Søren Bloch & Christian H. Christiansen

Simultaneously Optimizing Storage Location Assignment at Forward Area and Reserve Area – a Decomposition Based Heuristic

CORAL - Centre for Operations
Research Applications in Logistics
Simultaneously Optimizing Storage Location Assignment at Forward Area and Reserve Area
- a Decomposition Based Heuristic

Søren Bloch,
Christian H. Christiansen

Center for Operations Research Applications in Logistics
Department of Business Studies
Aarhus School of Business,
University of Aarhus

Abstract: This paper addresses the Two-level Storage Location Assignment Problem (TSLAP) for two level low-level picker-to-part warehouses. The first level consists of a forward area from which all picking is done. The second level consists of a reserve storage area, from which the forward area is replenished. A large part of the cost of operating the warehouse is a function of the assignment of items to locations at both the reserve storage and the forward area. The cost of the storage location assignment at the forward area is dependent on the storage location assignment at the reserve area and visa versa. This, however, is often neglected in the existing literature. We solve the TSLAP simultaneously for the reserve area and the forward area. Based on randomly generated test instances we show that the solutions of TSLAP compare favorably to solutions found by other algorithms proposed in the literature.

Keywords: Warehouse Management, Logistics, Tabu-search, Storage Location Assignment.

1. Introduction

The research body concerning Warehouse Management has been growing steadily in recent years. Companies often turn their attention to the warehouse operating costs when searching for reduction in their logistic costs. For further details on the role of warehouses and their impact on the Logistic costs see e.g. [13]. A large proportion of the warehouse operation costs can be attributed to order picking, and replenishments of locations. A key element in minimizing these costs is an efficient storage location assignment.
We consider a two level low-level picker-to-part warehouse layout. In low-level picker-to-part warehouses an order picker travels to the picking locations and all picking can be made from floor level. The first of the two levels consists of the forward area from which all orders are picked. The capacity of each location at the forward area is limited, and might stock out during operations. If this happens the stocked out location is replenished from the second level which is the reserve storage. In the proposed warehouse layout all goods enter at the reserve area, then flow from the reserve area to the forward area, and exit the warehouse from the forward area. To minimize the cost associated with the above system we formulate the Two-level Storage Location Assignment Problem (TSLAP), in which the objective is to minimize the cost of picking orders (henceforth denoted routing cost) and the cost of replenishing the forward area (henceforth denoted the replenishment cost).

Depending on the warehouse setup alternative flows may exist, and for some warehouse setups several flows may exist simultaneously. For examples of different flows see [10].

To the best of our knowledge, no author has so far addressed the interdependence between the storage assignment at the reverse area and the forward area. In this paper we recognize that some assignment at the forward area might result in a high replenishment cost. Therefore, we propose a heuristic method for simultaneously assigning storage locations to items at the reserve area and the forward area. We report results showing that the simultaneous approach can have a significant impact on the routing and replenishment costs.

The remainder of the paper is organized as follows: Section 2 contains an overview of the related literature. Section 3 contains a mathematical formulation of the proposed model and the decomposition on which our heuristic is based. Section 4 contains a description of the heuristic itself. A testing scheme for our heuristic is developed in section 5, and computational results are reported in section 6. Finally, concluding remarks are given in section 7.

2. Related literature

Since the TSLAP has not been addressed directly in the literature, we consider its two closest related problems. These are the Storage Location Assignment Problem (SLAP)
and the Forward/Reserve Problem (FRP). Several excellent reviews have already been made on these two problems see e.g. [13], [21] and [5]. Therefore we only outline and exemplify the general lines of the SLAP and FRP.

The research area regarding the SLAP is highly active and has been so since its emergence in the early sixties, see [11] and [12]. In general the literature can be put on a continuum between two extreme approaches to solve the SLAP. In one extreme items are allocated to storage locations completely randomly, whereas in the other extreme a storage location is dedicated to a specific item. The choice of assignment strategy often depends on the information available regarding the products to be stored [5]. Especially for low-level picker-to-part warehouses both of these extremes may have drawbacks. To compensate for this, hybrid assignment strategies have been developed. We now discuss each of these three approaches in turn.

Random Location Assignment (RLA) has been addressed in e.g. [9]. The main benefit of this approach is that it maximizes the space utilization, since no location needs to be kept unoccupied due to stocked out items. The main drawback is that the routing cost may become very high indeed, and therefore pure RLA strategies are most often used in fully automated warehouses. For further insight see e.g. [2].

