Multi-Output) systems equipped with several sensors enabling them to benefit from the Geographical Positioning Systems (GPS) and several other sensors to detect the distance to other vehicles, etc,

- the unit capacity of an IAV is one TEU and for transporting any container between 1-TEU and 1-FFE, two IAVs pair in a leader-follower manner to make a 40-foot capacity available to transport the object.

- the IAVs can form platoons based on a leader-follower manner and every IAV can be leader or follower,

- all four wheels possess actuators and a failure to any one of the wheels individually does not stop the vehicle from operation. Rather it runs into a degraded mode of performance in which the operations continues with a lower performance,

- IAVs move lateral and longitudinal without need to a large space to make turns. The wheels offer 360 degree movements,

[1] Intelligent Transportation for Dynamic Environment (InTraDe) http://www.intrade-nwe.eu/

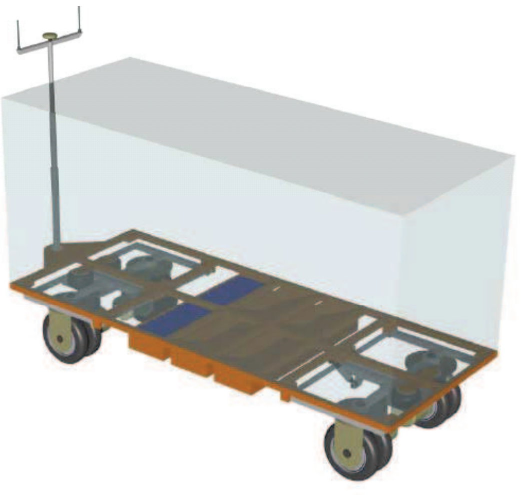

Figure 1: Intelligent and Autonomous Vehicle (IAV).

- Finally, an IAV system should adapt to its surrounding environment, while for the case of the AGV system, the existing environment should adapt to them.

While in total, hundreds of ports all around the globe are involved in the liner transport network, there are only few major LSPs serving some major ports. This makes it quite natural that as far as the professional and academic literature are concerned, there has been a rich body of studies on the optimization and simulation of operations at the terminals.

1.1. Objective and contribution

This paper first rectifies a flaw in the model presented in Ng et al (2007) for AGVs. We show that while the model is infeasible, it can still be corrected and generalized to be applicable for both AGVs and IAVs. We then examine the impact of a class of valid inequalities on the computational performance of a general-purpose solver for solving instances of model. As an inherent property of most of the scheduling problems, only instances of very small size are solvable by general-purpose solvers and we propose a lagrangian decomposition approach equipped with a variable fixing and primal bound generation heuristic to solve instances of the problem for high quality solutions and in reasonable time. Several class of violated valid inequalities are identified and relaxed in Lagrangian fashion to accelerate the convergence. As a case study, we apply the model on the Dublin Ferryport Terminal in Ireland —one of the terminals for which IAVs are designed.

1.2. Literature review

As mentioned earlier, IAVs are technologically superior but very similar to the AGVs. Many differences such as independency from detecting any sign on the earth surface in order to follow a path, the ability of in-place 90 degree rotation of wheels, leader-follower behavior etc., are more relevant in the routing problems. However, from among them the ability to join and cooperate in performing a task is of interest in scheduling and minimization of makespan. The literature is aware of several works dealing with scheduling of AGVs in container terminals and also in manufacturing systems. Meersmans and Wagelmans (2001b,a) proposed a heuristic algorithm for combined AGV and crane allocation problems.

Grunow et al (2004) proposed an online logistics control by using a priority-based approach which is compared against an offline approach. A simulation study of AGVs in an automated container terminal is proposed by Grunow et al (2007) to examine the efficiency of different dispatching strategies.

For other similar works, one can refer to Bish et al (2001) for AGV dispatching and yard allocation, Bish (2003) extended the work to consider loading and unloading scheduling at quay cranes. Bish et al (2005) extended the work in Bish (2003) and adds some analytical performance studies on the proposed algorithm.

Cao et al (2010) proposed a MIP formulation for an integrated yard truck and yard crane scheduling problems while only the import containers were taken into account. They applied a combinatorial Benders decomposition on their model.

Lee et al (2010) studied a transshipment port where both loading and discharging containers are considered. While it has been often simplified by other authors, Lee et al (2010) consider the delays at yard crane as well. The objective is to minimize the makespan of quay side operations as to reduce the turnaround time of vessels. Because the vessel can leave as soon as the last job of quay cranes is finished. The proposed a MIP formulation but two heuristic approaches were used to solve the problem. Lee et al (2010) work is based on Chen et al (2007) but they consider loading and unloading simultaneously.

Ng et al (2007) proposes a MIP model for scheduling a fleet of trucks at a container terminal and the fleet size is assumed to be given exogenously.

However, as we show later, the model in Ng et al (2007) does not always produce a feasible solution to the problem of scheduling AGVs.

Other works on dispatching different equipments at container terminals can be found in Kim and Bae (1999, 2004) for AGV dispatching problem while Kim and Bae (2004) employs a look-ahead strategy which considers local and temporal information of future tasks and assumes a dual cycle operations of AGVs; Nguyen and Kim (2009) extended Kim and Bae (2004) for automated lifting vehicles (ALV); Hartmann (2004) for a wider range of equipments; Narasimhan and Palekar (2002) in which every truck is dedicated to a particular quay crane as opposed to the approach in Kim and Bae (2004).

2. Problem statement

As mentioned earlier, IAVs are intelligent vehicles which can work in groups. That means given a set of individual IAVs it is possible that anyone plays the role of a leader and the rest be connected to it as followers to form a platoon. This resembles the behavior of locomotive and wagons in a train (However, in this article we restrict the size of such a train of vehicles to 2, i.e. the size of a 1-FFE container).

Given such a property, in order to perform a job for an FFE container, two single IAVs must join together and perform the task. That is, if a 35-foot container is going to be imported/exported then two IAVs move towards the corresponding crane to collect it.

The IAV scheduling problem we consider in this article can be described as in Ng et al (2007):

At the earliest ready-time for vehicle $m \in \{1, \dots, M\}$, i.e. t_m the vehicle is at location L_m. There are N tasks and to every task $n \in \{1, \dots, N\}$ a pick-up P_n and a delivery D_n location is associated. There is an approximate travel time $t_{ll'}$ between every two locations $l \in L = \{L_1, \dots, L_M, P_i, \dots, P_N, D_i, \dots, D_N\}$ and $l' \in L' = \{P_i, \dots, P_N, D_i, \dots, D_N\}$ and $t_{ll'} \neq t_{l'l}$, in general. The duration of job i, T_i is the time elapsed from the moment that IAV(s) arrive(s) at the pick-up location of task i, P_i, to the moment that IAV(s) depart the drop-off location of task i, D_i (the average delays both at quay crane and yard crane are implicitly included). The T_i plus the time that IAV(s) travel(s) empty from D_{i-1} to P_i, is the processing time of task i. There is a time at which a crane can generate the task i and expect one or two IAVs to arrive at that location —not earlier. This is referred to by ready-time a_i and $a_i \leq a_{i+1}, \forall i \in \{1, \dots, N-1\}$. IAV scheduling problem seeks for minimizing the makespan such that the turn-around time of vessels are minimized.

3. Mathematical Model

The model is a mixed integer programming with objective function of minimizing the makespan. This means that we are minimizing the completion time of the last task before vessel is ready to depart.

We employ a very similar notation as used by Ng et al (2007).

Parameters. The parameters are listed here:

Parameters
r_m: the earliest time that the IAV$_m$ will be available,
L_m: the initial location of IAV$_m$ at t_m,
T_j: the elapsed time from the arrival of an IAV to the location of pick up until the time that its container being unloaded at the destination,
t_{ij}: the travel time between locations i, j in the terminal,
a_i: the ideal time for performing task i, i.e., the time the task becomes available.
N: the total number of tasks,
M: the total number of available vehicles,
I: the set of all tasks $I = \{i : 1 \leq i \leq N\}$,
I': the set of all tasks $I' = \{i : 0 \leq i \leq N + 1\}$ where 0, $N + 1$-th task are dummy source and sink tasks, respectively,
V: the set of all tasks $V = \{m : 1 \leq m \leq M\}$,
S_i: size of container of task i,
K: a sufficiently big constant (calculated based on known upper bound on optimal solution).

