Sequencing Heuristics for Storing and Retrieving Unit Loads in 3D Compact Automated Warehousing Systems

Yugang Yu and René B.M. De Koster

ERIM REPORT SERIES *RESEARCH IN MANAGEMENT*	
ERIM Report Series reference number	ERS-2011-003-LIS
Publication	February 2011
Number of pages	37
Persistent paper URL	http://hdl.handle.net/1765/22722
Email address corresponding author	yyugang@rsm.nl
Address	Erasmus Research Institute of Management (ERIM) RSM Erasmus University / Erasmus School of Economics Erasmus Universiteit Rotterdam P.O.Box 1738 3000 DR Rotterdam, The Netherlands Phone: + 31 10 408 1182 Fax: + 31 10 408 9640 Email: info@erim.eur.nl Internet: www.erim.eur.nl

ABSTRACT AND KEYWORDS

Abstract	Sequencing unit load retrieval requests has been studied extensively in literature for conventional single-deep automated warehousing systems. A proper sequence can greatly reduce the makespan when carrying out a group of such requests. Although the sequencing problem is NP-hard some very good heuristics exist. Surprisingly the problem has not yet been investigated for compact (multi-deep) storage systems, which have greatly increased in popularity the last decade. This paper studies how to sequence a group (or block) of storage and retrieval requests in a multi-deep automated storage system with the objective to minimize the makespan. We adapt well-known sequencing heuristics for the multi-deep system, and propose and evaluate a new heuristic: percentage priority to retrievals with shortest leg (PPR-SL). Our results show the PPR-SL heuristic consistently outperforms all the other heuristics. Generally, it can outperform the benchmark first-come first-served (FCFS) heuristic by 20-70%. The nearest neighbor (NN) heuristic that performs very well in conventional single-deep storage systems, appears to perform poorly in the multi-deep system; even worse than FCFS. In addition, based on FCFS and PPR-SL, we find robust rack dimensions yielding a short makespan, regardless of the number of storage and retrieval requests.
Free Keywords	logistics, TSP, warehouse, compact storage, sequencing, AS/RS
Availability	The ERIM Report Series is distributed through the following platforms: Academic Repository at Erasmus University (DEAR), DEAR ERIM Series Portal Social Science Research Network (SSRN), SSRN ERIM Series Webpage Research Papers in Economics (REPEC), REPEC ERIM Series Webpage
Classifications	The electronic versions of the papers in the ERIM report Series contain bibliographic metadata by the following classification systems: Library of Congress Classification, (LCC) LCC Webpage Journal of Economic Literature, (JEL), JEL Webpage ACM Computing Classification System CCS Webpage Inspec Classification scheme (ICS), ICS Webpage

Sequencing heuristics for storing and retrieving unit loads in 3D compact automated warehousing systems

Yugang Yu
Rotterdam School of Management, Erasmus University, the Netherlands, yyugang@rsm.nl

School of Management, University of Science and Technology of China, Hefei, China

René B.M. De Koster
Rotterdam School of Management, Erasmus University, the Netherlands, rkoster@rsm.nl

Sequencing unit load retrieval requests has been studied extensively in literature for conventional single-deep automated warehousing systems. A proper sequence can greatly reduce the makespan when carrying out a group of such requests. Although the sequencing problem is NP-hard some very good heuristics exist. Surprisingly the problem has not yet been investigated for compact (multi-deep) storage systems, which have greatly increased in popularity the last decade. This paper studies how to sequence a group (or block) of storage and retrieval requests in a multi-deep automated storage system with the objective to minimize the makespan**.** We adapt well-known sequencing heuristics for the multi-deep system, and propose and evaluate a new heuristic: percentage priority to retrievals with shortest leg (PPR-SL). Our results show the PPR-SL heuristic consistently outperforms all the other heuristics. Generally, it can outperform the benchmark first-come first-served (FCFS) heuristic by 20-70%. The nearest neighbor (NN) heuristic that performs very well in conventional single-deep storage systems, appears to perform poorly in the multi-deep system; even worse than FCFS. In addition, based on FCFS and PPR-SL, we find robust rack dimensions yielding a short makespan, regardless of the number of storage and retrieval requests.

Key words: logistics; TSP, warehouse; compact storage; sequencing; AS/RS

1. Introduction

Warehouses increasingly face requirements of shortening order picking time and simultaneously saving product storage space. Automated Storage and Retrieval Systems (AS/RS) have been introduced by many companies to replace conventional manual warehouses as they can significantly save order retrieval (picking) time. Lee and Schaefer (1996) indicate that 70% of manual retrieval time can be saved by automating the product storage/retrieval (**S/R**) process. Products are stored on standardized unit loads (i.e., standard containers like totes, pallets, and cartons). A distinction can be made between single-deep and multi-deep systems. In a single-deep system (also called a 2D system), unit loads are stored single deep in the rack. Aisle-bound automated S/R machines store and retrieve the unit loads. As S/R machines require an aisle between every two racks, much floor space is wasted.

To improve order picking efficiency and to save floor space, multi-deep (**3D**) storage systems (also called *compact* or *super high-density* storage systems) have been introduced on the market (Retrotech 2006; Westfalia 2006). In one common type of 3D compact AS/RS (called a 3D system hereafter), each unit load can be accessed individually by a combination of an automated S/R machine and multiple, independently operating automated depth movement mechanisms (**DMM**) (Yu and De Koster, 2009b). Compared with a 2D system, such a 3D system has two main advantages:

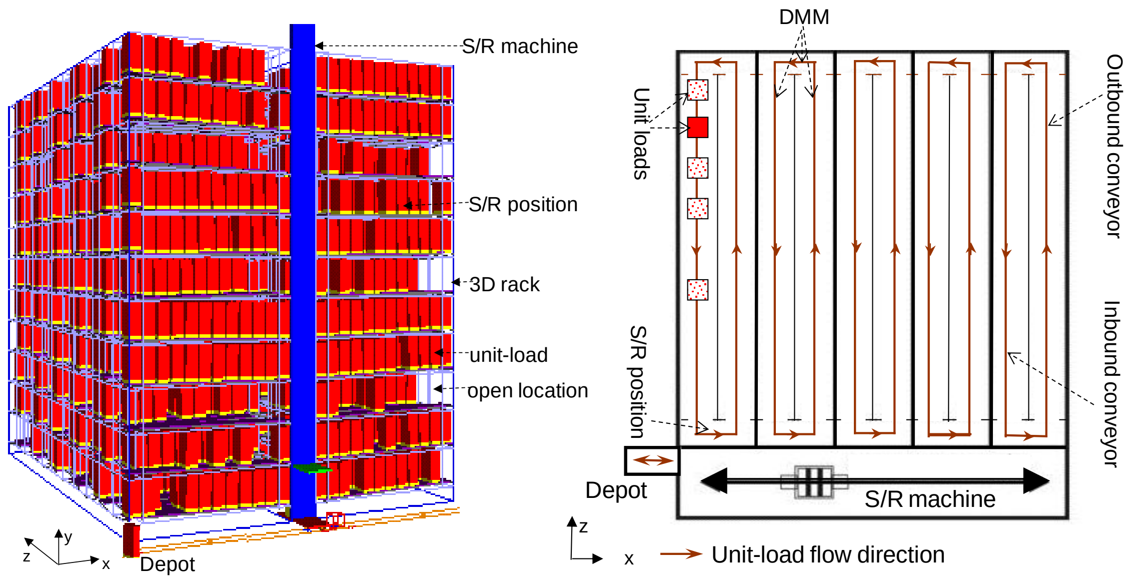
- The load throughput time is much shorter due to prepositioning of unit loads, the compactness, and full automation of the system. DMMs can operate independently and simultaneously to preposition open locations (i.e., empty locations) and locations with retrieval loads while the S/R machine is working.
- Much warehouse space can be saved because most aisles are eliminated by storing unit loads multi-deep. Viscon (2008) argues that warehouse space utilization can be increased up to 98%, which is far beyond less-than-50% utilizations in 2D systems (Piasecki, 2003).

Compact 3D systems have become increasingly popular for storing products (Hu et al., 2005; Van den Berg and Gademann, 2000). An example is the system of Miele in Germany, where a combination of S/R machines and shuttles (a kind of DMMs) store and retrieve individual palletized white goods (like washing machines and dish washers), and automatically sequence them for loading trains and trailers. More examples have been described by Graves et al. (2002), and can be found at websites of many system suppliers (e.g., Retrotech, 2006; Westfalia, 2006). To save storage space, realized 3D systems are often quite deep; 10 unit loads or more is not an exception.

The 3D system studied in this paper is sketched in Figure 1(a) and has been studied by other literature (De Koster et al., 2008; Yu and De Koster, 2009a; Yu and De Koster, 2009b) under single command modes. Similar to a 2D system, the 3D system consists of a storage rack, a depot (or I/O point), and an S/R machine (an automated crane). The S/R machine can drive and lift simultaneously and thus takes care of the movements in the *x* and *y* directions independently. Different from a 2D system, the rack is three-dimensional (multi-deep). Each level of the 3D rack is equipped with multiple conveyor-based DMMs as shown in Figure 1(b). Each DMM consists of a pair of powered conveyors and an S/R position. The paired conveyors rotate unit loads anti-clockwise, and are responsible for the movements of unit loads in the *z* direction.

The system can operate in two modes:

Single-command Cycle (SC) mode: Either a storage or retrieval operation is carried out. For a retrieval operation, the S/R machine departs from the depot, and moves to a target DMM. Meanwhile the target DMM rotates the requested unit load to its S/R position. For example, the bold unit load on the leftmost DMM in figure 1(b) is the unit load to be retrieved next. The S/R machine picks up the unit load at the S/R position and brings it to the depot. For a storage operation, the process is similar.



(a) Overall sketch

(b) Top-view sketch

Figure 1: The Sketch of the Studied 3D Compact AS/RS

Dual-command Cycle (DC) mode: A storage operation and a retrieval operation are combined. The S/R machine picks up a unit load from the depot to store it at the selected open location, which is moved simultaneously to its S/R position. After storage, the S/R machine moves empty to the S/R position of the selected retrieval to retrieve the unit load when it is available at its S/R position. Next, the S/R machine returns to the depot to drop off the unit load.

In operation, the DC mode in a 2D system can bring approximately 30% travel time reduction per storage or retrieval request compared with the SC mode (Graves et al., 1977). A similar improvement also exists in the 3D system because the S/R machine travels directly from a storage DMM to a retrieval DMM, while in the SC mode the machine first returns to the depot and only then moves to the retrieval DMM. Moreover, the DC mode is common in practice because many of these systems work in conjunction with order pick stations. To pick units of an item from a unit load stored in the rack, the unit load has to be retrieved and brought to the depot for picking items and then has to be restored back in the rack. In such a system, each retrieval is paired with the unit load of a previous pick which now has to be stored. In this paper, we mainly study the DC mode and discuss an extension to the SC mode in Subsection 6.4.

The performance of both 2D and 3D systems highly depends on the sequence in which multiple dual command cycles are carried out. Storage and retrieval requests are commonly processed in FCFS sequence. For every single storage, when multiple open locations are available, usually the open location closest to the depot is selected. The FCFS sequencing heuristic is reasonable for storages since most AS/RSs are fed by accumulating conveyors for input (Han et al., 1987). On such a conveyor there is generally no possibility for changing the sequence of unit loads to be stored (e.g. Van den Berg and Gademann, 1999). However, for retrievals, FCFS is less compelling in the sense that retrieval requests are only messages in control computers which can be re-sequenced (Han et al., 1987; Lee and Schaefer, 1996). In 3D systems, we have more opportunity to make smart selections for the load to be retrieved and the storage location as all DMMs can simultaneously rotate to bring the desired location to its S/R position. This paper focuses on this sequencing problem in 3D systems: we have to pair open locations and retrievals and sequence them, given a number of double cycles to be carried out, and taking into account intermediate prepositioning of the loads. The objective is to minimize the makespan.

The sequencing problem in 2D systems is NP-complete (Han et al., 1987). Unfortunately, the problem becomes even more difficult to solve in 3D systems, because:

- The number of movement elements in a 3D system is several dozens of times more than in a 2D system. In a 3D system, generally there are more than 50 DMMs and one S/R machine. However, there is only one S/R machine in a 2D system.
- All DMMs can work both independently of and in cooperation with each other for prepositioning multiple open locations and retrieval locations.
- The distance between every two locations on different DMMs changes constantly and is sequence dependent because of the prepositioning. A DC time of a storage-retrieval pair therefore highly depends on all previous DCs.