The Dedicated Location Assignment (DLA) lies on the opposite end of the continuum. The major benefit of DLA is a better performance regarding routing costs; however, this benefit generally comes at the expense of lower space utilization. The DLA has received some research interest since its first appearance in [11] and [12]. In these papers the SLAP was solved for a single level warehouse, by computing a Cube-Per-Order Index (COI) for each item. For some item the COI is computed as the ratio between the space needed to store the item and the number of trips required to satisfy its demand for some specified period. Based on this ratio items were assigned to dedicated storage locations. Another publication dealing with COI is [14]. In this paper COI-based dedicated location assignment is compared to random storage. They show that dedicated storage has a shorter order completion time and less waiting time than random storage. On the contrary random storage requires less storage space than dedicated storage. They argue that the choice of strategy should take into account both the cost of storage and the cost of order picking.
Hybrid Location Assignment (HLA) combines the RLA and the DLA, often using a two
step approach. In the first step items are grouped e.g. according to their turnover or their
complementarity. Once these item groups are formed, warehouse space is dedicated to
each item group. Within the space dedicated to an item group RLA is applied. In [9] the
authors address the problem of finding appropriate item groups and assigning them to
storage locations. The work was later extended in [20]. For examples of grouping
according to item turnover see [18] and [15] and examples of grouping items according
to complementarity see [1] and [16].

The objective of the FRP is to determine which items should be stored at the forward
area and which items should be stored at the reserve area, and in what quantity the items
should be stored. The literature regarding the FRP is very sparse, and only a few papers
have been published, since its first mention in [6] and [7]. In their work items are
allocated to an automated storage and retrieval system for fast picking and a general
storage area. The objective is to minimize the total picking and the replenishment costs.
In [22] the workload at the forward area is divided into a picking and idle period, and the
objective is then to determine which replenishments should be made in order to minimize
the expected labor during the picking period at the forward area. Note that the FRP does
not assign items to locations within the storage area, but only considers an overall item
allocation between the reserve area and forward area.

Common for both SLAP and FRP is that they are problems at the tactical level. Hence,
they are determined for a future medium term period (henceforth calls planning period)
based on historical data and/or forecasts.

We propose an algorithm based on DLA which assigns items to dedicated storage
locations at the forward and reserve area simultaneously in order to minimize the routing
and replenishment costs. In fact, our approach combines the FRP and SLAP resulting in a
more integrated storage assignment method than previously seen.

3. Notation and model formulation

We model a two-level low-level picker-to-part storage system containing a reserve area
and a forward area. Each item is stored at exactly one location at the reserve area and one
location at the forward area. We assume that for each item the location at the reserve area holds a sufficient amount to ensure that no stockout occurs. When a location at the forward area is depleted by some order picker, he replenishes the location immediately, before continuing his picking route.

All picking is done within the picking period and exclusively at the forward area. We assume that items are independent in all aspects.

Picking can be done from both sides of an aisle without any additional travel distance. We allow for two way traffic within aisles, but u-turns are not allowed. Furthermore, we assume that the forward area consists of a number of pick aisles each containing a number of pick locations and two cross aisles which are needed for travelling between pick aisles. All picking trips start and end at the “Pickup/Drop off” point (P/D).

For a given order the picker follows a traversal picking strategy. In traversal picking the picker proceeds from the P/D along the front aisle to the outermost left aisle containing a pick and then traverses this aisle until the back aisle is reached. The picker now traverses this cross aisle until the second most left pick aisle containing a pick is reached, and then traverses this pick aisles and so on. When all pick aisles containing a pick have been traversed the picker returns to the P/D. Several other picking strategies have been proposed in the literature. For examples of different picking strategies see e.g. [19], [4] and [17]. The traversal picking strategy is illustrated in Figure 1. Each of the marked locations requires a pick.
At the time of planning we assume that some data are known. This data is the probability of an item being picked, the number of customer orders and the average size of an order. In particular, we define an item’s picking probability as the probability of any single pick regards that particular item. An item’s picking probability can be estimated by looking at historical data. For example if the total number of picks in some period is 10,000 picks of which 300 of these regards item $i$, then item $i$’s picking probability is $300/10,000 = 0.03$.

If the items’ demands and order profiles are stationary this assumption is not restrictive, since this would imply that the picking probabilities are also stationary.