Decision Variables. The decision variables are listed below:

Variables
x_{ijm}: 1, if task j is performed after task i on vehicle m, 0 otherwise,
y_{im}: 1, if task i is performed on vehicle m,
C_i: the completion time of task i,
W: makespan, i.e. the completion time of the last task.

The AGV scheduling problem in Ng et al (2007) follows:

AGV-Scheduling (AGVS)

$$\begin{aligned}
\min \quad & W && && (1)\\
s.t. \quad & C_i \leq W && \forall i && (2)\\
& \sum_{1}^{M} y_{im} = 1 && \forall i \in I && (3)\\
& \sum_{j \in I'} x_{ijm} \leq y_{im} && \forall i \in I, m \in M && (4)\\
& \sum_{i=1}^{N} x_{ijm} \leq y_{jm} && \forall j \in I, m \in M && (5)\\
& 1 \geq x_{ijm} + x_{jim} \geq y_{im} + y_{jm} - 1 && \forall i, j \in I : j \neq i, m \in M && (6)\\
& C_i + t_{D_i,P_j} + T_j \leq K(1 - x_{ijm}) + C_j && \forall i, j \in I : j \neq i, m \in M && (7)\\
& a_i + T_i \leq C_i && \forall i \in I, m \in M && (8)\\
& r_m + t_{L_m,P_i} + T_i \leq C_i && \forall i \in I, m \in M && (9)\\
& x_{ijm}, y_{im} \in \{0, 1\}, i \in I, j \in I', m \in M && && (10)\\
& W \geq 0, C_j \geq 0, j \in I. && && (11)
\end{aligned}$$

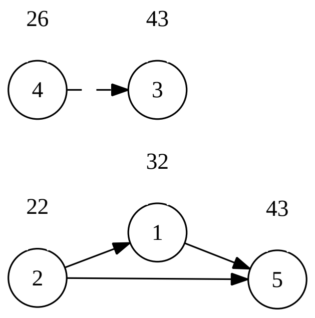

Figure 2: x_{ijm}: task j is performed after i on vehicle m.

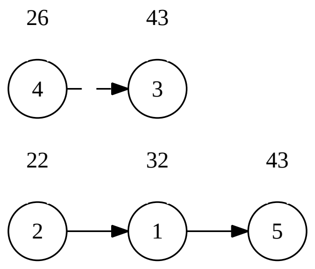

Figure 3: x_{ijm}: task j is performed *immediately* after i on vehicle m.

Remark 1. AGV-Scheduling (AGVS) is infeasible.

The definition of x_{ijm} is confusing. In Figure 2 and Figure 3 we depict an example with 5 tasks and 2 two vehicles (distinguished by connection style) —excluding the dummy source and sink nodes in this representation. Depending on a precise definition of x_{ijm} one of the Figure 2 and Figure 3 is expected.

The issue arises in the definition of x variables. One of the following definitions is correct:

- **the task j is performed after the task i on the vehicle m:** in which the variable $x_{ij'm} = 1$ for all tasks $j' \neq i$ which is completed after completion of i on the same vehicle m. Therefore the total number of links arriving at any non-first task is more than one and this contradicts with (5). This is depicted in Figure 2 (machine 2, with solid connection between tasks) where only $x_{212} = x_{152} = x_{252} = 1$. In such case we have $y_{12} = y_{22} = y_{52} = 1$ and consequently the constraint (5) becomes infeasible by $2 = \sum_{i=1}^{N} x_{ijm} \leq y_{jm} = 1$ for $j = 5, m = 2$.

- **the task j is performed *immediately* after the task i on the vehicle m:** in which the variable $x_{ij'm} = 0$ for all tasks $j' \neq i$ which is completed after completion of i on the same vehicle m. There is no link between two non-consecutive tasks on the same machine which contradicts with (6). This is depicted in Figure 3 (machine 2, with solid connection between tasks) where only $x_{212} = x_{152} = 1$. In such case we have $y_{12} = y_{22} = y_{52} = 1$ and consequently the constraint (6) becomes infeasible by $0 = x_{252} + x_{522} \geq y_{22} + y_{52} - 1 = 1$.

Under of the aforementioned definitions of x_{ijm} the model is infeasible. However, it is still possible to correct the formulation and also extend for being capable of accommodating joint operations of handling an FFE by pairing of vehicles.

Henceforward we use the following definition for x_{ijm}:

x_{ijm}: 1, if task j is performed *immediately* after task i on vehicle m, 0 otherwise.

3.1. Intelligent and Autonomous Vehicle (IAV) Scheduling

In the following we use the term IAV and machine alternatively. We also need to employ two extra dummy tasks: *Source* and *Sink*. The source task is the starting task on every IAV and the sink task is the last task performed on every machine.

IAV Scheduling (IAVS)

$$\min \quad W \tag{12}$$

$$s.t. \quad C_i \leq W \qquad \forall i \in I \tag{13}$$

$$\sum_{m \in M} y_{im} = S_i \qquad \forall i \in I \tag{14}$$

$$x_{0im} + \sum_{j=1, j \neq i}^{N} x_{jim} = y_{im} \qquad \forall i \in I, m \in M \tag{15}$$

$$x_{j\ N+1\ m} + \sum_{j=1, j \neq i}^{N} x_{ijm} = y_{im} \qquad \forall j \in I, m \in M \tag{16}$$

$$x_{ijm} + x_{jim} \leq 1 \qquad \forall i, j \in I : j \neq i, m \in M \tag{17}$$

$$\sum_{i} x_{0im} = 1, \qquad \forall m \in M \tag{18}$$

$$\sum_{i} x_{i\ N+1\ m} = 1, \qquad \forall m \in M \tag{19}$$

$$C_i + t_{D_i, P_j} + T_j \leq K(1 - x_{ijm}) + C_j \qquad \forall i, j \in I : j \neq i, m \in M \tag{20}$$

$$a_i + T_i \leq C_i \qquad \forall i \in I, m \in M \tag{21}$$

$$r_m + t_{L_m, P_i} + T_i \leq C_i \qquad \forall i \in I, m \in M \tag{22}$$

$$x_{ijm} \in \{0, 1\}, (i, j, m) \in (I' \times I' \times M), y_{im} \in \{0, 1\}, (i, m) \in (I \times M) \tag{23}$$

$$W \geq 0, C_j \geq 0, j \in I \tag{24}$$

The objective function (12) minimizes the makespan as the earliest possible time to complete the mission.

Constraints (13) is a minimax constraints to determine the completing time of last event. Constraints (14) indicates that the number of IAVs allocated to the task i must be equal to the size of task. On the same machine, every task (including the source task and excluding the sink) is followed to a consecutive task. In the graph sense from any node representing a task one arc is encompassed. This is considered in (15). Similarly, on the same IAV, every node representing a task in the network is carried out after at another task (including sink and excluding source). Constraints (16) stands for this. If both tasks i and j are carried out by machine m then one proceeds another as stated by constraints (17). The dummy source and sink tasks are performed the first and the last on every IAV in constraints (18)-(19). If task j is performed after task i on machine m then the completion time of task j is at least as late as the completion time of task i plus the travel time from the drop-off location of task i to the pick-up location of task j plus the process time of task j (we say at least because one IAV might need to wait for another one for performing a task). Constraints (20) indicate this. Constraints (21) states that the completion time of task i cannot be earlier than the ready time plus the process time. If task i is the first task assigned to machine m then it cannot be completed before the earliest time that the machine is ready plus the travel time of machine from its initial location to the pick-up location of task i plus the process time of task i as stated in constraints (22).