In practice, the computation time for sequencing is limited to seconds because the system has to update solutions to the sequencing problem frequently as blocks of S/R requests (a number of dual command cycles) arrive continuously. Therefore, some simple heuristics like FCFS are applied widely. Our contribution is to develop heuristics for the sequencing problem in 3D systems, given a block of S/R requests to be performed, and evaluate their performance compared to FCFS.

In order to do this, we first derive the makespan of a block of S/R requests and then propose a mathematical model for the sequencing problem. After that, we collect all well-known sequencing heuristics: FCFS, nearest-neighbor (**NN**), shortest-leg (**SL**), shortest dual command cycle (**SDC**), $1+e$ optimization, and the transportation problem (**TP**) known in 2D systems, and adapt the feasible ones (i.e. FCFS, NN, SL, SDC) for 3D systems. We propose a new heuristic: the percentage priority to retrievals with the shortest leg (**PPR-SL**). All the heuristics are then tested by simulation to evaluate their performance in the 3D system. Finally, we develop recommendations for determining the parameter in PPR-SL, for choosing the proper block size of storage and retrievals requests, and for determining the dimensions of the 3D rack for a given total storage capacity.

The remainder of the paper is organized as follows. Section 2 reviews previous studies on sequencing heuristics and picks feasible ones for 3D systems. Section 3 defines the research problem and formulates a general model. Section 4 introduces all studied sequence heuristics. Section 5 evaluates all selected heuristics by simulation and analyzes their performance with different system parameters. Section 6 develops recommendations for applying the heuristics in practice and investigates dynamic use of the heuristics. Section 7 concludes the paper by giving some managerial insights for using the heuristics.

2. Literature review

The retrieval sequencing problem has been widely studied in 2D AS/RSs with aisle-bound S/R machines or with carousels.

A carousel AS/RS has a rotary rack, like a DMM, to move items to the depot. Different from our 3D system, conventional carousels do not use aisle-bound S/R machines to move unit loads to and from the depot. Therefore, storages and retrievals are rarely combined in dual-command cycles. Related research is mainly about how to sequence a block of retrieval requests (carried out in single cycles) by minimizing average or total retrieval time (Litvak and Vlasiou, 2008; Wen and Chang, 1988). Because selecting open locations for storage requests is avoided, the optimal solution can be found (Bartholdi and Platzman, 1986; Van den Berg, 1996) or estimated (Litvak, 2006) for a single carousel system. Sequencing retrievals with multiple carousels is complex and has hardly been addressed (Litvak and Vlasiou, 2008). All in all, it is not feasible to use heuristics developed for carousel systems, such as monomaniacal and shorter-direction heuristics for bi-directional rotary racks (Bartholdi and Platzman, 1986), for our problem. One heuristic, the nearest item heuristic (Bartholdi and Platzman, 1986; Litvak and Vlasiou, 2008) can be translated for our research and will be included in the nearest neighbor heuristic below.

The sequencing problem considering both the selection of open locations and sequencing retrievals has been studied for an AS/RS with an aisle-bound S/R machine. It is a warehouse-specific Traveling Salesman Problem (TSP) (Han et al., 1987) where the distance between every two requested unit loads is represented by the Chebyshev metric. The problem is even more complex when multiple candidate open locations are available for storage space selection, which much increases the solution space. Han et al.

(1987) show that the sequencing problem with one open location in 2D systems is already NP-hard. Due to this complexity and limitations on computation time, most solution methods are heuristics (see, Han et al., 1987; Lee and Schaefer, 1996, 1997; Mahajan et al., 1998), evaluated by simulation. Two papers aim to solve the problem by finding a near optimal solution (Lee and Schaefer, 1996) or the optimal solution in a special case (Van den Berg and Gademann, 1999). Well-known sequencing heuristics include:

FCFS (first-come-first-served). FCFS is the policy most frequently used in practice. Both storage and retrieval requests are carried out in FCFS sequence. For every storage location, the open location closest to the depot (closest open location, COL) is selected. FCFS is considered as a benchmark by many researchers to evaluate heuristics for storing and retrieving unit loads (Dooly and Lee, 2008; Han et al., 1987; Mahajan et al., 1998; Van den Berg and Gademann, 1999) or for selecting loads in other, similar, operational settings (De Koster et al., 2004; Dekker et al., 2004).

NN (nearest-neighbor). Han et al. (1987) propose the NN heuristic to reduce the travel-between time. Storage and retrieval requests are sequenced by matching a storage open location and a retrieval location with the minimum travel-between time. They show that, compared with FCFS, NN can bring an increase in throughput of 10-15%. NN is one of the most well known heuristics, and cited by many researchers (e.g., Dooly and Lee, 2008; Gu et al., 2007; Van den Berg and Gademann, 2000).

SL (shortest-leg). Han et al. (1987) discuss the "no cost zone" for the Chebyshev metric, i.e., the area in the rack with open locations that may be visited for a storage without extra travel time according to the Chebyshev metric, while the S/R machine is traveling from the depot to the retrieval location. Based on this, they propose the SL heuristic by selecting open locations and retrieval locations which yield the shortest storage plus travel-between time. If there is an open location in the "no cost zone" of the retrieval location, then it will be selected as storage point. It is shown that this heuristic has a performance about similar to NN (Han et al., 1987; Lee and Schaefer, 1996).

SDC (shortest dual command cycle). Assuming *n* storage and retrieval requests are waiting, Lee and Schaefer (1996) propose the SDC heuristic that greedily selects a storage open location and retrieval location pair using the shortest DC time at every step. They test the heuristic, compare it with NN and SL, and show that SDC is consistently better than NN and SL in their experiments. SDC, also called the "minimum-perimeter" heuristic, is applied by Keserla and Peters (1994) in an AS/RS where the S/R machine has two shuttles.

1+ε optimization and TP (transportation problem). Instead of using the above simple greedy-style heuristics, Lee and Schaefer (1996) first formulate the sequencing problem as a variant of the linear assignment problem and then present an ε -optimum algorithm for solving it approximately where ε indicates a tolerance gap between the objective value of a solution and a problem lower bound. Their algorithm can solve moderate-size problems within a reasonable computation time and tolerance gap ε. In their experiments, the number of open locations is always less than 50. They compare their algorithm with FCFS, NN, SDC, and SL, and demonstrate that NN, SDC, and SL can provide very good near optimal solutions with gaps mostly less than 5% from the optimum. Assuming that the storage location of each storage request is fixed beforehand, Van den Berg and Gademann (1999) show that, in this special case, the sequencing problem can be solved optimally in polynomial time by formulating it as a Transportation problem (**TP**). The optimal solution of the sequencing problem corresponds to an optimal solution of the TP. Unfortunately both methods are not applicable to our research. They need cycle travel times for all possible DCs as input parameters for their models. However, in 3D systems, because open or retrieval location positions are changing with time due to prepositioning, the cycle travel times for DCs are changing with time as well and cannot be pre-determined.

The problem of sequencing retrievals in compact storage systems has not been addressed in previous research. Yu and De Koster (2009a; 2009b) focus on handling individual requests which are instantaneously generated and handled one by one, instead of block by block.

This paper fills this gap as the first paper to study the sequencing problem in 3D compact systems. Some heuristics developed in literature, such as FCFS, NN, SDC, and SL can be adapted to our sequencing problem.

3. Problem Description and General model

3.1 Problem Description

We study the 3D compact AS/RS sketched in Figure 1(a). Unit loads for storage arrive at the depot and wait at an accumulating conveyor until the S/R machine transports these to selected open locations in the rack. Retrieved unit loads are dropped off by the S/R machine at the depot on an outgoing conveyor for further transport. Following the assumptions of related literature (e.g., Graves et al., 1977; 1987; Lee and Schaefer, 1996; Van den Berg and Gademann, 1999), we assume that:

- A block of storage and retrieval requests needing sequencing is instantaneously given at the start of the planning period. All open locations and retrieval locations are known at this moment.
- The total storage capacity, the speed of the depth movement mechanisms, and the S/R machine's speeds in the horizontal and vertical direction are known and constant. The S/R machine can drive and lift simultaneously.
- The objective function is to minimize the average cycle time. It equals the makespan (i.e., the total cycle time) of the given block of storage and retrieval requests divided by the total number of cycles. The (constant) pick-up and deposit time of a load is omitted.
- The S/R machine only operates in dual command cycle (DC) mode. The extension to single command modes is addressed in Subsection 6.4.
- The depot is located at the lower left-hand corner of the rack (see Figure 1(a)). The S/R machine dwells at the depot at the start of the block.
- The storages are performed in FCFS sequence; the sequence of storage unit loads cannot be changed.

Unlike most literature for 2D systems, we do not treat the 3D rack as continuous. This is necessary to keep track of individual DMMs and the exact positions of storage or retrieval locations on DMMs during the sequencing process. DMMs can simultaneously begin to preposition the required open locations and retrieval locations to their S/R positions once the S/R machine starts to perform the first DC.

3.2 Notations

According to Figure 1, we define the sizes of the rack in the *x*, *y*, *z* directions by its length (*L*), its height (*H*), and the perimeter of a DMM with two paired conveyors (*2S*), respectively. The speed of the DMMs and the S/R machine's speeds in the horizontal and vertical directions are denoted by s_c , s_h , and s_v respectively. The notations are:

Sets and indices

p	Sequence index of dual command cycles, $p = 1,...,\| R_1 \|$.
R_p	Set of retrieval requests at the beginning of the p^{th} DC. R_1 is the set of initial retrieval requests.
S_p	Set of open locations at the beginning of the p^{th} DC. S_1 is the set of initial open locations.
r	Index of a retrieval request with r Î R_p at the beginning of the p^{th} DC.
s	Index of a storage open location with s Î S_p at the beginning of the p^{th} DC.

Parameters

V_M , V_T , V_N	Rack volume in cubic meters, in cubic seconds, and in the number of unit loads respectively.
t_c	$= 2S / s_c$; length (in time) of two conveyors in a pair.
t_h	$= L / s_h$; length (in time) of the rack.
t_v	$= H / s_v$; height (in time) of the rack.
X_k , Y_k	Time needed for the S/R machine to travel from the depot to the S/R position of open location (or retrieval) location *k* in horizontal and vertical directions respectively.

Functions

B_{sr}^{p}	Joint travel-between time between open location *s* just loaded, and retrieval *r*

just picked up in the p^{th} DC by the S/R machine and the DMMs.

DC_{sr}^{p}	DC time of the p^{th} DC for a (*s, r)* pair.
TCT	The total cycle travel time (or makespan) for the sequencing problem.
U_r	Return time of the S/R machine from the S/R position of retrieval *r* to the depot.
W_s^p	Time needed for the system (S/R machine and DMMs) between picking up a unit load, and storing it at open location *s* in the p^{th} DC.
Z_k^p	*z* coordinate (in time) of open location or retrieval *k* at the start of the p^{th} DC, i.e., the time needed for a DMM to rotate *k* to its S/R position in the p^{th} DC.

Decision variables

l_{sr}^{p}	$l_{sr}^{p}=1$ represents that open location *s* is paired with retrieval *r* to form the p^{th} DC, otherwise $l_{sr}^{p}=0$.
$v_i(t)$	$v_i(t)=1$ if open or retrieval location *i* is being repositioned at time t, otherwise $v_i(t)=0$. $v_i(t)$ represents an infinite number of variables as for each value of *t*, $v_i(t)$ is a decision variable.

3.3 Makespan and sequencing model

The objective of the problem is to minimize the total cycle makespan taken by the 3D system to perform $|R_1|$ dual command cycles. Decisions to be sought are a sequence of $|R_1|$ DCs which involves: a) sequencing $|R_1|$ retrieval requests and b) assigning open locations to $|R_1|$ storage requests that are paired with the retrievals to form $|R_1|$ DCs. In order to do this, $|S_1|$ should be larger than or equal one. The locations of open locations and the *z* coordinates of the S/R requests keep changing while the system performs DCs. DC_{sr}^{p} $(p = 1,\dots,|R_1|)$ thus depends on its previous $p-1$ DCs. With a given sequence of the $p-1$ DCs preceding the p^{th} DC, we have:

$$DC_{sr}^{p} = W_s^p + B_{sr}^{p} + U_r\,, \tag{1}$$

which consists of three components:

- ***Storage travel time (W_s^p):*** The time needed to store a unit load involving two parallel processes: a) the S/R machine moves a unit load from the depot to the S/R position of open location *s* and b) the DMM circulates open location *s* (a position which depends on the previous *p*-1 DCs) to the same S/R position in the p^{th} DC. The movements in horizontal and vertical directions of the S/R machine, and in depth direction of the DMMs are independent of each other. Therefore W_s^p equals:

$$W_s^p = \max\{X_s, Y_s, Z_s^p\}, \qquad (2)$$

where Z_s^p is a function of Z_s^{p-1}, DC_{sr}^{p-1}, and t_c for $p \geq 2$.