When determining the allocation, we also assume that the number of picks on all picking orders is identical and deterministic. For practical problems this assumption may be restrictive, since the number of picks may vary between orders. We incorporate an evaluation of the consequences of this assumption into the testing of our algorithm. In particular, for some test instance we generate a number of orders in such a way that the number of picks may vary between orders. We then use our algorithm to find a solution based on the average number of picks per order. Finally, we evaluate the quality of our solution on the originally generated orders.

In summary, we develop an algorithm which optimizes the assignment of items to forward locations and reserve locations, based on deterministic picking probabilities for all items and an identical deterministic number of picks per order. We then evaluate our solutions on instances in which the latter assumption is relaxed.
The model

We now give a mathematical formulation of the storage system described above. Denote by $I$ a set of items, by $F$ a set of forward locations and by $R$ a set of reserve locations. Note that because we assume that each item is assigned to exactly one forward location and one reserve location, we have that $|I| = |F| = |R|$. For each combination of item $i \in I$, forward location $f \in F$ and reserve location $r \in R$ we define a binary decision variable $x_{ifr}$ which is 1 if item $i$ is assigned to forward location $f$ and reserve location $r$, and 0 otherwise. With each combination of forward location $f$ and reserve location $r$ we associate a replenishment distance denoted $d_{fr}$, and with each item $i$ we associate the parameters $k_i$ and $a_i$ which is the number of replenishments required by item $i$ within the planning period, and the picking probability of item $i$, respectively. Furthermore, let $N$ be the number of picks per order and let $V$ be the total number of picks within the planning period. To model the routing cost we denote by $CO = V / N$ the number of customer orders in the planning period and by $E[Route]$ the expected travel distance associated with picking a single order. With this notation we can formulate the TSLAP as follows:

\[ P_{(ORG)} : \]

\[
\min \sum_{i \in I} \sum_{f \in F} \sum_{r \in R} x_{ifr} d_{fr} k_i + CO(E[Route]) \tag{1}
\]

s.t.:

\[
\sum_{i \in I} x_{ifr} = 1 \ \forall \ f \in F, \ r \in R \tag{2}
\]

\[
\sum_{i \in I} x_{ifr} = 1 \ \forall \ i \in I, \ r \in R \tag{3}
\]

\[
\sum_{r \in R} x_{ifr} = 1 \ \forall \ i \in I, \ f \in F \tag{4}
\]

\[
x_{ifr} \in \{0,1\} \ \forall \ i \in I, \ f \in F, r \in R \tag{5}
\]

The model $P_{(ORG)}$ is a 3-Dimensional Assignment Problem (3AP) which has been extended to accommodate the routing costs. The first part of (1) contains the replenishing cost. Note that this part is additive, since the contribution from a partial assignment is computed simply by adding the replenishment costs resulting from each positive $x_{ifr}$. The second part concerns the routing cost which is the more complicated part of (1). The
routing costs do not have the additive property, since the contribution from some positive $x_{ijr}$ depends on the values of the remaining $x_{ijr}$. The constraints (2) – (4) and (5) contain the assignment constraints, and the binary constraint on the decision variables, respectively.

Because $CO(E[Route])$ does not possess the additive property, we establish an approximation of $E[Route]$ for some feasible solution to the 3AP defined by the first part of (1) and constraints (2) – (4) and (5).

We now consider the approximation of the routing cost in further detail. We do this in two steps. The first step is to approximate the contribution to the routing cost from a single order of length $N$ based on the $a_i$'s and some setup at the forward area. The second step is then simply to multiply this contribution with $CO$. For a given order, a pick aisle is traversed in full length with some probability. This probability depends on which items are stored at the pick aisle, and the length of the order.

To approximate the contribution from a single order we apply a generalization of the results obtained in [8]. In [8] the author considers random assignment at the forward area and assumes that for a given order each pick aisle has the same probability of being traversed. The author then proceeds to computing the vertical and horizontal travel distance for a given order.

We generalize these computations to the case of dedicated storage locations. As in [8] we decompose $E[Route] = E[Horizon] + E[Vertical]$ where $E[Horizon]$ is the horizontal travel distance (travelling on cross aisles), and $E[Vertical]$ is the vertical travel distance (travelling along pick aisles). However, instead of assuming equal probability of traversal for all pick aisles, we propose an approximation which shows good empirical results.