Figure 4 depict an optimal solution to an instance with $|M| = 3$ and $|I| = 20$. Three IAVs are deployed (IAV1 (solid), IAV2 (dashed) and IAV3 (dotted)). IAV1 and IAV2 cooperate to perform the tasks 1 and 3, while IAV3 does the task 2 independently as it is a 1-TEU container. Then IAV2 decouples from IAV1 after completing task 3 and performs the task 4 and afterwards IAV1 and IAV2 ally to perform task 5. The process continues following the same

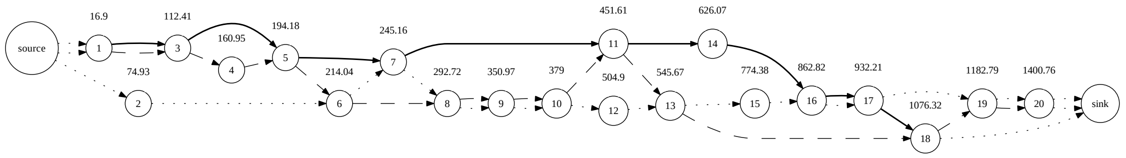

Figure 4: Illustration of optimal solution of an IAV instance.

manner until the final task which is 20 is completed in a cooperation between IAV2 and IAV3 in 1400.67 unit of time.

3.1.1. valid inequalities

If the machines were working independently then we could calculate the accumulated time-in-service of every machine independently and take the maximum as the objective value. However, since in our model two AGV may cooperate on a single machine, therefore in a cooperative environment the vehicles must wait for each others to be able to pair and start ¿1- TEU job. Therefore, unless a machine *m* is only performing on 1-TEU throughout the time containers the value of term:

$$\sum_{i,j\neq i}\sum_{m}(t_{D_i,P_j} + T_j)x_{ijm}, \qquad \forall m$$

does not exactly coincide with the real time-in-service of that particular machine because of the wait-for-pairing time of machines with cooperating jobs. However, this is still a good lower bound on the makespan of the machine *m*.

Let $\phi_i = \max\{a_i + T_i, r_m + t_{L_m\ P_i} + T_i : \forall m \in M\}, \forall i \in I$:

$$W \geq \sum_{i\in I}\phi_i x_{0\ i\ m} + \sum_{i,j\neq i}(t_{D_i\ P_j} + T_j)x_{ijm}, \qquad \forall m \in M \tag{25}$$

It states that in a network representation of problem as in Figure 4, the length of path from source to the node preceding the sink node of every machine is a lower bound on the W.

We show in the numerical section that these inequalities can have a significant impact on the performance of general-purpose solvers.

4. Solution method

IAVS is a challenging model for which even very small size instances with three to four vehicles and 20 tasks are quite time consuming and inefficient to solve by the general-purpose solvers. The major part of the works in literature adopted (meta-)heuristic strategies, often without any indication of quality of such solutions.

Here, we exploit the decomposable structure of the problem and propose a decomposition approach based on Lagrangian relaxation equipped with an efficient local search approach and a variable fixing phase in order to produce lower and upper bound on the optimal solution and obtain an indication of optimality of the solutions.

4.1. Lagrangian Decomposition for IAVS

Lagrangian relaxation for solving (mixed) integer programming problems was first proposed in Fisher (1981, 2004).
The idea behind this method is to relax *complicating constraints* by penalizing the objective function upon violation of these constraints. The relaxed problem is expected to be easier to solve than the original problem and provides a dual bound on the optimal value (as well as valuable information about the dual) of the problem (see Guignard (2003) for a comprehensive survey of the method.).

Three well-known methods are commonly practiced in the literature for solving the Lagrangian relaxation problems. The oldest and most well-known one is the subgradient methods as an iterative methods for solving convex minimization problems. The subgradient method was originally proposed in 60's in the former Soviet Union. Very similar methods has been also proposed in Held and Karp (1971) for solving traveling salesman problem. Later, Lemarechal (1975) proposed the well-known bundle methods as an extension to the subgradient. The volume algorithm was proposed in Barahona and Anbil (2000) as a methods which simultaneously produces a primal feasible solution for the problem and a further analysis of its relation with bundle methods is reported in Bahiense et al (2002).

Several variants of the subgradient methods have been proposed in the literature. Here, based on some observation from the performance of bundle and volume algorithm in presence of big-M, we choose to employ the variant of subgradient proposed in (Larsson et al, 1999) where an Ergodic sequence of subproblem solutions converges to the primal solution set.

We chose to relax constraints (15), (16):
(LRX-IAVS)

$$\begin{aligned}
\min \quad & W + \\
& \sum_{i,m} u^1_{im}\Big(x_{0im} + \sum_{j=1, j\neq i}^{N} x_{jim} - y_{im}\Big) + \\
& \sum_{i,m} u^2_{im}\Big(x_{j\ N+1\ m} + \sum_{j=1, j\neq i}^{N} x_{ijm} - y_{im}\Big) && (26) \\
& && (27) \\
s.t. \quad & C_i \le W && \forall i \in I \quad (28) \\
& \sum_{m \in M} y_{im} = S_i && \forall i \in I \quad (29) \\
& x_{ijm} + x_{jim} \le 1 && \forall i, j \in I : j \neq i, m \in M \quad (30) \\
& \sum_{i} x_{0im} = 1, && \forall m \in M \quad (31) \\
& \sum_{i} x_{i\ N+1\ m} = 1, && \forall m \in M \quad (32) \\
& C_i + t_{D_i,P_j} + T_j \le K(1 - x_{ijm}) + C_j && \forall i, j \in I : j \neq i, m \in M \quad (33) \\
& a_i + T_i \le C_i && \forall i \in I, m \in M \quad (34) \\
& r_m + t_{L_m,P_i} + T_i \le C_i && \forall i \in I, m \in M \quad (35) \\
& y_{im} \le z_m, && \forall i \in I, m \in M \quad (36) \\
& z_m, x_{ijm} \in \{0,1\}, (i,j,m) \in (I' \times I' \times M), y_{im} \in \{0,1\}, (i,m) \in (I \times M) && (37) \\
& W \ge 0, C_j \ge 0, j \in I && (38)
\end{aligned}$$

In this relaxation the lagrangian multipliers are chosen to be $u^1_{im} \in \mathbb{R}, u^2_{im}\mathbb{R},\ \forall\ i, m.$

4.1.1. More constraints to relax

From Figure 5 one observes that in a solution of a given iteration of subgradient the above relaxation, there is no arc arriving to nodes 5,6,7 and 8 and no arc departing from 1, 3 and 4 as it is not enforced by any constraint. Moreover, there is no constraint to enforce that there must be one job (even the dummy source and sink wherever applies) before and after any (non-dummy) task.

In order to encourage this and improving the convergence of subgradient we also dualize the following constraints:

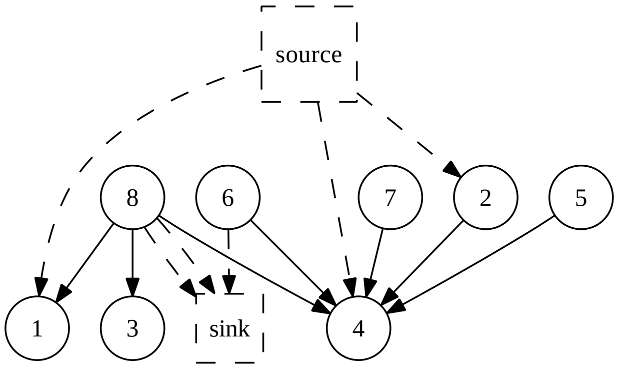

Figure 5: There is no arc arriving to nodes 5,6,7 and 8 and no arc departing from 1, 3 and 4.

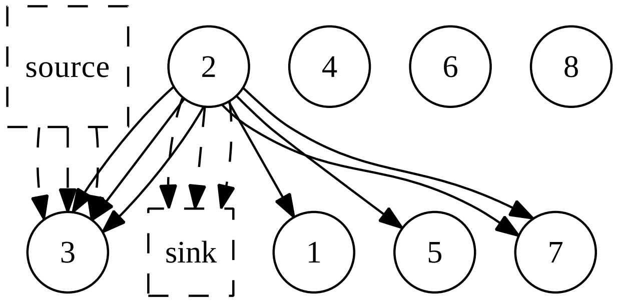

Figure 6: The total number of arc arriving to a task node is at least equal to the size of task.

$$x_{0\ i\ m} + \sum_{j \neq i,\ m} x_{ijm} = x_{i\ N+1\ m} + \sum_{j \neq i,\ m} x_{ijm}, \qquad \forall i \tag{39}$$

with $u_i^3 \in \mathbb{R},\ \forall\ i$.