- ***Travel-between time (B_{sr}^p):*** the travel time needed of two parallel movement processes between open location *s* just loaded and retrieval *r* available for pick up at its S/R position in the p^{th} DC. The two parallel movements are: a) the S/R machine moves from the S/R position of open location *s* to the S/R position of retrieval *r*, and b) the DMM circulates retrieval *r* to its S/R position. We have

$$B_{sr}^p = \max\{|X_r - X_s|, |Y_r - Y_s|, ZB_r^{p,t_s}\}. \qquad (3)$$

where ZB_r^{p,t_s} is the *z* coordinate of *r* at the time open location *s* has just been loaded in the p^{th} DC.

If *s* and *r* are on two different DMMs,

$$ZB_r^{p,t_s} = \begin{cases} Z_r^p - W_s^p & \text{if } Z_r^p - W_s^p \geq 0 \\ 0 & \text{otherwise} \end{cases}, \qquad (4)$$

and if *s* and *r* are on the same DMM,

$$ZB_r^{p,t_s} = \begin{cases} Z_r^p - Z_s^p & \text{if } Z_r^p - Z_s^p \geq 0 \\ Z_r^p - Z_s^p + t_c & \text{otherwise} \end{cases}. \qquad (5)$$

- ***Return time (U_r)***: the S/R machine's travel time from the S/R position of retrieval *r* to the depot.

$$U_r = \max(X_r, Y_r) \qquad (6)$$

We then obtain the makespan given by Equation (7)

$$TCT = \sum_{p=1}^{|R_1|} \sum_{s \in S_1 \cup R_1} \sum_{r \in R_1} \lambda_{sr}^p DC_{sr}^p(Z_s^p, Z_r^p) \qquad (7)$$

The sequencing-problem model (denoted by SP) can now be formulated. Details can be found in Appendix 1.

Model SP is still in a conceptual form and quite impossible to solve as $v_i(t)$ is a decision function. In the objective function DC_{sr}^{p} (corresponding to the distance between two locations in the TSP) is not constant, but a function of Z_s^p and Z_r^p. Z_s^p and Z_r^p are functions of decision variables l_{sr}^{p} and $v_i(t)$. That is to say, DC_{sr}^{p} not only depends on the selected pair (s ,r) for the p^{th} DC but also depends on the sequence of the previous p-1 DCs and the way they have been prepositioned.

Model SP is a nonlinear mixed integer programming as the objective function and constraints (18g)-(18i) are nonlinear and l_{sr}^{p} and $v_i(t)$ are binary. Theorem 1 proves a simplified version of Model SP (neglecting $v_i(t)$ for prepositioning) is strongly NP-Hard.

Theorem 1. The problem defined by Model SP is strongly NP-Hard.

Proof. See Appendix 2.

In order to ease the discussion and some comparisons hereafter, this paper assumes without loss of generality the rack volume, V_T , equals 1 using Theorem 2 below.

Theorem 2. For a given rack-shape factor $t_h : t_v : t_c$, for any given rack volume $V_T = \bar{V}_T > 0$, the makespan of the sequencing problem can be normalized to that with $V_T = 1$, and $TCT_{\bar{V}_T} / TCT_{V_T=1} = \sqrt[3]{\bar{V}_T}$.

Proof. See Appendix 3.

According to Theorem 2 and its proof, we can hereafter assume the system capacity is normalized to $V_T = 1$. For any given $\bar{V}_T$, t_h , t_v , and t_c can be normalized by setting $t_h \neg t_h / \sqrt[3]{\bar{V}_T}$, $t_v \neg t_v / \sqrt[3]{\bar{V}_T}$, $t_c \neg t_c / \sqrt[3]{\bar{V}_T}$, and $t_h t_v t_c = 1$. The average cycle transaction time of a DC also will be provided in its normalized form by using $TCT_{\bar{V}_T} / (|R_1| \sqrt[3]{\bar{V}_T})$.

4. Sequencing heuristics

The result above suggests it is unlikely to develop a polynomial-time algorithm to find an optimal solution. Efficient heuristics that provide "good" solutions to the sequencing problem become desirable. We first introduce the FCFS heuristic, as a benchmark. After that, we adapt NN, SDC and SL for 3D systems. Finally we propose a new heuristic by combining SL with a new prepositioning method. SL is selected because it appears to perform better than the other heuristics: FCFS, NN and SDC.

4.1 FCFS heuristic

Storage and retrieval requests are processed in FCFS sequence while the closest open location (COL) from S_p is selected as a storage location. Furthermore, on every DMM, the open location or retrieval closest to its S/R position is prepositioned there first for later possible usage. This prepositioning process begins when a block of storage and retrieval requests arrive. The prepositioning brings open locations and retrievals closer to their S/R positions, which potentially reduces the S/R machine waiting time at those S/R positions, and eventually can reduce the makespan. FCFS is implemented with the following steps:

Step 0 (Initialization): Give the initial open location set: S_1, the initial retrieval set: R_1, and p=1.

Step 1 (Check R_p): If $R_p \neq \emptyset$, go to the next step. Otherwise stop and output all l_{sr}^{p} and *TCT*.

Step 2 (Calculate selection criteria): Calculate W_s^p for all open locations.

Step 3 (Obtain open-location-retrieval pair (s, r)): Select the closest open location (COL) s with s=arg $\min\{W_s^p \mid s \in S_p\}$ as the storage location, and the first-come retrieval r to compose pair (s, r).

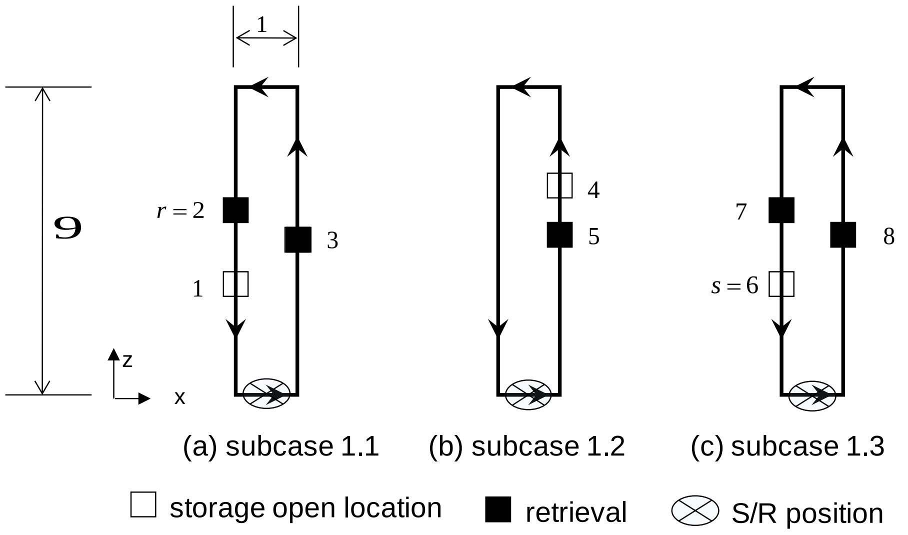
Step 4 (Perform (s, r)): Execute the selected DC of pair (s, r). Obtain the current DC time of cycle p, and the makespan of all the p DCs. Obtain l_{sr}^{p} and DC_{sr}^{p}.

Step 5 (Update sets): $S_{p+1} \leftarrow S_p + \{r\} - \{s\}$, and $R_{p+1} \leftarrow R_p - \{r\}$.

Step 6 (Update z coordinates by using the FCFS prepositioning rule): Let $p \leftarrow p+1$, update the z coordinates of every open location and retrieval $k \in S_p \cup R_p$, and go to Step 1.

Updating *z* coordinates of open location or retrieval *k*, k Î S_p È R_p for a given pair (*s*, *r*) works as follows. Let DMM_s and DMM_r denote the sets of open locations or retrievals on the DMM of *s* and *r* at the beginning of DC *p*-1, respectively. For a given open location or retrieval *i* Ï DMM_s È DMM_r , $\overline{DMM}_i$ denotes the set of open locations and retrievals on DMM *i* that is neither the DMM of *r* nor *s*. In the example of Figure 2, (*s*, *r*)=(6, 2), DMM_s ={6,7,8}, and DMM_r ={1,2,3}. $\overline{DMM}_k$ ={4,5} with *k*=4 or 5. We distinguish two cases: $DMM_s \neq DMM_r$ and $DMM_s = DMM_r$.

Case 1 (DMM_s ¹ DMM_r):



Note: $Z_1^{p-1} = Z_6^{p-1} = 4,\ Z_2^{p-1} = Z_7^{p-1} = 6,\ Z_3^{p-1} = Z_5^{p-1} = Z_8^{p-1} = 15,\ Z_4^{p-1} = 14,$ and $t_c = 20$

Figure 2: Three Subcases for DMM_s ¹ DMM_r

In this case, any open location or retrieval $k \in S_p \cup R_p$ must belong to one of the three sub-cases $k \in DMM_r$, $k \in \overline{DMM}_i$ or $k \in DMM_s$, and its *z* coordinate can be updated as follows.

Sub-case 1.1 (k Î DMM_r):

After retrieval *r* has been picked up in DC *p*-1, the retrieval location becomes an open location with S_p ¬ S_{p-1} + $\{r\}$ according to Step 5, and this location remains at its S/R position with Z_r^p =0. Therefore we have, for k Î DMM_r ,

$$Z_k^p = \begin{cases} Z_k^{p-1} - Z_r^{p-1} & if\ Z_k^{p-1} - Z_r^{p-1} \geq 0 \\ Z_k^{p-1} - Z_r^{p-1} + t_c & Otherwise \end{cases} . \tag{8}$$

In the example of Figure 2(a), r=2 in DC p-1 and the z-coordinate of this location becomes $Z_r^p = Z_r^{p-1} - Z_r^{p-1}$ =6-6=0. Correspondingly, for the other locations on this DMM, we get $Z_1^p = Z_1^{p-1} - Z_r^{p-1} + t_c$ =4-6+20=18 and Z_3^p =15-6=9.

Sub-case 1.2 ($k \in \overline{DMM_i}$):

The shortest time needed for prepositioning an open location or retrieval to its S/R position in DC p-1 on the DMM, $d_i^{p-1,\min}$, is

$$d_i^{p-1,\min} = \min\{Z_j^{p-1} \mid j \in \overline{DMM_i}\} .$$

The available time for this prepositioning is DC_{sr}^{p-1}. Therefore we have, for $k \in \overline{DMM_i}$,

$$Z_k^p = Z_k^{p-1} - \min\{d_i^{p-1,\min}, DC_{sr}^{p-1}\} . \tag{9}$$

If $\min\{d_i^{p-1,\min}, DC_{sr}^{p-1}\} = DC_{sr}^{p-1}$, no open location or retrieval can be prepositioned to the S/R position. Otherwise, $\min\{d_i^{p-1,\min}, DC_{sr}^{p-1}\} = d_i^{p-1,\min}$ and one storage open location or retrieval $k \in \overline{DMM_i}$ can be prepositioned to the S/R position.

In the example of Figure 2(b), $\overline{DMM_i} = \overline{DMM_4} = \overline{DMM_5}$ in DC p-1. If DC_{sr}^{p-1} =10, we get $\min\{d_i^{p-1,\min}, DC_{sr}^{p-1}\} = DC_{sr}^{p-1}$=10. We then have $Z_4^p = Z_4^{p-1} - 10$=14-10=4.