To address $E[Horizon]$ and $E[Vertical]$ some additional notation is needed. Denote the entire set of pick aisles as $A = A_{\text{Left}} \cup A_{\text{Right}}$, where $A_{\text{Left}}$ and $A_{\text{Right}}$ are the set of pick aisles to the left and right of the P/D, respectively. Furthermore, denote by $M_m$, where $m \in A$, the set of forward locations in aisle $m$, by $P(m)$ the probability of at least one pick in
aisle $m$, by $l$ the length of pick aisles and by $P(Cross)$ the probability that a picker ends up at the back aisle after having finished all $N$ picks following the traversal picking strategy. We then approximate the cost of travelling along pick aisles in the following way:

$$E[Vertical] = l \sum_{m \in A} P(m) + P(Cross)l$$  \hspace{1cm} (6)

Formula (6) contains two elements. The first term is the probability that aisle $m$ contains at least one pick across all aisles multiplied with its length $l$. The second term is the additional distance travelled in case a picker ends up at the back aisle. $P(Cross)$ is a function of the assignment of items at the forward area, the number of pick aisles, the number of locations in each pick aisle, and the order length $N$. To illustrate this consider the following examples in figure 2. Assume that picking can be done from both sides of an aisle, that no U-turns are allowed, and that traversal picking is applied. For convenience we assume that the items have equal picking probabilities. Scenario A contains four items in two pick aisles. In this case if $N=1$ the probability of ending at the opposite cross aisle is 1, if $N=2$ the probability is 1/3, and if $N=3$ or 4 the probability is 0. In scenario B four items are distributed among 3 pick aisles. In this case if $N=1$ the probability of ending at the opposite cross aisle is 1, if $N=2$ the probability is 0, if $N=3$ the probability is 1/2 and if $N=4$ the probability is 1. In scenario C we have six items in two pick aisles. In this case if $N=1$, the probability of ending at the opposite cross aisle is 1. If $N=2$ the probability is 2/5, if $N=3$ the probability is 1/10, and if $N=4$ or 5 or 6 the probability is 0. Clearly, if the picking probabilities are not equal for all items the above probabilities may change.
As this example illustrates, $P(Cross)$ is not easily analysed. To approximate $P(Cross)$ we simply assume that the picker ends at the back aisle in 50% of the orders, hence we set $P(Cross) = 0.5$. This assumption is also done in [8]

The final element of (6) that needs a word of explanation is the computation of $P(m)$, which can be computed as follows:

$$P(m) = \left(1 - \left[1 - \sum_{i=1}^{\nu} \sum_{f \in M_i} \sum_{r \in R} x_{i f r} a_i \right]^N \right)$$

(7)

The summation in square brackets determines the probability that one of the items located in aisle $m$ occurs on the order. Hence, subtracting this from 1 gives the probability that none of the items located in aisle $m$ occur on the order. Note, that picking in an aisle is an event following a binominal distribution; either there is a pick in an aisle or there isn’t. This means that the probability of no picks in an aisle given $N$ items per order is the probability of no pick in an aisle given 1 pick per order raised to the power of $N$. As a result $P(m)$ becomes 1 minus the probability of no picks in $m$.

We now turn our attention to the computation of $E[Horizon]$, but first we need to introduce some further notation. Let $P^L(m)$ [$P^R(m)$] denote the probability that aisle $m$ is the outermost left [right] aisle that requires a visit, and let $w_m$ denote the distance
between pick aisle \( m \) and the P/D. With this notation we can compute \( E[\text{Horizon}] \) as follows:

\[
E[\text{Horizon}] = \sum_{m \in A_{l, eff}} 2w_m P^L(m) + \sum_{m \in A_{r, eff}} 2w_m P^R(m) \tag{8}
\]

The first term of (8) contains the cost of travelling between the outermost left aisle and the P/D. If \( A_{l, eff} = A \) this term constitutes the entire horizontal travelling distance. However, if \( A_{l, eff} \subset A \) we incur some additional travelling distance resulting from travelling between the outermost right aisle and the P/D. This additional travelling cost is contained in the second term. Note, if \( A_{r, eff} = A \) then the entire horizontal travelling cost is included in the second term of (8).