In this particular example in Figure 5, three tasks namely 2, 6 and 8 are 2-TEU tasks and the rest are 1-TEU tasks. But the number of arcs arriving to all these three are zero. Therefore we also consider dualizing the following constraints:

$$\sum_{j \neq i,\ m} x_{jim} \geq S(i), \qquad \forall i \tag{40}$$

However, extensive computational experiment revealed that the following less tighter constraints provide better results from the bound quality point of view:

$$\sum_{j \neq i,\ m} x_{jim} \geq 1, \qquad \forall i \tag{41}$$

Similarly, the following constraints:

$$\sum_{j\neq i,\ m} x_{ijm} \geq 1, \qquad \forall i \tag{42}$$

are dualized using $u^4, u^5_i \in \mathbb{R}^-,\ \forall\ i \neq j$ multipliers:

Theorem 1. *The total number of arcs ($x_{ijm} = 1,\ i, j \neq i, N{+}1$) in an optimal solution is within $N{-}M \leq |E| \leq \sum_i S_i{-}1$.*

Proof. If there is no job i where $S(i) = 2$ then we have:

a. if the total number of tasks is equal to the total number of machines and every machine does one task, therefore, the total number of arcs is 0. The lower bound obtained.

b. if all the tasks are carried out by one single machine therefore the only tree contains $N - 1$ arcs and upper bound is obtained.

On the other hand if there is at least one task i with $S(i) = 2$ then:

a. if the total size of tasks (total w.r.t the size) is equal to the total available capacity such that every machine does only one task then $M > N$ and $N - M < 0$ and the lower bound is valid.

b. if no machine participate in performing a task i with $S(i) = 2$ as its first task then the total number of arcs is equal to $\sum_i S_i$.

The Theorem 1 can be presented as following constraints:

$$\sum_{i,j\neq i,m} x_{jim} \geq N - M = q \tag{43}$$

This constraint is added to the model.

4.1.2. decomposition

To facilitate the resolution for larger instances we decompose the $(LRX - IAVS)$ into two sub-problems by taking into account that the sequencing part and the scheduling parts are linked only by the big-M constraints. Therefore we add a duplication constraints and dualize them using $u^6_{ijm} \in \mathbb{R},\ \forall\ m, i \neq j$:

$$x_{ijm} = x'_{ijm} \qquad \forall i, j, m \tag{44}$$

$$x'_{ijm} \in \{0, 1\} \tag{45}$$

By doing so, we obtain two problems, one for sequencing (LRX-IAVS-Seq in the space of binary x, y) and one for scheduling (LRX-IAVS-Sch space of continuous W, C).

The resulting relaxation follows: (LRXS-IAVS)

$$\begin{aligned}\min\quad & W+\\ & \sum_{i,m} u^1_{im}(x_{0\ im} + \sum_{j=1,j\neq i}^{N} x_{jim} - y_{im})+\\ & \sum_{i,m} u^2_{im}(x_{i\ N+1\ m} + \sum_{j=1,j\neq i}^{N} x_{ijm} - y_{im})+\end{aligned}$$

$$\sum_i u_i^3(x_{0\ i\ m} + \sum_{j\neq i,\ m} x_{jim} - x_{i\ N+1\ m} - \sum_{j\neq i,\ m} x_{ijm})+$$
$$\sum_i u_i^4(1 - \sum_{j\neq i,\ m} x_{jim})+$$
$$\sum_i u_i^5(1 - \sum_{j\neq i,\ m} x_{jim})+$$
$$\sum_{i,j,m} u_{ijm}^6(x_{jim} - x'_{jim})+ \tag{46}$$

$$s.t.\quad C_i \leq W \qquad \forall i \in I \tag{47}$$
$$\sum_{m\in M} y_{im} = S_i \qquad \forall i \in I \tag{48}$$
$$x_{ijm} + x_{jim} \leq 1 \qquad \forall i, j \in I : j \neq i, m \in M \tag{49}$$
$$\sum_i x_{0im} = 1, \qquad \forall m \in M \tag{50}$$
$$\sum_i x_{i\ N+1\ m} = 1, \qquad \forall m \in M \tag{51}$$
$$C_i + t_{D_i,P_j} + T_j \leq K(1 - x'_{ijm}) + C_j \qquad \forall i, j \in I : j \neq i, m \in M \tag{52}$$
$$a_i + T_i \leq C_i \qquad \forall i \in I, m \in M \tag{53}$$
$$r_m + t_{L_m,P_i} + T_i \leq C_i \qquad \forall i \in I, m \in M \tag{54}$$
$$y_{im} \leq z_m, \qquad \forall i \in I, m \in M \tag{55}$$
$$\sum_{i,j\neq i} x_{jim} \geq q, \qquad \forall m \in M \tag{56}$$
$$\sum_{i,j\neq i,m} x_{jim} \geq N - M = q \tag{57}$$
$$z_m, x_{ijm} \in \{0,1\}, (i,j,m) \in (I' \times I' \times M), y_{im} \in \{0,1\}, (i,m) \in (I \times M) \tag{58}$$
$$W \geq 0, C_j \geq 0, j \in I \tag{59}$$

The problem separates into 4 independent problems:

a. Scheduling Sub-Problem with continuous variables:

(LRX-IAVS-Sch)

$$\min \quad W - \sum_{i,j,m} u_{ijm}^6 x'_{jim}$$
$$s.t.\quad C_i \leq W \qquad \forall i \in I \tag{60}$$
$$C_i + t_{D_i,P_j} + T_j \leq K(1 - x'_{ijm}) + C_j \qquad \forall i, j \in I : j \neq i, m \in M \tag{61}$$
$$a_i + T_i \leq C_i \qquad \forall i \in I, m \in M \tag{62}$$
$$r_m + t_{L_m,P_i} + T_i \leq C_i \qquad \forall i \in I, m \in M \tag{63}$$
$$W \geq 0, C_j \geq 0, j \in I \tag{64}$$
$$x_{i,j,m} \in (I \times I \times M) \tag{65}$$

b. A semi-knapsack problem:

(LRX-IAVS-Seq)

$$\begin{aligned}
\min \quad & \sum_{i,m} u^1_{im} \sum_{j=1, j\neq i}^{N} x_{jim} + \sum_{i,m} u^2_{im} \sum_{j=1, j\neq i}^{N} x_{ijm} + \sum_{i} u^3_{i} \sum_{j\neq i,\ m} x_{ijm} - \\
& \sum_{i} u^3_{i} \sum_{j\neq i,\ m} x_{jim} - \sum_{i} u^4_{i} \sum_{j\neq i,\ m} x_{jim} - \\
& \sum_{i} u^5_{i} \sum_{j\neq i,\ m} x_{jim} + \sum_{i,j,m} u^6_{ijm} x_{jim} + \sum_{i} u^4_{i}(1) + \sum_{i} u^5_{i}(1) && (66) \\
s.t. \quad & x_{ijm} + x_{jim} \leq 1 && \forall i, j \in I : j \neq i, m \in M \quad (67) \\
& \sum_{i, j\neq i} x_{jim} \geq q, && \forall m \in M \quad (68) \\
& x_{ijm} \in \{0, 1\}, (i, j, m) \in (I' \times I' \times M) && (69) \\
& W \geq 0, C_j \geq 0, j \in I && (70)
\end{aligned}$$

c. A semi-assignment problem which is again decomposable for source and sink:
(LRX-IAVS-SourceSink)