Sub-case 1.3 ($k \in DMM_s$):

When s has just been stored in DC p-1, the z coordinate of k, k≠s, Z_k^{temp}, is

$$Z_k^{temp} = \begin{cases} Z_k^{p-1} - Z_s^{p-1} & if\ Z_k^{p-1} - Z_s^{p-1} \geq 0 \\ Z_k^{p-1} - Z_s^{p-1} + t_c & Otherwise \end{cases} . \tag{10}$$

At this point in time, the shortest time needed for prepositioning an open location or retrieval on DMM_s to its S/R position, $d_s^{p-1,\min}$, is

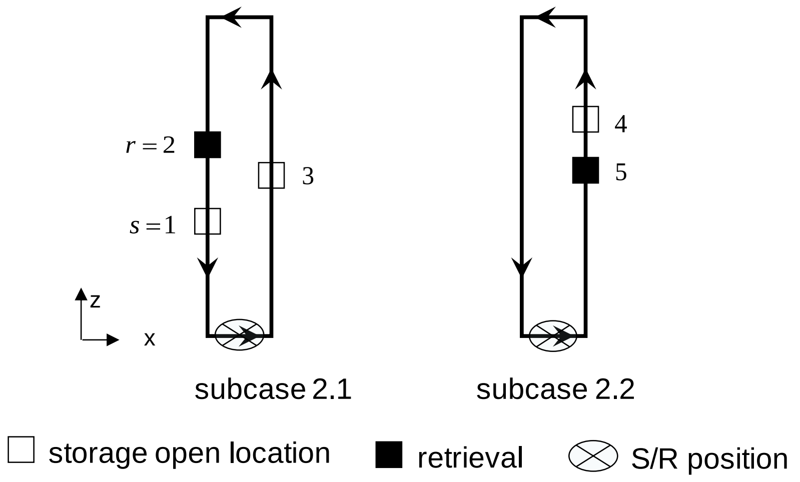
$$d_s^{p-1,\min} = \min\{Z_j^{temp} \mid j \in DMM_s, and\ j \neq s\}.$$

The available time for prepositioning in DC p-1 is $B_{sr}^{p-1} + U_r$. We therefore have, for $k \in DMM_s$,

$$Z_k^p = Z_k^{temp} - \min\{d_s^{p-1,\min}, B_{sr}^{p-1} + U_r\}. \qquad (11)$$

For the example of Figure 2(c), according to Equation (10), Z_7^{temp} =6-4=2 and Z_8^{temp} =15-4=11. Because $d_s^{p-1,\min} = \min\{2,11\}$ =2, $\min\{d_s^{p-1,\min}, B_{sr}^{p-1} + U_r\}$ =2 assuming $B_{sr}^{p-1} + U_r$ >2. We then have $Z_7^p = Z_7^{temp} - 2 = 2 - 2 = 0$ and $Z_8^p = Z_8^{temp} - 2 = 11 - 2 = 9$ according to Equation (11).

Case 2 ($DMM_s = DMM_r$*):*



Note: $Z_1^{p-1} = 4,\ Z_2^{p-1} = 6,\ Z_3^{p-1} = Z_5^{p-1} = 15,\ Z_4^{p-1} = 14,$ and $t_c = 20$

Figure 3: Two Subcases for $DMM_s = DMM_r$

In this case, any open location or retrieval $k \in S_p \cup R_p$ must belong to one of the two sub-cases $k \in DMM_r$ or $k \in \overline{DMM}_i$ for $i \notin DMM_r$, and its *z* coordinate can be updated as follows.

Sub-case 2.1 ($k \in DMM_r$ or DMM_s): Similar to sub-case 1.1 above, retrieval location *r* remains at the S/R position after the retrieval in DC *p*-1 and becomes an open location in DC *p*. The *z* coordinate of *k* can then be updated by Equation (8). For example, in Figure 3(a), $DMM_s = DMM_r = \{1,2,3\}$ and we then have $Z_2^p = 0$, $Z_1^p = 4 - 6 + 20 = 18$ and $Z_3^p = 15 - 6 = 9$.

Sub-case 2.2 ($k \in \overline{DMM_i}$): Similar to sub-case 1.2 above, the *z* coordinate of *k* is updated by Equation (9). For example, in Figure 3(b), $\overline{DMM_i} = \{4,5\}$ and we then have $Z_4^p = Z_4^{p-1} - DC_{sr}^{p-1} = 14 - 10 = 4$ and Z_5^p=15-10=5.

4.2 NN, SDC and SL heuristics

The only difference between FCFS and the heuristics NN, SDC and SL is the criterion to select an open-location-retrieval pair (*s*, *r*) for a DC. That is to say, we only need to replace steps 2 and 3 in Subsection 4.1 with corresponding criteria for NN, SDC and SL to obtain a (*s*, *r*) pair.

Nearest Neighbor (NN) heuristic:

*Step 2***:** Calculate B_{sr}^p for all possible open-location-retrieval pairs (*s*, *r*).

*Step 3***:** Select the pair (*s*, *r*) with $\min\{B_{sr}^p \mid s \in S_p, r \in R_p\}$ to perform the *p*th DC.

Shortest DC (SDC) heuristic:

*Step 2***:** Calculate $DC_{sr}^p = W_s^p + B_{sr}^p + U_r$ for all possible open-location-retrieval pairs (*s*, *r*).

*Step 3***:** Select the pair (*s*, *r*) with $\min\{DC_{sr}^p \mid s \in S_p, r \in R_p\}$ to perform the *p*th DC.

Shortest Leg (SL) heuristic:

*Step 2***:** Calculate $SL_{sr}^p = W_s^p + B_{sr}^p$ for all possible open-location-retrieval pairs (*s*, *r*).

*Step 3***:** Select the pair (*s*, *r*) with $\min\{SL_{sr}^p \mid s \in S_p, r \in R_p\}$ to perform the *p*th DC.

4.3 Priority percentage to retrievals with shortest leg (PPR-SL)

The previous heuristics assume that all open locations and retrievals have the same priority to be prepositioned to their S/R positions. However, normally there are many more open locations in the rack than retrievals in a block. If the system prepositions open locations and retrievals at the S/R positions with equal priority, retrievals therefore have a high probability to be pushed further from their S/R positions. To reduce this probability, we confer a higher priority to retrievals for prepositioning than to open locations on the same DMM if condition (12) is satisfied:

$$\frac{\text{the number of DMMs with retrievals}}{\text{the total number of DMMs with retrievals or openings}} \le a, \tag{12}$$

where a is the given priority percentage or fraction between 0 and 1. Note that SL is a special case of PPR-SL with a =0.

That is to say, when condition (12) is satisfied in DC *p* for a given a , and there are retrievals on a DMM, the closest retrieval on the DMM will be prepositioned first. Assume $\alpha = 1$ in the example of Figure 2(a), and then condition (12) is satisfied. In $\overline{DMM}_4$, retrieval 5 has a higher priority than open location 4 for being prepositioned. In DMM_s, the closest retrieval 7 will be prepositioned immediately after finishing the current storage request *s*.

The PPR-SL heuristic considers the above condition and combines it with SL. SL is chosen because it is shown in Section 5 to outperform FCFS, NN, and SDC.

Similar to SL, PPR-SL can be implemented for any given α in 6 steps. The differences between them include Step 0 for initialization, Step 6 to update the *z* coordinates, and new steps 7-9 for finding a near optimal a .

Step 6 can be split into the following sub-steps:

Sub-step 6.1 (check condition (12)): If condition (12) is satisfied, go to Sub-step 6.2. Otherwise, go to Sub-step 6.3.

Sub-step 6.2 (update *z* coordinates with higher priority to retrievals): Corresponding to the cases given in Subsection 4.1, we need to update the *z* coordinates of open location or retrieval *k*, $k \in S_p \cup R_p$.

On every DMM, if there is no retrieval on the DMM in DC *p*, go to Sub-step 6.3. Otherwise, there is at least one retrieval on the DMM and the *z* coordinate of any open location or retrieval *k* on the DMM is updated according to the two respective sub-cases ($DMM_s \neq DMM_r$ and $DMM_s = DMM_r$) as follows:

Case 1 ($DMM_s \neq DMM_r$):

S*ub-case 1.1*($k \in DMM_r$). At the time that the retrieval has just been picked up, the *z* coordinate of *k* on the DMM of *r*, Z_k^{temp} , is

$$Z_k^{temp} = \begin{cases} Z_k^{p-1} - Z_r^{p-1} & if\ Z_k^{p-1} - Z_r^{p-1} \geq 0 \\ Z_k^{p-1} - Z_r^{p-1} + t_c & Otherwise \end{cases}.$$

After that, the DMM does not stop but prepositions a retrieval to the S/R position. The shortest time to preposition a retrieval to the S/R position , $d_r^{p-1,\min}$, is

$$d_r^{p-1,\min} = \min\{Z_j^{temp} \mid j \in R_p \cap DMM_r\} .$$

The available time for the prepositioning is U_r . Therefore we have, for $k \in DMM_r$,

$$Z_k^p = \begin{cases} Z_k^{temp} - \min\{d_r^{p-1,\min}, U_r\} & if\ Z_k^{temp} - \min\{d_r^{p-1,\min}, U_r\} \geq 0 \\ Z_k^{temp} - \min\{d_r^{p-1,\min}, U_r\} + t_c & otherwise \end{cases}. \qquad (13)$$

Sub-case 1.2($k \in \overline{DMM_i}$). Because there is at least one retrieval in DMM_i ($R_p \cap \overline{DMM_i} \neq \emptyset$), the z coordinate of the retrieval closest to the S/R position on the DMM is:

$$d_i^{p-1,\min} = \min\{Z_j^{p-1} \mid j \in \overline{DMM_i} \cap R_p\} .$$

The available prepositioning time is DC_{sr}^{p-1} . Therefore we have, for $k \in DMM_i$,

$$Z_k^p = \begin{cases} Z_k^{p-1} - \min\{d_i^{p-1,\min}, DC_{sr}^{p-1}\} & if\ Z_k^{p-1} - \min\{d_i^{p-1,\min}, DC_{sr}^{p-1}\} \geq 0 \\ Z_k^{p-1} - \min\{d_i^{p-1,\min}, DC_{sr}^{p-1}\} + t_c & otherwise \end{cases}. \qquad (14)$$

Sub-case 1.3 ($k \in DMM_s$). Because $R_p \cap DMM_s \neq \emptyset$, we set $d_s^{p-1,\min} = \min\{Z_j^{temp} \mid j \in R_p \cap DMM_s\}$ where Z_j^{temp} is determined by Equation (10). Therefore we have, for $k \in DMM_s$,

$$Z_k^p = \begin{cases} Z_k^{p-1} - \min\{d_s^{p-1,\min}, B_{sr}^{p-1} + U_r\} & if\ Z_k^{p-1} - \min\{d_s^{p-1,\min}, B_{sr}^{p-1} + U_r\} \geq 0 \\ Z_k^{p-1} - \min\{d_s^{p-1,\min}, B_{sr}^{p-1} + U_r\} + t_c & otherwise \end{cases}. \qquad (15)$$

Case 2 ($DMM_s = DMM_r$): the *z* coordinate of *k* on a DMM is updated by Equations (13) and (14).

Sub-step 6.3 (Condition (12) is NOT satisfied): The *z* coordinates of open locations and retrievals are updated with equal priority by Equations (8)-(11) in Subsection 4.1.

The parameter α in Equation (12) can be optimized by adding some extra steps to the algorithm.

In Step 0, we add α =0 for initializing α . We then add three steps below:

Step 7 (update a): With a given step size (0.1 in our numerical studies), update a ($a \neg a + 0.1$ in our simulation).

Step 8 (compare and output results): If $a < 1$, compare the makespan at the current a and that of the previous a (if there is one), select and remember the result that provides the shorter makespan, and go to the next step. Otherwise, $a \geq 1$, compare all solutions and output the final solution a^* that gives the shortest makespan.

Step 9 (re-initialize DC index *p*): If $a < 1$, reset *p*=1, recall the original coordinates of all open locations and retrievals, and go to Step 1.

5. Simulations

Simulation experiments have been carried out to evaluate all the heuristics, and to compare their performances. In these experiments, the rack is randomly filled up to a given utilization. Retrieval and open locations are randomly distributed. Then all the heuristics are dynamically simulated and the makespans are recorded. We compare the performances of the heuristics: FCFS, NN, SDC, SL, and PPR-SL.

5.1 Simulation setup

The layout of the 3D AS/RS is given by Figure 1. System parameters for the base examples are given in They are based on realistic compact pallet-based storage systems according to our observations. Based on the base examples, we vary the system storage capacity and rack shape for sensitivity analyses.

All the simulations have been programmed in C++, and run on a notebook with CPU1.2GHz, and 512M of RAM.

Simulation procedures. For a given set of system parameters for a discrete rack, we use Monte Carlo simulation to generate a large number of replications. Each replication is solved by using the different heuristics discussed before. The different steps of this procedure are described below.

Step 0 (initialization): Given V_N , we have $V_M = V_N$ ´ h ´ v ´ c where h, v , and c are the width, height, and depth of a unit load (see Table 1). Using $V_T = (L / s_h) * (H / s_v) * (2S / s_c) = 2 * H * L * S / (s_h s_v s_c)$ $= 2V_M / (s_h s_v s_c)$, we obtain

$$V_T = 2V_M / (s_h s_v s_c) . \tag{16}$$

Table 1. System Parameters

Parameters		Values
Unit load size in meter (width × height × depth)	Net Gross	1.2 × 1.2× 1 1.4 × 1.5× 1.2
S/R machine	Operating policy Vertical speed (s_v) Horizontal speed (s_h)	dual-command cycle 0.6 (meter per second) 2.0 (meter per second)
Rotating speed of a DMM (s_c)		0.3 (meter per second)
Rack capacity (V_N)		1000
t_v under a normalized SIT rack (t_v=t_h and V_T=1)		0.5
Rack shape		SIT (t_v=t_h)
Number of storage requests $\|S_1\|$		5, 30, 50, 100, 200, 500
Number of retrievals ($\|R_1\|$)		5, 15, 30, 50
Rack utilization (1-$\|S_1\|$/V_N)×100%.		50%-99.5%

Note. The values of the unit load size, s_v, s_h and s_c vary in practice, but have little influence on subsequent results as V_N commonly is large (more than hundreds of unit loads). Therefore, we do not carry out sensitivity analyses for them.