The computation of \( P^L(m) \) and \( P^R(m) \) requires some additional explanation. We number the aisles in \( A \), such that aisle 1 is the outermost left aisle of the entire forward area and aisle \( |A| \) is the outermost right aisle. Furthermore, we define \( \bar{P}(m) = 1 - P(m) \) as the probability of not having a pick in aisle \( m \). \( P^L(m) \) and \( P^R(m) \) are found by calculating the conditional probabilities described below:

\[
P^L(m) = \begin{cases} 
P^L(m) = P(m) & \text{if } m = 1 \\
P^L(m) = P(m) \prod_{i=1}^{m-1} \bar{P}(i) & \text{if } m > 1
\end{cases} \tag{9}
\]

\[
P^R(m) = \begin{cases} 
P^R(m) = P(m) & \text{if } m = |A| \\
P^R(m) = P(m) \prod_{i=m+1}^{|A|} \bar{P}(i) & \text{if } m < |A|
\end{cases} \tag{10}
\]

Formula (9) states that the probability that pick aisle \( m > 1 \) is the outermost left pick aisle containing a pick is the probability that \( m \) contains at least one pick while pick aisles further to the left of the P/D contain no picks. (9) further states that for \( m = 1 \) this probability reduces to \( P(1) \). In a similar way (10) states the probability that pick aisle \( m < |A| \) is the outermost right pick aisle containing a pick, is the probability that \( m \) contains a pick while pick aisles further to the right contain no picks. For \( m = |A| \) this probability reduces to \( P(|A|) \).
**Decomposing \( P_{\text{ORG}} \)**

The model \( P_{\text{ORG}} \) is non-linear because of the routing cost. If the routing cost is assumed to be zero and the \( k_i \)'s are assumed constant then \( P_{\text{ORG}} \) reduces to a standard 3AP, which is NP-complete as shown in [3]. To solve the problem we, therefore, decompose the problem into two new problems: a forward assignment subproblem \( P_{\text{FAP}} \) and a reserve assignment subproblem \( P_{\text{RAP}} \).

The objective of \( P_{\text{FAP}} \) is to find an assignment of items to forward locations which minimizes the routing and replenishment costs, given a fixed assignment of reserve locations to forwards locations. The objective of \( P_{\text{RAP}} \) is then to find an assignment of reserve locations to forward locations which minimizes the replenishment cost, given a fixed assignment of items to forward locations. This decomposition gives us two 2-Dimensional Assignment Problems (2AP), one linear and one non-linear. In particular, the objective function of \( P_{\text{RAP}} \) is linear, whereas the objective function of \( P_{\text{FAP}} \) is non-linear. We now discuss \( P_{\text{FAP}} \) and \( P_{\text{RAP}} \) in further detail.

Let \( x_{if} \) be a binary variable which is 1 if item \( i \) is assigned to forward location \( f \), and 0 otherwise, let \( v_r \) be a parameter which is 1 if forward location \( f \) is assigned to reserve location \( r \) and 0 otherwise, and let \( d_{fr} \) denote the distance from forward location \( f \) to reserve location \( r \). The \( P_{\text{FAP}} \) can then be formulated as follows:

\[
P_{\text{FAP}}: \quad \min CO(E[\text{Route}]) + \sum_{f \in F} \sum_{r \in R} d_{fr} v_r \left( \sum_{i \in I} x_{if} k_i \right)
\]

\[
s.t.: \quad \sum_{i \in I} x_{if} = 1 \quad \forall f \in F \tag{12}
\]

\[
\sum_{f \in F} x_{if} = 1 \quad \forall i \in I \tag{13}
\]

\[
x_{if} \in \{0,1\} \quad \forall i \in I, f \in F \tag{14}
\]
The first part of (11) contains the previously introduced total routing cost which is computed as illustrated in formulas (6) – (10) with the exception that \( P(m) \) is replaced by \( P'(m) \). \( P'(m) \) is a simplification made to fit the 2AP, and is computed as 
\[
P'(m) = \left( 1 - \left( 1 - \sum_{i=1}^{I} \sum_{j \in M_{m}} x_{ij} a_{ij} \right)^{N} \right).
\]
The second term is the replenishment cost resulting from an assignment of items to forward locations. The constraints (12) and (13) ensure that an item (forward location) is assigned to exactly one forward location (item) and finally (14) contains the binary constraint on the decision variables.