$$\begin{aligned}
\min \quad & \sum_{i,m} u^1_{im} x_{0\ im} + \sum_{i,m} u^2_{im} x_{i\ N+1\ m} + \sum_{i} u^3_{i} x_{0\ i\ m} - \sum_{i} u^3_{i} x_{i\ N+1\ m} \\
& && (71) \\
s.t. \quad & \sum_{i} x_{0im} = 1, && \forall m \in M \quad (72) \\
& \sum_{i} x_{i\ N+1\ m} = 1, && \forall m \in M \quad (73) \\
& x_{i\ N+1\ m}, x_{0\ i\ m} \in \{0, 1\}, (i, j, m) \in (I' \times I' \times M), y_{im} \in \{0, 1\}, (i, m) \in (I \times M) && (74)
\end{aligned}$$

d. A knapsack problem:
(LRX-IAVS-y)

$$\begin{aligned}
\min \quad & -\sum_{i,m} u^1_{im} y_{im} - \sum_{i,m} u^2_{im} y_{im} && (75) \\
s.t. \quad & \sum_{m\in M} y_{im} = S_i && \forall i \in I \quad (76) \\
& y_{im} \in \{0, 1\}, (i, m) \in (I \times M) && (77) \\
& && (78)
\end{aligned}$$

where $V^*(LRX - IAVS) = V^*(LRX - IAVS - Sch) + V^*(LRX - IAVS - Seq) + V^*(LRX - IAVS - SourceSink) + V^*(LRX - IAVS - y)$ and $V^*(.)$ stands for the optimal value.

Three out of the all fours separated problems are binary problems which can be solved by inspection without resorting to any LP/MIP solver. The only different problem is $(LRX - IAVS - Sch)$ which does not show any integrality property according to our extensive numerical experiments and can be efficiently solvable by using a MIP solver.

4.1.3. algorithms for solving subproblems

Except for $(LRX - IAVS - Sch)$, the rest of problems namely, $(LRX - IAVS - Seq)$, $(LRX - IAVS - SourceSink)$ and $(LRX - IAVS - y)$ can be solved by inspection.

In the following we outline specialized algorithms for each one.

Algorithm for (**$LRX - IAVS - Seq$**). Let α_{ijm} be the coefficient of x_{ijm} in ($LRX - IAVS - Seq$) after re-ordering terms. The problem does not have too many constraints:

Algorithm 1: Inspection algorithm for ($LRX - IAVS - Seq$).

```
Input: lagrangian multipliers
Output: $x_{ijm}, \forall i, j, m$
$counter = 0$ ;
for $s = 1$ to $q$ do
    $(i, j, m) = argmin\{\alpha_{ijm} : i, j \neq i \in I, m \in M\}$;
    $x_{ijm} := 1$;
    $x_{jim} := 0$;
    $\alpha_{ijm} = \infty$;
```

This algorithm finishes in linear time.

Algorithm for (**$LRX - IAVS - SourceSink$**). Let β_{0im} be the coefficient of x_{0im} and $\beta_{i\ N+1\ m}$ the coefficient of $x_{i\ N+1\ m}$. Then let $(i', m') = argmin\{\beta_{0im} : i \in I, m \in M\}$, we set $x_{0i'm'} = 1$. Also let $(i'', m'') = argmin\{\beta_{i\ N+1\ m} : i \in I, m \in M\}$, we set $x_{0i''m''} = 1$. The rest of variables fixed to zero.

Algorithm 2: Inspection algorithm for ($LRX - IAVS - SourceSink$).

```
Input: lagrangian multipliers
Output: $x_{0im}, x_{i\ N+1\ m}, \forall i, m$
$counter = 0$ ;
for $m \in M$ do
    $(i', m') = argmin\{\beta_{0im} : i \in I, m \in M\}$;
    $x_{0i'm'} := 1$;
for $m \in M$ do
    $(i'', m'') = argmin\{\beta_{i\ N+1\ m} : i \in I, m \in M\}$;
    $x_{i\ N+1\ m} := 1$;
```

This algorithm finishes in $O(M)$.

Algorithm for (**$LRX - IAVS - y$**). Let γ^y_{im} be the cost of variable y_{im}. If $S(i) = 1$ then $(i, m') = argmin\{\gamma^y_{im} : m \in M\}$, $y_{im'} = 1$. If $S(i) = 2$ then $(i, m'') = argmin\{\gamma^y_{im} : m \neq m'\}$ and $y_{i'm'} = y_{i''m''} = 1$.

Algorithm 3: Inspection algorithm for ($LRX - IAVS - y$).

```
Input: lagrangian multipliers
Output: $y_{im}, \forall i, m$
for $i \in I$ do
    $(i, m') = argmin\{\gamma^y_{im} : m \in M\}$;
    $y_{im'} = 1$;
    if $S(i) = 2$ then
        $(i, m'') = argmin\{\gamma^y_{im} : m \neq m', m \in M\}$;
        $y_{i'm'} = y_{i''m''} := 1$;
```

This algorithm finishes in $O(N)$.

4.2. Variable Fixing

In the course of subgradient optimization, after a few steps when the multipliers are stabilized we have lower bound obtained from the Lagrangian relaxation, LB^{LRX}, and also an upper bound ,UB^{heur}, using a heuristic which is described in the following.

Given the reduced cost of a binary variable var, if reduced cost $RC^{var} > UB - LB$ the this variable will not take 1 in any optimal solution and we can remove its column from further computations and get a reduced problem.

We give priority to $y_{im},\ \forall i \in I, m \in M$ variables for testing the possibility of elimination. Because, eliminating one y_{im} implies reduction of all variables x_{ijm} and $x_{jim}\ \ \forall j \in I', m \in M$ which is quite significant and makes the problem size iteratively smaller and resolution becomes easier.

We perform this test whenever lower bound improves in the course of subgradient.

4.3. Primal Bound

A heuristic algorithm is needed to exploit the information obtained from the LR model to produce high quality feasible solution while being computationally very viable. For every given solution to the LR the assignment of jobs to the machines are determined in $(LRX - IAVS - y)$ while the sequencing part might not make a complete or even feasible solution.
The basic idea behind the heuristic is to do the following:

- accepting the assignment reported by $(LRX - IAVS - y)$ and optimize for sequence by taking into account the tasks which need to be done in cooperation between two machines,
- optimizing for the assignment which inevitably will lead to a sequence optimization.

We employ a local search approach which tries to re-assign the jobs in a systematic way aiming at minimizing the makespan for every machine.

4.3.1. initial solution

We start by distributing the jobs $(\omega_i, \forall i \in I)$ between the machines in a way that the first tasks of all machines have a_i very close to each others. We follow the same pattern for all the tasks. Let $A = \{1, 4, 7, 11, 15, 16, 17, 19, 21\}$ and $M = 3$ then we distribute the jobs as following:

$$\omega_1(a_1 = 1) \rightarrow \omega_4(a_4 = 11) \rightarrow \omega_7(a_7 = 17)$$
$$\omega_2(a_2 = 4) \rightarrow \omega_5(a_5 = 15) \rightarrow \omega_8(a_8 = 19)$$
$$\omega_3(a_3 = 7) \rightarrow \omega_6(a_6 = 16) \rightarrow \omega_9(a_9 = 21)$$

Let assume that $S(\omega_4) = S(\omega_8) = 2$ then the following pattern applies:

$$\omega_1(a_1 = 1) \rightarrow \omega_4(a_4 = 11) \rightarrow \omega_6(a_6 = 16) \rightarrow \omega_8(a_8 = 19)$$
$$\omega_2(a_2 = 4) \rightarrow \omega_4(a_4 = 11) \rightarrow \omega_7(a_7 = 17) \rightarrow \omega_9(a_9 = 21)$$
$$\omega_3(a_3 = 7) \rightarrow \omega_5(a_5 = 15) \rightarrow \omega_8(a_8 = 19)$$

where a duplicate of every job with size > 1 is present in the representation. By doing so we are trying to distribute the jobs more equally such that the variance in the completion time of last task on machines is reduced.

4.3.2. neighborhood structure and move strategies

We employ two kind of moves: *Temporal* and *Spatial*. In temporal moves, the sequence of performing tasks on the same machine is modified while in the spatial moves, a jobs will be assigned to different machine(s).