Step 1 (design a discrete rack close to the given rack shape factor t_v): The rack dimensions must be an integral multiple of the pallet dimensions. In our case the rack horizontal dimension must be an even multiple of the pallet's horizontal dimension because the conveyors work in pairs in the DMMs. Therefore, we choose rack dimensions such that they are as close as possible to the given rack shape factor t_v (with t_h t_v t_c =1 and $t_h = t_v$) while the system storage capacity expressed in number of unit loads is no less than V_N . We can approximately calculate the horizontal rack length in number of unit loads, N_x , by rounding $\sqrt[3]{V_T}\, t_h / (v / s_h)$ to the closest even number, the vertical rack length in number of unit loads, N_y , by rounding $\sqrt[3]{V_T}\, t_v / (h / s_v)$ to the closest integer, and the rack depth in number of unit loads, N_z , by rounding $V_N / (N_x N_y)$ to the closest integer; such that $N_x N_y N_z$ is no less than V_N . The real rack capacity then equals $(N_x$ ´ $h / s_h)$ ´ $(N_y$ ´ $v / s_v)$ ´ $(2N_z$ ´ $c / s_c)$.

Step 2 (generate the coordinates of the S/R requests): For every S/R request, randomly generate three integers (n_x, n_y, n_z) within the intervals [1, N_x], [1, N_y] and [1, N_z], respectively for every S/R point in

S and *R*. The *x* coordinate (in seconds) can be calculated by $(n_x - 0.5) \times h / s_h$ (if n_x is an odd number) or $(n_x - 1.5) \times h / s_h$ (if n_x is an even number). The *y* coordinate (in seconds) can be calculated by $(n_y - 0.5) \times v / s_v$, and the *z* coordinate can be calculated by $(n_z - 0.5) \times c / s_c$ (if n_x is an even number) or $(n_z - 0.5 + Nz) \times c / s_c$ (if n_x is an odd number). If two identical S/R points are generated, we generate a new one.

Step 3 (run heuristics): Using the heuristics given in Section 4, we obtain the corresponding results of each heuristic for each replication. The makespan $TCT = \sum_{i=1}^{|R|} DC_{sr}^{i}$.

Step 4 (check confidence level): According to Law (2007), we determine the minimal number of replications, N_{repli}, such that the relative error is smaller than γ $(0 < \gamma < 1)$ with a probability $1-\varepsilon$:

$$N_{repli} \geq S^2(N_{repli})[(1+\gamma)z_{1-\varepsilon/2} / (\gamma X(N_{repli}))]^2, \tag{17}$$

where $S^2(N_{repli})$ is the sample variance of TCT , $z_{1-\varepsilon/2}$ is the $1- \varepsilon / 2$ percentile of the standard normal distribution, and $X(N_{repli})$ is the sample mean of TCT . We set $\gamma = 1.5\%$ and $\varepsilon = 5\%$. In our experiment, we check condition (17) every 20 replications. If it is satisfied, we go to the next step. Otherwise go to step 2.

Step 5 (Normalize the average travel time of a DC): In order to make all the results obtained with different parameters comparable, we normalize the average DC travel time by basing it on a rack with $V_T = 1$, using Theorem 2. The average DC travel time of a DC is $X(N_{repli}) / |R_1|$, and the normalized result equals $(X(N_{repli}) / |R_1|) / \sqrt[3]{V_T^{real}}$, where $V_T^{real} = (N_x \times h / s_h) \times (N_y \times v / s_v) \times (2N_z \times c / s_c)$.

5.2 Results

All the results based on Table 1 are given in Table 2. Table 3 contains the results by varying V_N from 1000 to 5000. Table 4 shows the results for a non-SIT rack with $t_v/t_h=2$. Figure 4 gives the DC times under a wide range of $|R_1|$ between 1-100 and under rack utilizations of 70% and 95%. All the results are normalized to those with $V_T = 1$ by using Theorem 2. The computation time of each replication is less than a second.

Based on these results, we make the following observations:

1) *PPR-SL is much better than the other heuristics: FCFS, NN, SDC and SL*. Tables 2-4 show that PPR-SL substantially outperforms SL, the second best performer, if the rack utilization is below 97%. Figure 4 shows that such high improvements hold for a large range of $|R_1|$ between 1-100 regardless of the rack utilization which is 95% in Figure 4(a) and 70% in Figure 4(b). This is because SL is only a special case of PPR-SL by fixing α =0, whereas PPR-SL chooses the optimal α as the final results.

Table 2. Results of the Instances for the Base Scenarios (V_N=1000)

$\|R_1\|$	$\|S_1\|$	*Util* (%)	REP	Normalized average DC time					Improvement over FCFS (%)				Imp' (%)	α^*
				FCFS	NN	SDC	SL	PPR-SL	NN	SDC	SL	PPR-SL		
5	5	99.5	770	1.30	1.36	0.99	0.98	0.95	-5.14	23.67	24.25	26.41	2.86	0.5
5	30	97.0	1470	1.34	1.77	1.11	1.10	0.86	-31.77	17.17	17.92	36.09	22.15	0.3
5	50	95.0	1590	1.44	1.99	1.25	1.24	0.85	-38.50	13.32	14.03	40.68	30.99	0.2
5	100	90.0	1550	1.62	2.36	1.48	1.48	0.85	-45.92	8.66	8.78	47.66	42.62	0.2
5	200	80.0	1290	1.87	2.72	1.75	1.74	0.85	-45.65	6.41	6.70	54.71	51.46	0.1
5	500	50.0	1030	2.13	3.04	2.04	2.04	0.85	-43.02	3.98	4.02	60.19	58.52	0.1
15	5	99.5	390	1.15	1.03	0.85	0.83	0.80	10.74	26.64	27.58	30.44	3.94	0.5
15	30	97.0	530	1.26	1.55	0.95	0.94	0.71	-23.38	24.22	25.30	43.36	24.18	0.4
15	50	95.0	630	1.37	1.81	1.07	1.05	0.70	-32.51	21.70	23.03	49.13	33.91	0.4
15	100	90.0	550	1.55	2.21	1.30	1.29	0.70	-43.18	15.60	16.39	55.02	46.20	0.4
15	200	80.0	510	1.82	2.63	1.59	1.58	0.69	-44.74	12.59	13.03	61.97	56.27	0.4
15	500	50.0	350	2.10	2.93	1.89	1.89	0.69	-39.45	9.94	9.93	67.17	63.55	0.4
30	5	99.5	230	1.15	0.93	0.80	0.80	0.78	18.98	30.26	30.56	32.46	2.74	0.7
30	30	97.0	310	1.32	1.50	0.92	0.90	0.70	-13.15	30.69	31.75	46.83	22.09	0.6
30	50	95.0	290	1.43	1.79	1.02	1.01	0.69	-25.41	28.55	29.20	51.64	31.70	0.6
30	100	90.0	270	1.61	2.19	1.22	1.20	0.69	-36.59	24.14	25.32	57.18	42.65	0.6
30	200	80.0	250	1.86	2.54	1.46	1.45	0.68	-36.93	21.27	21.60	63.27	53.14	0.6
30	500	50.0	190	2.11	2.83	1.72	1.72	0.68	-34.18	18.32	18.59	68.00	60.70	0.6
50	5	99.5	130	1.20	0.89	0.79	0.79	0.77	25.98	34.39	34.58	36.03	2.23	0.8
50	30	97.0	170	1.38	1.47	0.88	0.88	0.71	-6.30	36.11	36.47	48.32	18.65	0.7
50	50	95.0	170	1.48	1.77	0.96	0.95	0.70	-20.03	35.10	35.38	52.77	26.92	0.7
50	100	90.0	150	1.66	2.14	1.12	1.11	0.68	-28.44	32.69	33.09	58.89	38.57	0.7
50	200	80.0	170	1.88	2.48	1.32	1.31	0.68	-32.24	29.89	30.24	63.54	47.74	0.7
50	500	50.0	130	2.11	2.74	1.53	1.53	0.68	-30.06	27.33	27.44	67.66	55.42	0.7

Note.1) "*Util*" is the rack utilization=100×(the number of filled slots)/V_N. 2) "REP" is the number of replications to satisfy the required confidence level and interval defined in Equation (17) for all the heuristics. 3) "Imp'" is the improvement of PPR-SL over SL. 4) α^* is the optimal priority percentage. 5) The lengths in number of unit loads in horizontal, vertical and depth dimensions are rounded to 18, 5, and 12 respectively.

2) *The average DC time appears to be nearly convex in the number of retrievals* ($|R_1|$) for all heuristics. Figure 4 shows that, for $|R_1|$<10, increasing $|R_1|$ can decrease the DC times of SDC, SL, and PPR-

SL significantly. However, the DC times hardly decrease when $|R_1|$ is beyond 30. For FCFS, the DC time even increases with an increase of $|R_1|$ beyond a certain value (e.g. 15 in Figure 4). Overall, all the heuristics reach the near-shortest DC time at a $|R_1|$ value between 10 and 30 in Tables 2-3 and Figure 4 if the rack utilizations are lower than 95%. If $|R_1|>30$, open and retrieval locations are so close to each other that the travel-between times hardly reduce further, and many requested retrieval locations have already been prepositioned to their S/R positions.

Table 3. Results for V_N=5000

$\|R_1\|$	$\|S_1\|$	*Util* (%)	REP	Normalized average DC time					Improvement over FCFS (%)				Imp' (%)	α^*
				FCFS	NN	SDC	SL	PPR-SL	NN	SDC	SL	PPR-SL		
5	5	99.9	550	1.25	1.30	0.97	0.96	0.95	-4.15	22.02	22.95	24.04	1.42	0.5
5	30	99.4	1150	1.23	1.45	0.98	0.97	0.87	-18.58	19.91	20.68	29.36	10.93	0.2
5	50	99.0	1390	1.28	1.61	1.05	1.04	0.87	-25.26	17.94	18.77	32.53	16.93	0.2
5	100	98.0	1790	1.39	1.88	1.21	1.21	0.86	-35.11	12.54	13.13	37.76	28.36	0.1
5	200	96.0	1630	1.55	2.25	1.43	1.42	0.86	-44.86	7.98	8.47	44.72	39.60	0.1
5	500	90.0	1290	1.90	2.79	1.82	1.81	0.86	-46.84	4.27	4.48	54.57	52.44	0.1
15	5	99.9	310	1.05	0.95	0.79	0.79	0.77	9.58	23.97	24.69	26.53	2.45	0.6
15	30	99.4	570	1.06	1.17	0.82	0.81	0.69	-10.52	22.34	23.80	35.02	14.72	0.3
15	50	99.0	630	1.10	1.32	0.87	0.86	0.68	-19.90	20.53	21.76	37.91	20.63	0.3
15	100	98.0	650	1.22	1.65	1.04	1.02	0.67	-34.38	15.48	16.81	44.92	33.79	0.2
15	200	96.0	630	1.42	2.04	1.26	1.25	0.67	-44.07	11.30	11.92	52.67	46.27	0.2
15	500	90.0	490	1.80	2.64	1.68	1.67	0.67	-46.88	6.59	6.97	62.80	60.01	0.2
30	5	99.9	170	1.01	0.84	0.75	0.75	0.74	16.66	25.53	25.65	26.90	1.69	0.7
30	30	99.4	330	1.08	1.10	0.80	0.79	0.68	-1.96	25.71	26.75	36.91	13.87	0.4
30	50	99.0	350	1.12	1.26	0.86	0.85	0.67	-12.92	23.37	24.26	39.70	20.38	0.4
30	100	98.0	330	1.23	1.58	0.99	0.98	0.67	-28.50	19.09	20.12	45.67	31.99	0.3
30	200	96.0	350	1.46	2.05	1.24	1.24	0.66	-40.67	14.86	15.31	54.47	46.24	0.3
30	500	90.0	230	1.83	2.64	1.65	1.65	0.66	-44.60	9.59	9.80	63.59	59.63	0.3
50	5	99.9	150	1.05	0.80	0.74	0.74	0.73	23.43	29.07	29.26	30.65	1.97	0.7
50	30	99.4	230	1.11	1.08	0.78	0.78	0.68	3.34	29.61	30.27	38.74	12.14	0.6
50	50	99.0	230	1.17	1.25	0.85	0.84	0.67	-6.54	27.32	27.98	42.44	20.08	0.5
50	100	98.0	190	1.29	1.60	0.98	0.97	0.67	-23.99	24.14	24.78	48.51	31.54	0.4
50	200	96.0	210	1.46	2.00	1.17	1.16	0.66	-36.67	20.42	20.60	54.86	43.15	0.4
50	500	90.0	130	1.83	2.59	1.56	1.56	0.66	-41.89	14.55	14.72	63.70	57.43	0.4

Note. 1) "*Util*" is the rack utilization=100× (the number of filled slots)/V_N. 2) "REP" is the number of replications to satisfy the required confidence level and interval defined in Equation (17) for all the heuristics. 3) "Imp'" is the improvement of PPR-SL over SL. 4) α^* is the optimal priority percentage. 5) The lengths in number of unit loads in horizontal, vertical and depth dimensions are rounded to 30, 8, and 21 respectively.