Temporal Move. A temporal move is a move which transforms the current solution to another solution by putting forward or postponing a job by only one step, if feasible. That is, if job ω_i is the j-th job on machine m then the temporal moves will result in a solution having ω_i the $(j-1)$-th or $(j+1)$-th job on the same machine.

Spatial Move. A spatial move is a move which transforms the current solution to another solution by changing the machine to which it is assigned to. That is, if job ω_j is the j-th job on machine m then the temporal moves will result in a solution having ω_i the j'-th job on another machine m' (the choice of j' is rather biased towards a greedy approach) minimizing possible increase in the completion time of every machine.

Search Strategy. Our main emphasis is on distributing the jobs on machines such that the machines finish their final tasks very close to each other. This helps to avoid having a few machines of heavily loaded with long makespan and the rest being less occupied and significantly shorter makespans. In order to achieve this goal, we first must ensure that, given our neighborhood structure and a greedy search, there is no other sequence better than the current one which makes the makespan for the given machine shorter. This means we employ a two-level search. In the first level we only employ temporal moves for each machine in order to obtain high quality sequences, for a given assignment. We try to greedily re-sequence the jobs by finding the best place of each job starting from the firsts to the last one in the current solution. It must be noted that for the jobs with $S(\omega_i) = 2$ any re-sequencing will result in the change of makespan in the collaborating machines. Therefore, except for the machines which are not cooperating at all with any other machine, the rest of machines are analyzed lexicographically to avoid any confusion. In the second phase, spatial moves are performed in which we start from the machines with the highest makespan and try to re-assign the tasks to another machines with less makespan and subsequent iterations of temporal moves is applied. Of course the issues related to the jobs with size 2 are taken into account.

Tabu list. We employ a tabu list which keeps track of the moves in the search space. A spatial or temporal move become forbidden for η^l, η^m iterations, repeatedly. That is if a move has caused a jobs on machine m being postponed or put forward, then the reverse move will be forbidden for η^t iterations. This analogously applies to the spatial moves.

Scape strategies. The greedy approaches often are in the danger of trapping in local optima and facing a premature convergence. In order to avoid that, we incorporate some degree of randomness in our approach. In moving from one solution to a neighboring solution, we will accept even degrading solution if the value of $e^{(-\frac{\Delta f}{\log iteration})}$ is bigger than θ^{tr} and reject otherwise.

Termination criteria. The termination criteria is set to a total number of consecutive non-improving iterations.

5. Numerical results

We have generated instances with perturbed data of Dublin Ferryport Terminal (DFT). Instances range from 10 to 400 jobs and number of deployed IAVs are in $\{1, 2, \dots, 6\}$. Through extensive experiments on instances of problem we observed that our heuristic is robust against changes in η^l and η^m and therefore we have fixed these two values to $\eta^l = \eta^m = 10$. The same applies to the choice of t in $e^{(-\frac{\Delta f}{\log iteration})}$. For θ^{tr}, the best value between $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$ is 0.8.
As for termination, we stop the search as soon as consecutive $\frac{N}{M}$ unsuccessful iterations being observed.

The instances are generated based on the reality, however as mentioned earlier there is still a minimal level of perturbation due to some confidentialities. A part of this perturbation tries to linearly scale the time unit. That means, the time unit we employ here is not any of second, minute or hour, rather a linearly scaled one.

There are in general 16 drop-off/pick-up locations. The terminal yard is composed of 12 stacks —6 import and 6 export. For the sake of simplicity and avoiding excessive computational efforts, we have considered that every stack has only one drop-off/pick-up point along it. There is an area where the empties are stacked and there are three quay cranes which can be potentially used[2].

[2]instances are publicly available via the corresponding author

All the numerical results has been carried out on an Intel(R) Core(TM)2 Duo CPU 2.93Ghz with 4.00 GB RAM.

In the following we first examine the effect of the tightening inequality and afterwards report the numerical experiment on instances of the problem.

5.1. Inequalities

In Table 1 we report the computational experiment carried out before and after adding the valid inequalities (25). The **bold** faced content represent the instances for which the added cut resulted in improve in performance whether CPLEX status or time-wise. The *italic* cases are those instances for which the added cut deteriorated performance of CPLEX in a way that optimality has been lost or the same solution quality has been obtained in longer computational time.
Instance names are written in format of mX_nY where X is the number of machines and Y is the number of tasks. The instance sizes are only those which can be tackled by CPLEX on our machine.
A time limit of 1200 seconds and a maximum number of branch-and-bound node 500,000 has been set for this experiment. In the Table 1, `Optimal` stands for optimality, `NodeLimFeas` indicates that the solver terminated with node limit but a feasible solution has been found, `NodeLimInfeas` states that the node limit has been reached but no feasible solution detected and `AbortTimeLim` indicates that a time limit is reached without any feasible solution being found.

For instances such as $m2\_n8$ the number of branch-and-bound nodes has been reduced by half million nodes. In fact, whenever an improvement has been observed the reduction in the tree size is impressive (by some hundred thousands nodes). There are of course cases where no improvement occurs or even the solver performance deteriorates. Instances of that are, for example, $m4\_n12$ and $m4\_n16$.

On larger instances, namely 60, 70, 80, 100, while CPLEX is unable to find any feasible solution, employing our valid inequality (25) helps CPLEX to find feasible solutions with good qualities. More precisely, the gap is always below 10 percent.

Table 1: Effect of valid inequalities (25) on the performance of CPLEX

	Initial Model				Model with (25)			
Instance	nbNodes	CpuTime	CplexStatus	MIPRelativeGap	nbNodes	CpuTime	CplexStatus	MIPRelativeGap
m2_n4	541	0.34	Optimal	0.00	60	**0.09**	Optimal	0.00
m2_n8	500001	118.00	NodeLimFeas	18.11	5613	2.81	**Optimal**	0.00
m2_n12	500001	204.58	NodeLimFeas	1.95	500002	356.14	NodeLimFeas	10.91
m2_n16	0	0.17	Optimal	0.00	500001	576.85	*NodeLimFeas*	23.06
m2_n20	500002	424.68	NodeLimInfeas	-	500001	602.24	NodeLimInfeas	-
m2_n24	485382	1430.86	AbortTimeLim	-	500002	1190.63	**NodeLimFeas**	17.63
m2_n28	0	0.64	Optimal	0.00	500001	1241.77	NodeLimFeas	58.92
m2_n32	379179	1427.03	AbortTimeLim	-	388520	1411.86	AbortTimeLim	-
m2_n36	277723	1406.96	AbortTimeLim	-	373301	1435.74	AbortTimeLim	-
m2_n40	379774	1385.73	AbortTimeLim	-	0	1.45	**Optimal**	0.00
m2_n50	186465	1430.03	AbortTimeLim	-	0	2.93	**Optimal**	0.00
m2_n60	500001	1865.13	NodeLimInfeas	-	500001	1836.03	**NodeLimFeas**	33.00
m2_n80	500001	1867.12	NodeLimInfeas	-	500001	2010.20	**NodeLimFeas**	24.00
m2_n100	500001	2556.10	NodeLimInfeas	-	500001	2785.94	**NodeLimFeas**	29.00
m3_n4	671	0.50	Optimal	0.00	20	**0.25**	Optimal	0.00
m3_n8	0	0.22	Optimal	0.00	0	**0.08**	Optimal	0.00
m3_n12	500001	518.22	NodeLimFeas	20.69	63367	51.39	**Optimal**	0.00
m3_n16	500001	966.22	NodeLimFeas	0.30	500002	772.81	NodeLimFeas	45.51
m3_n20	0	0.81	Optimal	0.00	0	0.76	Optimal	0.00
m3_n24	500002	1360.42	NodeLimFeas	18.08	500002	1337.47	NodeLimFeas	19.60
m3_n28	509	3.96	Optimal	0.00	500001	1177.17	*NodeLimFeas*	0.20
m3_n32	0	1.28	Optimal	0.00	0	1.40	Optimal	0.00
m3_n36	9671	42.93	Optimal	0.00	482	**9.67**	Optimal	0.00
m3_n40	0	2.17	Optimal	0.00	0	2.78	Optimal	0.00
m3_n50	112378	1412.29	AbortTimeLim	-	0	3.85	**Optimal**	0.00
m3_n60	500001	1972.34	NodeLimInfeas	-	500001		**NodeLimFeas**	16.00
m3_n80	500001	2228.63	NodeLimInfeas	-	500001		**NodeLimFeas**	24.00
m3_n100	500001	2882.55	NodeLimInfeas	-	500001		**NodeLimFeas**	12.00
m4_n4	42	0.31	Optimal	0.00	97	1.11	Optimal	0.00
m4_n8	238	0.33	Optimal	0.00	75	0.41	Optimal	0.00
m4_n12	0	0.41	Optimal	0.00	500001	698.87	*NodeLimFeas*	9.93
m4_n16	0	0.48	Optimal	0.00	500001	885.35	*NodeLimFeas*	66.08
m4_n20	0	0.78	Optimal	0.00	0	**0.70**	Optimal	0.00
m4_n24	0	1.95	Optimal	0.00	0	**1.39**	Optimal	0.00
m4_n28	0	2.56	Optimal	0.00	86060	274.09	Optimal	0.00
m4_n32	0	2.23	Optimal	0.00	3	3.76	Optimal	0.00
m4_n36	545	22.59	Optimal	0.00	2847	37.92	Optimal	0.00
m4_n40	106152	1413.60	AbortTimeLim	3.52	0	4.77	**Optimal**	0.00
m4_n50	140954	1403.62	AbortTimeLim	-	0	9.20	**Optimal**	0.00
m4_n60	500001	1800.58	NodeLimInfeas	-	500001	1777.01	**NodeLimFeas**	10.00
m4_n80	500001	2206.94	NodeLimInfeas	-	500001	1866.89	**NodeLimFeas**	35.00
m4_n100	500001	2602.40	NodeLimInfeas	-	500001	2893.39	**NodeLimFeas**	38.00
m5_n4	31	0.36	Optimal	0.00	37	**0.25**	Optimal	0.00
m5_n8	538	1.51	Optimal	0.00	61	**0.51**	Optimal	0.00
m5_n12	0	0.39	Optimal	0.00	3061	9.67	Optimal	0.00
m5_n16	21162	52.18	Optimal	0.00	500001	756.45	*NodeLimFeas*	14.76
m5_n20	12626	52.01	Optimal	0.00	0	**1.26**	Optimal	0.00
m5_n24	0	2.89	Optimal	0.00	0	**2.18**	Optimal	0.00
m5_n28	488	6.43	Optimal	0.00	482	7.77	Optimal	0.00
m5_n32	-	-	-	0.00	497	19.84	**Optimal**	0.00
m5_n36	0	6.07	Optimal	0.00	0	**4.74**	Optimal	0.00
m5_n40	0	9.03	Optimal	0.00	0	**6.13**	Optimal	0.00
m5_n50	82710	1446.27	AbortTimeLim	0.00	0	35.12	**Optimal**	0.00
m5_n60	500001	1668.49	NodeLimInfeas	-	500001	1516.95	**NodeLimFeas**	36.00
m5_n80	500001	2119.25	NodeLimInfeas	-	500001	2190.70	**NodeLimFeas**	17.00
m5_n100	500001	2675.28	NodeLimInfeas	-	500001	2410.12	**NodeLimFeas**	10.00
m6_n8	156	0.59	Optimal	0.00	18	**0.36**	Optimal	0.00
m6_n4	39	0.19	Optimal	0.00	0	**0.11**	Optimal	0.00
m6_n12	359	1.65	Optimal	0.00	0	**0.45**	Optimal	0.00
m6_n16	0	0.80	Optimal	0.00	0	1.45	Optimal	0.00
m6_n20	0	1.09	Optimal	0.00	18	2.79	Optimal	0.00
m6_n24	9689	46.27	Optimal	0.00	339680	1324.03	Optimal	0.00
m6_n28	0	3.14	Optimal	0.00	0	4.85	Optimal	0.00
m6_n32	46305	421.70	Optimal	0.00	220828	1523.63	*AbortTimeLim*	0.07
m6_n36	123198	1439.56	AbortTimeLim	-	0	7.58	**Optimal**	0.00
m6_n40	141421	1410.61	AbortTimeLim	-	0	8.97	**Optimal**	0.00
m6_n50	10633	1515.78	AbortTimeLim	-	0	23.21	**Optimal**	0.00
m6_n60	500001	1914.72	NodeLimInfeas	-	500001	1535.65	**NodeLimFeas**	22.00
m6_n80	500001	2136.14	NodeLimInfeas	-	500001	1842.02	**NodeLimFeas**	32.00
m6_n100	500001	2720.65	NodeLimInfeas	-	500001	2882.80	**NodeLimFeas**	40.00

As shown in table, as the instance size grows the cut becomes more effective in improving the quality of solution obtainer by CPLEX.

5.2. lagrangian relaxation vs. local search

As mentioned earlier, the instances are generated based on real data of the case study.
The Dublin Ferryport Terminal (DFT) is a rather small terminal and Dublin port is mostly a non-transhipment port. The terminal authority is going to deploy 2-6 IAVs inside the terminal.
In our generated instances, 2-TEU (i.e. 1-FFE) container comprise 60% of the total discharged/uploaded containers for every vessel call. As a result we know a priori that it is less likely that duplicating the number of IAVs will proportionably reduce the makespan.

we use the well-known subgradient algorithm to solve our lagrangian problem and every 10 iterations we invoke the heuristic algorithm to obtain a primal bound. The algorithm terminates when the average subgradient size drops below certain threshold depending on the size of instance or when there has not been any improvement in last 20 iterations even after adjusting the step size of subgradient. The big-M constraints are modeled by the concept of indicators in CPLEX to avoid numerical instability in the case that the big-M constraints are not removed at the preprocessing phase of CPLEX.

The variable fixing is applied procedure is invoked every 5 iterations if there has been any improvement in either bounds.

From the Table 2, the values are chosen as $N \in \{20, 30, 40, 50, 100, 150, 200, 250, 300, 350, 400\}$ and $M \in \{2, 3, 4, 5, 6\}$. The first column of table represent values of m, while the second report n. The computational time spent in LR resolution is reported in the next column which is followed by the column representing the number of subgradient iterations. We then report the computational time required for the heuristic algorithm in the next column. The following column reports the gap between the best solution of heuristic and the LR bound. Finally, the last column reports the best-found makespan.

Due to the numerical difficulties in solving MIPs in LR, the number of iterations are fairly small, except in cases where LR problem of instance looks rather easy. The computational time ranges from approx. 35 second to approx. 14,902. This time includes the time used in for heuristic search during the search. Nevertheless, the computational time of heuristic is very small when compared to the total LR iterations CPU time.

The gap between the heuristic and the LR bound are practically acceptable for this application for confirming the quality of solutions. The maximum average gap is 16.72 which is quite acceptable.

While the increase in the makespan is not linear with the number of tasks, decrease in the length of makespan is not proportional to the size of fleet of IAVs, either. This is justified by the fact that most of the discharged containers actually lead to cooperation between IAVs which in turn results in significant waiting time which is spent for IAVs waiting for each other to start a task.

Table 2: Numerical results: Lagrangian decomposition performance.