3) *Increasing the number of open locations $|S_1|$ (or reducing the rack utilization) can increase the average DC time for all heuristics except PPR-SL.* For FCFS, SDC, and SL, the more open locations the system has, the more possibility a DMM has to contain an open location closer to its S/R position than retrieval locations. Because the number of open locations is commonly much larger than that of retrievals, if this open location is prepositioned and will not be used afterwards, the prepositioning of

the open location will lengthen the time to preposition the retrievals in the end. However, PPR-SL keeps on improving with an increase in $|S_1|$. This is because retrieval locations are often prepositioned to their S/R positions with a higher priority than open locations and make retrievals closer to its S/R positions for PPR-SL than the other heuristics. Moreover, as the number of open locations is quite large, an open location close to its S/R position can usually be easily found on a DMM and can then be moved to its S/R position.

Table 4. Results for t_v/t_h=2

$\|R_1\|$	$\|S_1\|$	*Util* *(%)*	REP	Normalized average DC time					Improvement over FCFS (%)				Imp' (%)	α^*
				FCFS	NN	SDC	SL	PPR-SL	NN	SDC	SL	PPR-SL		
5	5	99.5	730	1.39	1.47	1.11	1.08	1.05	-6.07	20.00	21.85	23.98	2.73	0.5
5	30	97.0	1390	1.47	1.90	1.24	1.22	0.95	-29.29	15.69	17.20	35.53	22.13	0.3
5	50	95.0	1450	1.56	2.12	1.36	1.35	0.95	-36.44	12.35	13.45	39.19	29.74	0.2
5	100	90.0	1310	1.76	2.49	1.61	1.59	0.94	-41.45	8.92	9.55	46.54	40.89	0.2
5	200	80.0	1050	1.99	2.85	1.88	1.87	0.94	-43.39	5.49	5.92	52.75	49.78	0.2
5	500	50.0	950	2.25	3.16	2.17	2.17	0.94	-40.50	3.42	3.54	57.88	56.34	0.1
15	5	99.5	430	1.26	1.15	0.94	0.93	0.90	8.98	25.38	26.05	28.88	3.82	0.5
15	30	97.0	530	1.38	1.67	1.07	1.05	0.82	-21.63	22.36	23.79	40.24	21.59	0.4
15	50	95.0	510	1.48	1.94	1.18	1.16	0.81	-30.68	20.32	21.78	45.36	30.14	0.4
15	100	90.0	430	1.70	2.38	1.45	1.43	0.81	-39.57	15.17	16.15	52.71	43.60	0.4
15	200	80.0	390	1.93	2.71	1.70	1.70	0.80	-40.19	11.89	12.31	58.72	52.92	0.4
15	500	50.0	310	2.19	3.02	1.97	1.97	0.80	-37.93	9.92	10.09	63.66	59.58	0.4
30	5	99.5	230	1.30	1.06	0.90	0.90	0.88	18.44	30.73	30.90	32.83	2.79	0.7
30	30	97.0	290	1.43	1.63	1.03	1.01	0.81	-13.68	28.16	29.41	43.36	19.76	0.6
30	50	95.0	250	1.52	1.89	1.11	1.10	0.80	-23.74	27.03	27.83	47.55	27.32	0.6
30	100	90.0	210	1.73	2.30	1.32	1.31	0.80	-32.69	24.01	24.70	54.10	39.04	0.6
30	200	80.0	210	1.97	2.63	1.57	1.56	0.79	-33.92	20.36	20.55	59.85	49.47	0.6
30	500	50.0	210	2.19	2.91	1.78	1.77	0.78	-32.88	18.69	18.91	64.38	56.07	0.7
50	5	99.5	130	1.34	1.01	0.89	0.90	0.88	24.51	33.32	33.11	34.62	2.26	0.7
50	30	97.0	190	1.50	1.58	0.98	0.97	0.82	-5.42	34.69	35.29	45.55	15.85	0.8
50	50	95.0	170	1.60	1.88	1.06	1.05	0.80	-17.39	33.85	34.60	49.71	23.10	0.8
50	100	90.0	130	1.78	2.25	1.22	1.20	0.79	-26.31	31.57	32.42	55.34	33.92	0.7
50	200	80.0	150	1.98	2.54	1.40	1.39	0.79	-28.04	29.47	29.94	59.98	42.88	0.9
50	500	50.0	130	2.22	2.84	1.64	1.64	0.79	-28.09	26.11	26.13	64.58	52.05	0.9

Note. 1) "*Util*" is the rack utilization=100× (the number of filled slots)/V_N. 2) "REP" is the number of replications to satisfy the required confidence level and interval defined in Equation (17) for all the heuristics. 3) "Imp'" is the improvement of PPR-SL over SL. 4) α^* is the optimal priority percentage. 5) The lengths in number of unit loads in horizontal, vertical and depth dimensions are rounded to 12, 7, and 12 respectively.

4) By increasing V_N, the improvement of PPR-SL over FCFS becomes larger for a given rack utilization. Tables 2 and 3 show that, if V_N increases from 1000 to 5000, the improvement increases, for example from 47.66% to 54.57% at $|R_1|$=5 with 90% rack utilization. With an increase in V_N, the rack depth

increases and a smart prepositioning becomes increasingly important, which contributes to larger improvements of PPR-SL over FCFS.

5) Changing the rack shape from SIT (t_v/t_h=1) to Non-SIT (t_v/t_h=2) increases the DC times significantly, with PPR-SL still outperforming the other heuristics by about 30% for not too high rack utilizations. Tables 2 and 4 show that the increases in the DC times are often more than 6% with t_v/t_h changing from 1 to 2. This is because an SIT rack shape is optimal (Yu and De Koster, 2009a; Yu and De Koster, 2009b).

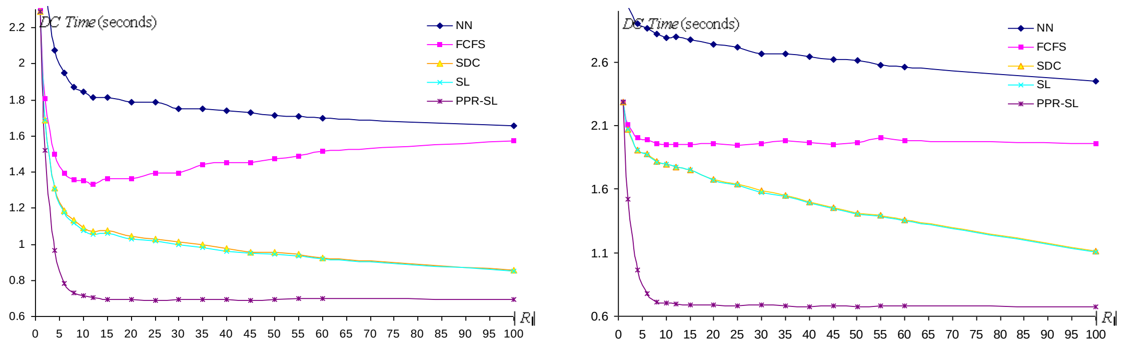


Figure 4: The DC Time Changes with Different $|R_1|$ for the Different Heuristics

Comparing our results with those in 2D systems (Graves et al., 1977; Han et al., 1987; Lee and Schaefer, 1996), we find a striking difference in the ranking of the heuristics. In 2D systems, the high-low performance ranking is SDC, SL, NN, and FCFS. SDC, SL, and NN nearly perform at the same level (Lee and Schaefer, 1996). However, in 3D systems the ranking is SL, SDC, FCFS, and NN for rack utilizations lower than about 95%. SL and SDC have about equal performance. Tables 2-3 show the average DC travel times of FCFS outperform those of NN significantly for rack utilizations lower than about 95% in most cases; even more than 40% in some cases. This is because NN completely disregards the storage travel time, W_s^p, which depends on the sequence of the open locations and retrievals. In contrast, FCFS does not take B_{sr}^p into account, but considers W_s^p in the COL policy. Both the travel-between time, B_{sr}^p, and W_s^p are considered by SDC and SL.

In addition, we find sequencing heuristics in 3D systems can reduce DC travel times to a far greater extent than in 2D systems. NN, SDC, and SL heuristics can obtain 10%-15% travel time reduction over FCFS in 2D systems (Han et al., 1987; Lee and Schaefer, 1996), and can hardly be improved further as the resulting travel times are only about 5% higher than the optimum (Lee and Schaefer, 1996). In 3D systems, compared with FCFS, the time reductions of PPR-SL can often be more than 20% when the rack utilizations are lower than 95%, and even exceed 60% in some cases.

6. Implementing the heuristics in practice

The analyses in Section 5 shows that the DC time is highly dependent on the number of retrievals, on the optimal a values for PPR-SL and on the rack dimension factor t_v (the other two dimensions are determined by $t_h = t_v$, and $t_c = 1/(t_h t_v)$). This section therefore focuses on recommendations to determine these factors for implementation.

6.1 Determining α

This subsection tests how to select a for different t_v values (0.4 and 0.9), and rack utilizations (95% and 70%). Figure 5 gives the results for 4 scenarios where $|R_1|$ is fixed at 10 and 30 for each scenario. Other intermediate values of the rack utilizations, rack shape, and $|R_1|$ give similar graphs and are omitted here.

From Figure 5, we conclude that

the choice of a influences the performance of PPR-SL significantly, but a =1 always yields a near-shortest DC time for PPR-SL.

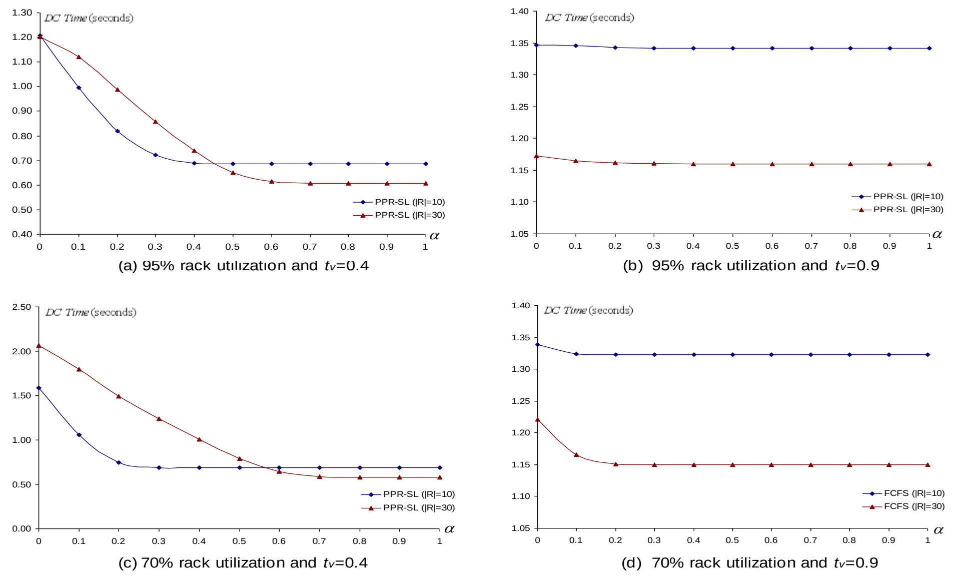


Figure 5: The DC Times as a Function of α under Four Scenarios

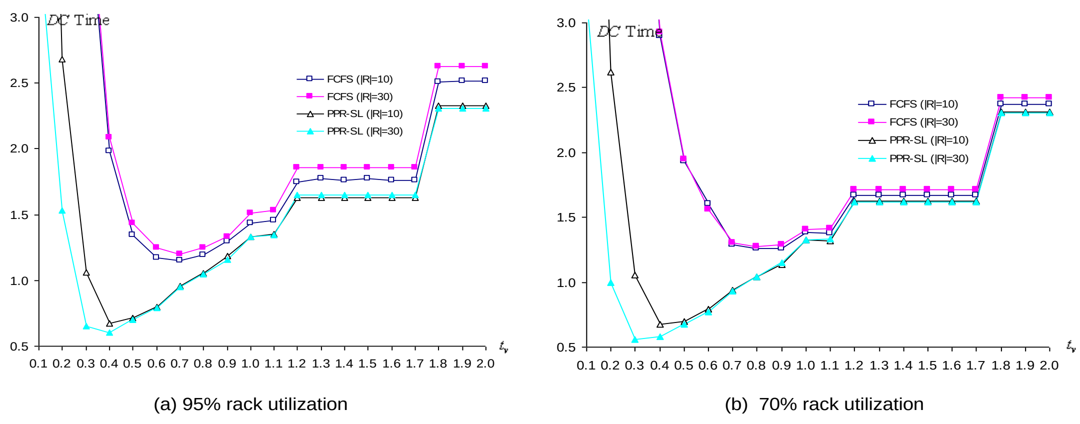
That is to say, if there is a retrieval on a DMM, it should be prepositioned first to its S/R position to reduce the DC time. If the rack utilization is extremely high (e.g. higher than 99.5%), this conclusion may not hold as only few open locations are available.

6.2 Choosing the block size

Figure 4 shows DC times are minimized for $|R_1|$ values between 10 and 30 for FCFS, and for $|R_1|>10$ for PPR-SL, for rack utilizations between 70% and 95%. For practical purposes a choice between 10 and 30 yields good results for all heuristics. Even when the retrieval queue length is longer, it is not necessary to use information beyond 30 retrievals.

6.3 Dimensioning the rack

We test rack configurations (i.e., the length-depth ratio) by assuming an SIT rack for realistic rack-utilization levels of 95% and 70% and block sizes, $|R_1|$, of 10 and 30. The results for FCFS and PPR-SL have been tested in four cases, and can be found in Figure 6(a) (under 95% rack utilization with $|R_1|$=10 and 30) and Figure 6(b) (under 70% rack utilization). From Figure 6, it can be seen that the optimal t_v is always between 0.7-0.8 for FCFS and 0.3-0.4 for PPR-SL and it is very robust for different numbers of retrievals $|R_1|$ and different rack utilizations.



Note. The depth dimension in number of unit loads N_z=84 at t_v=0.1, and N_z=1 at t_v=1.8, 1.9 or 2.0.

Figure 6: The DC Time Changes with Different Rack Dimensions for the Different Heuristics

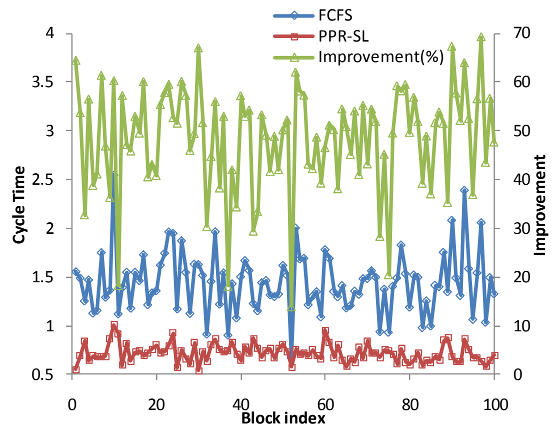
Because of the cost and technology, t_v is in practice normally larger than 0.4 (i.e., 18-unit load deep in our case) and FCFS is commonly used. If technology allows it, warehouse managers therefore can design a deeper rack with PP-SL than with FCFS, to reduce the DC time. The system then becomes more compact.

6.4 Dynamic settings

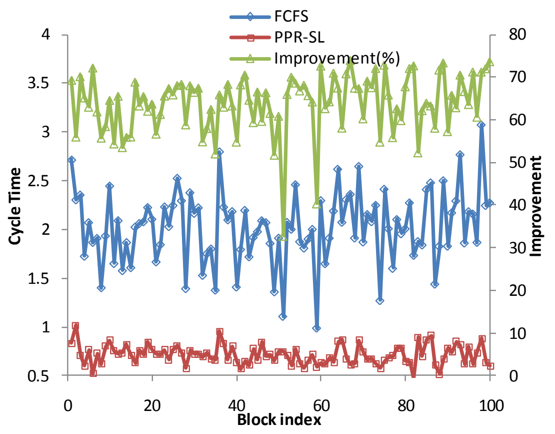
In this section, the performance of the heuristics is studied if blocks of requests dynamically enter the system, which has rack parameters as given in Table 1. Different from the examples shown in Section 5, for every block except the first block, the coordinates of open locations are not randomly generated but result from prepositioning locations in all the previous blocks. All requests arrive at the system block by block and are carried out sequentially. Two scenarios are considered.

We first consider the compact system in combination with an order picking process running in DC mode; unit loads retrieved and brought to the depot have to be returned to the rack after order picking. Each block size is determined by the number of retrievals needed for a customer order and is here randomly generated between 5-20. The average cycle time results after carrying out 100 blocks are shown in Figure 7(a)&(b) under rack utilizations of 95% and 70%, respectively.

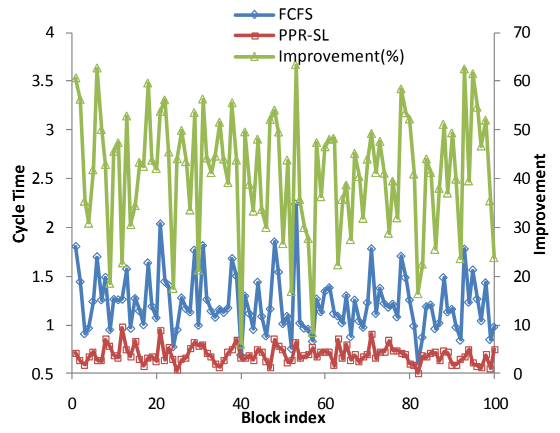
In a second scenario, it is assumed the DCs are interrupted by SCs. In this case, the number of storage requests does not equal the number of retrieval requests. In each block, we randomly generate 5-20 retrieval requests. The number of storage requests for the same block is then randomly generated in a 50% range around the number of retrieval requests, that is [0.75*the number of retrieval request, 1.25*the number of retrieval requests]. To be able to use the heuristics developed for DCs, we add dummy storage requests at the depot if the number of storage requests is smaller than the number of retrieval requests and vice versa (for details refer to Appendix 1). Figure 7(c)&(d) shows the results of the average cycle time under rack utilizations of 95% and 70% with the following inputs.



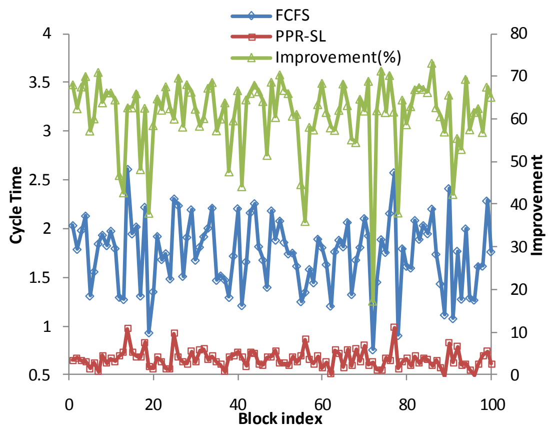
(a) Only DCs under a 95% rack utilization



(b) Only DCs under a 70% rack utilization



(c) Mix of SCs and DCs under a 95 % rack utilization



(d) Mix of SCs and DCs under a 70% rack utilization

Figure 7: The Cycle Time Changes over time

Figure 7(a)-(b) show that also in the dynamic setting, PPR-SL outperforms FCFS substantially on average over 50% under different rack utilizations.

The performance of PPR-SL over FCFS seems not to change over time in Figure 7. Open locations resulting from prepositioning over time in the past blocks may not bring a shorter DC time compared to without prepositioning; one the one hand, prepositioning open location in a past block may move open locations closer to their S/R positions, but on the other hand, prepositioning open locations may most likely move retrieval locations further to their S/R positions. Next, retrieval requests of the current block are only known at the beginning of the block and they cannot be prepositioned before carrying out the current block.

Similarly, in Figure 7(c)-(d), the improvement from PPR-SL over FCFS remains substantial and comparable to the improvement in Figure 7(a)-(b). This is because in a 3D system a smart sequence of SCs can reduce SC travel times due to prepositioning, especially for retrieval requests.

7. Conclusions

This paper is the first to investigate sequencing problems in automated compact warehouses. We address how to sequence a block of storages and retrievals in a 3D compact AS/RS in order to enhance the system operational efficiency under the dual command cycle mode. The sequencing problem is proven to be NP-complete, and thus intractable from a practical standpoint. We adapt four sequencing heuristics: FCFS, SDC, SL and NN known from 2D AS/RS to a 3D environment and propose the new PPR-SL heuristic. All heuristics are evaluated by simulation. We obtain the following insights:

- PPR-SL outperforms all the other heuristics significantly. PPR-SL outperforms SL (the second best heuristic) usually more than 20%
- The a parameter in PPR-SL can be set at a =1, which provides a close-to-best solution if the rack space utilization is not too close to 100% (see Figure 5).
- The makespan is highly influenced by the shape of the rack, but for a given heuristic, the optimal rack shape (determined by t_v) only changes slightly for varying $|S_1|$ and $|R_1|$. The optimal t_v should be between 0.7-0.8 for FCFS and 0.3-0.4 for PPR-SL in our results. Warehouse managers therefore should aim for deeper storage racks, and obtain substantial time savings from the change of dimensions and by using PPR-SL, rather than FCFS (refer to Figure 6).

In contrast to previous research in 2D systems (e.g., Han et al., 1987; Lee and Schaefer, 1996) we find NN does not perform well in the 3D compact system; it is even worse than FCFS. Also, the sequencing problem in 3D systems is much more important than in 2D systems in the sense that in 3D systems DC time can be reduced more compared with FCFS. Compared with FCFS, the DC time can often be reduced more than 50% in 3D systems, while in 2D systems this reduction is limited (Han et al., 1987).

Our research can be extended in several directions. Various storage policies (e.g., random, class-based, full turnover-based storage) are used in compact warehouses. It is interesting to study sequencing problems under such storage policies. Second, too many open locations do not help to decrease the DC time further for the heuristics. Therefore they could be grouped by the system during idle moments of the S/R machine has sufficient idle time by condensing the unit loads. Third, many variants of 3D compact systems exist. Our heuristics can be adapted for different 3D systems and evaluated. Finally, more criteria than the makespan can be used for comparing performance of sequencing heuristics of 3D systems.

Appendices:

Appendix 1. Model SP

Model SP:

$$\min TCT = \sum_{p=1}^{|R_1|} \sum_{s \in S_1 \cup R_1} \sum_{r \in R_1} \lambda_{sr}^{p} DC_{sr}^{p}(Z_s^p, Z_r^p) \tag{18a}$$

Subject to Constraints (1)-(6),

$$\sum_{s \in S_1 \cup R_1} \sum_{r \in R_1} \lambda_{sr}^{p} = 1 \quad p = 1,\ldots,|R_1| \tag{18b}$$

$$\sum_{p=1}^{|R_1|} \sum_{r \in R_1} \lambda_{sr}^{p} \le 1 \quad \forall s \in S_1 \cup R_1 \tag{18c}$$

$$\sum_{p=1}^{|R_1|} \sum_{s \in S_1 \cup R_1} \lambda_{sr}^{p} = 1 \quad \forall r \in R_1 \tag{18d}$$

$$\sum_{r\in R_1} l_{sr}^1 = 0 \qquad \forall s \in R_1 \tag{18e}$$

$$\sum_{t=p}^{|R_1|}\sum_{j\in R_1} l_{rj}^t \le \sum_{t=1}^{p-1}\sum_{s\in S_1\cup R_1} l_{sr}^t \qquad \forall r\in R_1,\ p=2,\dots,|R_1| \tag{18f}$$

$$Z_i^p = \mathrm{mod}\Big(Z_i^1 + \int_{t=0}^{\sum_{q=1}^{p-1}\sum_{s\in S_1\cup R_1}\sum_{r\in R_1} l_{sr}^q DC_{sr}^q(Z_s^q,Z_r^q)} v_i(t)\sigma_c\,dt, t_c\Big),\quad \forall i\in S_1\cup R_1,\ p=2,\dots,|R_1| \tag{18g}$$

$$\Big(\sum_{r\in R_1} l_{sr}^p\Big)\mathrm{mod}\Big(Z_s^p + \int_{\sum_{q=1}^{p-1}\sum_{s\in S_1\cup R_1}\sum_{r\in R_1} l_{sr}^q DC_{sr}^q(Z_s^q,Z_r^q)}^{\sum_{q=1}^{p-1}\sum_{s\in S_1\cup R_1}\sum_{r\in R_1} l_{sr}^q DC_{sr}^q(Z_s^q,Z_r^q)+W_s^p} v_i(t)\sigma_c\,dt, t_c\Big) = 0,\quad \forall s\in S_1\cup R_1\ p=2,\dots,|R_1| \tag{18h}$$

$$\Big(\sum_{s\in S_1\cup R_1} l_{sr}^p\Big)\mathrm{mod}\Big(Z_r^p + \int_{\sum_{q=1}^{p-1}\sum_{s\in S_1\cup R_1}\sum_{r\in R_1} l_{sr}^q DC_{sr}^q(Z_s^q,Z_r^q)}^{\sum_{q=1}^{p-1}\sum_{s\in S_1\cup R_1}\sum_{r\in R_1} l_{sr}^q DC_{sr}^q(Z_s^q,Z_r^q)+W_s^p+ZB_r^{p,ts}} v_i(t)\sigma_c\,dt, t_c\Big) = 0,\quad \forall r\in R_1,\ p=2,\dots,|R_1| \tag{18i}$$

$$v_i(t) = v_j(t),\ \text{if } i, j \text{ are positioned at the same DMM} \tag{18j}$$

$l_{s,r}^p$ are binary decision variables, with $p = 1,\dots,|R_1|$, $s \in S_1 \cup R_1$, and $r \in R_1$. $v_i(t)$ is a binary decision function for each *i*, which means that, at time *t* $v_i(t)$ is a binary variable. $v_i(t)$ represents an infinite number of variables. $v_i(t)$=1 if *i* is being repositioned at time *t*, and =0 otherwise.