$\|M\|$	$\|N\|$	Lagrangian Relaxation		Local Search		makespan	$\|M\|$	$\|N\|$	Lagrangian Relaxation		Local Search		makespan
		Time (sec.)	# iteration	Time (sec.)	GAP(%)				Time (sec.)	# iteration	Time (sec.)	GAP(%)	
2	20	208.53	81	21.19	28.72	134.40	5	20	751.29	71	13.81	2.93	99.48
	30	441.13	36	18.58	13.86	393.71		30	234.07	47	17.22	5.96	272.42
	40	132.90	49	13.14	16.50	815.27		40	185.25	33	13.82	2.50	564.10
	50	758.25	35	48.20	8.34	2409.73		50	420.68	84	48.30	28.18	1587.21
	100	820.60	46	78.97	25.54	6384.74		100	3812.05	37	17.00	8.24	4448.04
	150	35.95	59	208.29	6.59	11680.89		150	6303.94	14	450.62	26.58	8646.29
	200	95.29	77	28.47	29.00	26687.28		200	9316.84	46	586.58	18.31	17261.11
	250	1053.25	42	321.45	27.05	64034.56		250	11517.96	21	29.96	23.09	42228.12
	300	3460.36	52	111.83	10.17	127765.75		300	11587.74	83	515.69	0.87	89414.73
	350	4676.58	82	525.20	0.75	227634.73		350	901.47	81	74.06	2.34	147232.25
	400	4988.63	53	60.46	3.01	439510.88		400	8413.12	27	126.56	11.35	325677.56
avg.					15.41			avg.					11.85
3	20	515.30	16	39.70	15.36	123.20	6	20	145.11	14	22.99	10.53	87.88
	30	879.88	19	88.01	28.97	301.85		30	716.34	20	71.37	19.81	248.81
	40	816.33	24	18.68	9.83	625.04		40	234.54	35	32.51	5.72	515.21
	50	593.26	64	79.13	6.34	2088.44		50	1015.63	67	18.36	7.60	1449.65
	100	2596.86	41	235.69	17.26	5852.68		100	1262.78	89	95.09	28.76	3929.10
	150	3837.04	99	84.63	15.58	10707.48		150	456.43	48	109.62	12.46	7896.95
	200	2087.99	97	198.90	12.27	19125.88		200	2706.71	64	124.87	10.64	15247.31
	250	2135.31	46	506.60	23.82	52294.89		250	953.77	74	72.62	15.74	38568.35
	300	1911.92	89	528.74	13.15	110730.31		300	7798.86	95	407.25	29.48	81665.45
	350	5167.25	46	475.29	27.63	163138.22		350	3987.62	36	658.41	12.00	130055.15
	400	10404.03	41	1124.90	13.74	380909.43		400	7273.53	41	881.28	14.02	287681.85
avg.					16.72		avg.					15.16	
4	20	98.01	53	32.01	0.09	104.72							
	30	170.59	24	34.22	8.79	286.76							
	40	1263.83	35	138.95	7.94	593.79							
	50	249.93	40	150.46	25.30	1670.75							
	100	2215.72	41	238.49	11.34	4682.14							
	150	3515.87	42	165.68	9.55	9101.36							
	200	2766.83	60	186.66	9.84	18169.59							
	250	984.18	69	54.35	15.73	49680.15							
	300	9476.48	67	192.11	1.39	94120.77							
	350	7519.82	36	836.61	25.16	154981.31							
	400	14902.79	46	1184.40	6.35	361863.96							
avg.					11.04								

5.3. *Comparison with the current practice of port*

The current practice of port is comprised of a fleet of man-driven shunters each of which will transport a 1-TEU or 1-FFE containers. A precise performance comparison between current practice and the IAV-based logistics is not very straightforward as a fair comparison is rather difficult. That is, comparing an output of a deterministic model with the real practice including the stochasticity and failure etc might not be fair.

However, our discrete event simulation of the terminal by collecting the parameters from the field ensures an up to 11 percent reduction in the size of makespan.

6. Summary, conclusion and outlook to future works

An effective use of the emerging technologies is realized through optimization and simulation of environment over which they are operating. IAVs developed in the framework if InTraDE are expected to be deployed in some of the smaller container terminal in North-West Europe. The autonomy and intelligence of IAVs is promoting their applications in different environment, container terminals as the most important one.

We corrected the infeasibility and extended the model in Ng et al (2007) to accommodate cases where a 1-FFE container needs to be carried by two IAVs. Once IAV arrives at a crane to pick-up a container of more than 1-TEU size, it has to wait for another partner IAV to arrive and pair and do the pick-up jointly. We then developed a lagrangian decomposition to solve instance of a case-study of Dublin. The numerical results confirm the efficiency of our method and quality of the solutions.

The IAVs work in three conditions: 1) fully-functional, 2) degraded and 3) faulty. In the fully functional case, the speed is the desired speed and every thing is in order. In the degraded condition, perhaps part of the system (system comprised of 4 independent wheels and several independent sensors) has been failed but still the system is able to complete the jobs with less performance and in the later case IAVs have to stop operations when it is faulty. In the future, we will take into account this features and extend the model to accommodate this case. Of course studies on improving we the mathematical model is of major importance and effective solution approaches deserve particular attentions.

7. Acknowledgement

This work has been supported by the European Commission in the framework of Intelligent Transportation for Dynamic Environment (InTraDe).

8. Bibliography

Bahiense L, Maculan N, Sagastizábal CA (2002) The volume algorithm revisited: relation with bundle methods. Math Program 94(1):41–69
Barahona F, Anbil R (2000) The volume algorithm: producing primal solutions with a subgradient method. Mathematical Programming 87(3):385–399
Bish E (2003) A multiple-crane-constrained scheduling problem in a container terminal. European Journal of Operational Research 144(1):83–107
Bish E, Leong T, Li C, Ng J, Simchi-Levi D (2001) Analysis of a new vehicle scheduling and location problem. Naval Research Logistics (NRL) 48(5):363–385
Bish E, Chen F, Leong Y, Nelson B, Ng J, Simchi-Levi D (2005) Dispatching vehicles in a mega container terminal. OR Spectrum 27(4):491–506
Cao J, Lee D, Chen J, Shi Q (2010) The integrated yard truck and yard crane scheduling problem: Benders' decomposition-based methods. Transportation Research Part E: Logistics and Transportation Review 46(3):344–353
Chen L, Bostel N, Dejax P, Cai J, Xi L (2007) A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal. European Journal of Operational Research 181(1):40–58
Fisher ML (1981) The lagrangian relaxation method for solving integer programming problems. Management Science 27(1):1–18
Fisher ML (2004) The lagrangian relaxation method for solving integer programming problems. Management Science 50(12):1861–1871
Grunow M, Günther H, Lehmann M (2004) Dispatching multi-load agvs in highly automated seaport container terminals. OR Spectrum 26(2):211–235
Grunow M, Günther H, Lehmann M (2007) Strategies for dispatching agvs at automated seaport container terminals. Container Terminals and Cargo Systems pp 155–178
Guignard M (2003) Lagrangian relaxation. TOP 11(2):151–228
Hartmann S (2004) A general framework for scheduling equipment and manpower at container terminals. OR Spectrum 26(1):51–74
Held M, Karp R (1971) The traveling-salesman problem and minimum spanning trees: Part II. Mathematical Programming 1(1):6–25

Kim K, Bae J (1999) A dispatching method for automated guided vehicles to minimize delays of containership operations. International Journal of Management Science 5(1):1–25

Kim K, Bae J (2004) A look-ahead dispatching method for automated guided vehicles in automated port container terminals. Transportation science 38(2):224

Larsson T, Patriksson M, Strömberg A (1999) Ergodic, primal convergence in dual subgradient schemes for convex programming. Mathematical Programming 86(2):283–312

Lee L, Chew E, Tan K, Wang Y (2010) Vehicle dispatching algorithms for container transshipment hubs. OR Spectrum 32(3):663–685

Lemarechal C (1975) An extension of Davidon methods to non differentiable problems. Nondifferentiable Optimization pp 95–109

Meersmans P, Wagelmans A (2001a) Dynamic scheduling of handling equipment at automated container terminals. ERIM Report Series Reference No ERS-2001-69-LIS

Meersmans P, Wagelmans A (2001b) Effective algorithms for integrated scheduling of handling equipment at automated container terminals. Erasmus Research Institute of Management, Erasmus Universiteit

Narasimhan A, Palekar US (2002) Analysis and algorithms for the transtainer routing problem in container port operations. Transportation Science 36(1):63–78

Ng W, Mak K, Zhang Y (2007) Scheduling trucks in container terminals using a genetic algorithm. Engineering Optimization 39(1):33–47

Nguyen V, Kim K (2009) A dispatching method for automated lifting vehicles in automated port container terminals. Computers & Industrial Engineering 56(3):1002–1020