Equation (18a) is the objective to minimize the makespan. Constraints (18b) represent that every DC only contains one open-location-retrieval pair (*s*, *r*), and thus $|R_1|$ DCs are carried out in total. Constraints (18c) make sure that an open location can receive at most one unit load. Equations (18d) represent that each retrieval request is processed exactly once. The combination of Constraints (18e) and (18f) ensures that retrieval location *r* cannot serve as an open location in the p^{th} DC if the unit load on it has not been taken away before. The constraints can also be used to eliminate possible subtours as indicated in Lee and Schaefer (1996). Constraints (18g) contain all possible prepositionings of each location $i \in S_1 \cup R_1$. Constraints (18h) and (18i) ensure that, in the *p*th DC, the storage and retrieval locations must be located at their S/R position at *t*=$\sum_{q=1}^{p-1}\sum_{s\in S_1\cup R_1}\sum_{r\in R_1} l_{sr}^q DC_{sr}^q(Z_s^q,Z_r^q) + W_s^p$ and $\sum_{q=1}^{p-1}\sum_{s\in S_1\cup R_1}\sum_{r\in R_1} l_{sr}^q DC_{sr}^q(Z_s^q,Z_r^q) + W_s^p + ZB_r^{p,ts}$, respectively. Constraints (18j) show that for all locations on the same DMM, once one location moves all the others move simultaneously.

The model is also valid in a dynamic setting where blocks of requests arrive online and sequentially. For a dynamic setting, the *z* coordinates of open locations and retrieval requests for a new block should be the updating results of the immediately preceding block.

The model is also applicable to the case where the numbers of storage and retrieval requests are not equal. If there are more storage requests than retrieval requests, we can add some dummy retrieval requests located at the I/O point with (*X*, *Y*, *Z*)=(0,0,0) such that SC storage operations can be considered as virtual DC operations. Similarly, some dummy storage requests at the depot can be added if there are more retrieval than storage requests in a block.

Note that *t* is a continuous value in $v_i(t)$ for every DMM, which complicates the model substantially. *t* has to be discretized but making *t* discrete may reduce the solution quality.

Appendix 2. Proof of Theorem 1

The component $DC_{sr}^{p}(Z_s^p, Z_r^p)$ is a main factor that complicates Model SP because it is dependent on both the previous *p-1* pairs for DCs and their sequence. We simplify Model SP by setting $Z_s^p = 0$ and $Z_r^p = 0$. In this case, DC_{sr}^{p} becomes a function of the coordinates of p^{th} DC pair (*s, r*) only. This simplified model is identical to the sequencing model with multiple open locations defined by Lee and Schaefer (1996) and Han et al. (1987). It has proven to be equivalent to a TSP in its simplified version (where there is only one open location) by Han et al. (1987). TSP belongs to the class of so-called “NP-complete” problems (Garey and Johnson, 1979) in the strong sense (Papadimitriou and Steiglitz, 1998). Hence we have completed the proof.

Appendix 3. Proof of Theorem 2

We rescale all input parameters by setting the dimensions of the rack (in time): $t_h \neg t_h / \sqrt[3]{V_T}$, $t_v \neg t_v / \sqrt[3]{V_T}$, and $t_c \neg t_c / \sqrt[3]{V_T}$, the positions of any open location or retrieval *k*: $X_k \neg X_k / \sqrt[3]{V_T}$, $Y_k \neg Y_k / \sqrt[3]{V_T}$, and $Z_k^p \neg Z_k^p / \sqrt[3]{V_T}$. Substituting these into Equations (2), (3) and (6), we easily obtain

$W_s^p|_{V_T=\bar{V}_T} = W_s^p|_{V_T=1} \times \sqrt[3]{\bar{V}_T}$, $B_{sr}^p|_{V_T=\bar{V}_T} = B_{sr}^p|_{V_T=1} \times \sqrt[3]{\bar{V}_T}$, and $U_r|_{V_T=\bar{V}_T} = U_r|_{V_T=1} \times \sqrt[3]{\bar{V}_T}$. We then substitute these results into Equation (1), and obtain $DC_{sr}^p|_{V_T=\bar{V}_T} = W_s^p|_{V_T=\bar{V}_T} + B_{sr}^p|_{V_T=\bar{V}_T} + U_r|_{V_T=\bar{V}_T}$ $= W_s^p|_{V_T=1} \times \sqrt[3]{\bar{V}_T} + B_{sr}^p|_{V_T=1} \times \sqrt[3]{\bar{V}_T} + U_r|_{V_T=1} \times \sqrt[3]{\bar{V}_T} = DC_{sr}^p|_{V_T=1} \times \sqrt[3]{\bar{V}_T}$.

From Equation (18a), we obtain $TCT_{\bar{V}_T} / TCT_{V_T=1} = \sqrt[3]{\bar{V}_T}$. Hence we have completed the proof of Theorem 2.

References

Bartholdi, J.J. and Platzman, L.K., 1986. Retrieval strategies for a carousel conveyor. *IIE Transactions*, 18(2), 166-173.

De Koster, M.B.M., Le-Anh, T. and van der Meer, J.R., 2004. Testing and classifying vehicle dispatching rules in three real-world settings. *Journal of Operations Management*, 22(4), 369-386.

De Koster, R.B.M., Le-Duc, T. and Yugang, Y., 2008. Optimal storage rack design for a 3-dimensional compact AS/RS. *International Journal of Production Research*, 46(6), 1495-1514.

Dekker, R., De Koster, M.B.M., Roodbergen, K.J. and van Kalleveen, H. Improving order-picking response time at Ankor's warehouse in Interfaces Vol. 34, 2004.

Dooly, D.R. and Lee, H.F., 2008. A shift-based sequencing method for twin-shuttle automated storage and retrieval systems. *IIE Transactions*, 40(6), 586-594.

Garey, M.R. and Johnson, D.S., 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, pp. (New York: W.H. Freeman and Company).

Graves, R.J., McGinnis, L.F., Wilhelm, M.R. and Ward, R.E., 2002. High volume, automated, picking and replenishment systems: characteristics and concepts, In: *Progress in Material Handling Research*, edited by R. Meller and M.K. Ogle and B.A. Peters and G.D. Taylor and J. Usher, pp. 185-203, 2002Charlotte, North Carolina).

Graves, S.C., Hausman, W.H. and Schwarz, L.B., 1977. Storage-retrieval interleaving in automatic warehousing systems. *Management Science*, 23, 935-945.

Gu, J.X., Goetschalckx, M. and McGinnis, L.F., 2007. Research on warehouse operation: A comprehensive review. *European Journal of Operational Research*, 177(1), 1-21.

Han, M.H., McGinnis, L.F., Shieh, J.S. and White, J.A., 1987. On sequencing retrievals in an automated storage/retrieval system. *IIE Transactions*, 19(1), 56-66.

Hu, Y.H., Huang, S.Y., Chen, C.Y., Hsu, W.J., Toh, A.C., Loh, C.K. and Song, T.C., 2005. Travel time analysis of a new automated storage and retrieval system. *Computers & Operations Research*, 32(6), 1515-1544.

Keserla, A. and Peters, B.A., 1994. Analysis of dual-shuttle automated storage/retrieval systems. *Journal of Manufacturing Systems*, 13(6), 424-434.

Law, A.M. and Kelton, W.D., 2007. *Simulation Modeling and Analysis* (4 ed.), pp. (New York: McGraw Hill).

Lee, H.F. and Schaefer, S.K., 1996. Retrieval sequencing for unit-load automated storage and retrieval systems with multiple openings. *International Journal of Production Research*, 34(10), 2943-2962.

Lee, H.F. and Schaefer, S.K., 1997. Sequencing methods for automated storage and retrieval systems with dedicated storage. *Computers & Industrial Engineering*, 32(2), 351-362.

Litvak, N., 2006. Optimal picking of large orders in carousel systems. *Operations Research Letters*, 34(2), 219-227.

Litvak, N. and Vlasiou, M. A Survey on Performance Analysis of Warehouse Carousel Systems, 2008 (Memorandum Department of Applied Mathematics No. 1864, Enschede: University of Twente.

Mahajan, S., Rao, B.V. and Peters, B.A., 1998. A retrieval sequencing heuristic for miniload end-of-aisle automated storage/retrieval systems. *International Journal of Production Research*, 36(6), 1715-1731.

Papadimitriou, C.H. and Steiglitz, K., 1998. *Combinatorial optimization: algorithms and complexity* (2 ed.), pp. (New York: Dover publications, INC.).

Piasecki, D. The Aisle Width Decision. Available online at: http://www.inventoryops.com/Aisle%20Width.htm (accessed on: 26 May 2010).

Retrotech. ACTIV Systems: Super High Density Dynamic Storage Technology. Available online at: http://www.retrotech.com/activ_systems.htm (accessed on: 21 Nov. 2006).

Van den Berg, J.P., 1996. Multiple order pick sequencing in a carousel system: A solvable case of the rural postman problem. *Journal of the Operational Research Society*, 47(12), 1504-1515.

Van den Berg, J.P. and Gademann, A.J.R.M., 1999. Optimal routing in an automated storage/retrieval system with dedicated storage. *IIE Transactions*, 31(5), 407-415.

Van den Berg, J.P. and Gademann, A.J.R.M., 2000. Simulation study of an automated storage/retrieval system. *International Journal of Production Research*, 38(6), 1339-1356.

Viscon. Flexcom System. Available online at: http://www.visserite.com/viscon (accessed on: 22 August 2008).

Wen, U.P. and Chang, D.T., 1988. Picking rules for a carousel conveyor in an automated warehouse. *OMEGA International Journal of Management Science*, 16(2), 145-151.

Westfalia. High Density Automated Storage and Retrieval System. Available online at: http://www.westfaliausa.com/asrs/multiple_deep_storage-01.htm (accessed on: 21 Nov. 2006).

Yu, Y.G. and De Koster, M.B.M., 2009a. Designing an optimal turnover-based storage rack for a 3D compact automated storage and retrieval system. *International Journal of Production Research*, 47(6), 1551-1571.

Yu, Y.G. and De Koster, R.B.M., 2009b. Optimal zone boundaries for two-class-based compact three-dimensional automated storage and retrieval systems. *IIE Transactions*, 41(3), 194-208.

Publications in the Report Series Research[*] in Management

ERIM Research Program: “Business Processes, Logistics and Information Systems”

2011

Sequencing Heuristics for Storing and Retrieving Unit Loads in 3D Compact Automated Warehousing Systems
Yugang Yu and René B.M. De Koster
ERS-2011-003-LIS
http://hdl.handle.net/1765/22722

A Local Search Algorithm for Clustering in Software as a Service Networks
Jelmer P. van der Gaast, Cornelius A. Rietveld, Adriana F. Gabor, and Yingqian Zhang
ERS-2011-004-LIS
http://hdl.handle.net/1765/22723

[*] A complete overview of the ERIM Report Series Research in Management:
https://ep.eur.nl/handle/1765/1

ERIM Research Programs:

LIS	Business Processes, Logistics and Information Systems
ORG	Organizing for Performance
MKT	Marketing
F&A	Finance and Accounting
STR	Strategy and Entrepreneurship