Before formulating \( P_{(R,AP)} \), we need to introduce a new set of parameters. In particular, let \( k_{f} \) denote the number of replenishments required for forward location \( f \). With these added parameters, \( P_{(R,AP)} \) can be formulated as a 2AP as follows:

\[
P_{(R,AP)}:
\]
\[
\begin{align*}
\min & \sum_{f \in F} \sum_{r \in R} x_{fr} d_{fr} k_{f} \\
\text{s.t.} & \sum_{r=1}^{R} x_{fr} = 1 \quad \forall f \in F \quad (16) \\
& \sum_{f \in F} x_{fr} = 1 \quad \forall r \in R \quad (17) \\
& x_{fr} \in \{0,1\} \quad \forall f \in F, \ r \in R \quad (18)
\end{align*}
\]

The objective function (15) contains the replenishment cost of some assignment at the forward area. Constraints (16) and (17) ensure that one reserve location is assigned to exactly one forward location and vice versa and, finally, (18) contains the binary constraint on the decision variables. As can be seen from (15) – (18) \( P_{(R,AP)} \) is in fact a standard 2AP which can be efficiently solved to optimality by e.g. the Hungarian method.

In section 4 we develop a solution procedure, which iteratively solves the \( P_{(F,AP)} \) and \( P_{(R,AP)} \) in order to reach some minimum. However, before going into further detail we would like to emphasize exactly how the \( P_{(F,AP)} \) and \( P_{(R,AP)} \) are interrelated.
Property 1:

Knowing a solution to \( P_{(FAP)} \) (i.e. an assignment of items to forward locations) we can compute \( k_f \) for any \( f \in F \) in the following way: \( k_f = \sum_{i \in I} x_{if} k_i \)

Property 1 implies that the solution to \( P_{(FAP)} \) has an effect on the parameters in \( P_{(RAP)} \).

Indeed, if the allocation of items to forward locations is changed, then the \( k_f \)'s may change too, and thereby also the optimal solution to \( P_{(RAP)} \).

Property 2:

Knowing a solution to \( P_{(RAP)} \) (i.e. an assignment of reserve locations to forward locations) we can compute the parameters \( v_{fr} \) simply by: \( v_{fr} = x_{fr} \)

Property 2 implies that the solution to \( P_{(RAP)} \) has an effect on the parameters in \( P_{(FAP)} \). If the solution to \( P_{(RAP)} \) is changed, then the \( v_{fr} \) values change in \( P_{(FAP)} \), and may give rise to a new solution to \( P_{(FAP)} \).

The interdependence between \( P_{(RAP)} \) and \( P_{(FAP)} \), which is reflected in property 1 and 2 is the key to our solution method. Indeed, we exploit this interdependence and develop an algorithm which iteratively solves the \( P_{(RAP)} \) and \( P_{(FAP)} \).

4. The algorithm

We now present our algorithm for solving TSLAP using the above decomposition. A generic illustration of our iterating algorithm is given in Figure 3 below. Let the superscript \( \tau = (1, ..., \infty) \) denote the iteration number (superscript 0 indicates the initializing step of the algorithm). Furthermore, let \( s^\tau \) denote best solution to \( P_{(FAP)} \) found at iteration \( \tau \) and let \( q^\tau \) denote the optimal solution to \( P_{(RAP)} \) at iteration \( \tau \).

Finally, let \( z^\tau \) denote the objective value of \( s^\tau \), let \( s^* \) denote the best found solution to \( P_{(FAP)} \), and \( z^* \) the associated objective value.
The algorithm works as follows. Initially a feasible solution $s^0$ is found to $P_{(FAP)}$. This solution is used to find the $k^0_f \forall f \in F$ parameters using property 1. Based on these we solve $P_{(RAP)}$ resulting in the solution $q^0$. Then using property 2 we find $v^0_\tau$ for all $f \in F, r \in R$. Using (11) we calculate $z^0$, and we set $z^* = z^0$ and $s^* = s^0$. We thereafter set $\tau = 1$. In iteration $\tau$ we first run a tabu search procedure in order to find a new solution $s^\tau$ given the $v^{\tau-1}_\tau$ values. When $s^\tau$ is found we, compute the $k^\tau_f$ values, and check if any stop criteria have been met. If not, we find $q^\tau$ based on the $k^\tau_f$ values, compute the $v^\tau_\tau$ values, increment $\tau$ by 1, and return to the tabu search procedure. In each iteration we evaluate whether the algorithm should be terminated. For this purpose we have two stop criteria. The first criterion is met if $k^\tau_f = k^{\tau-1}_f \forall f \in F$, and the second stop criterion is met if $\tau > \tau_{Max}$, where $\tau_{Max}$ is an upper limit on the number of iterations. The latter criterion is imposed to limit the time consumption of our algorithm. We now turn to a more detailed description of each element of the algorithm.

**Initial solution to $P_{(FAP)}$ and $P_{(RAP)}$**

We establish $s^0$ by modifying volume based location assignment strategies proposed in the SLAP literature. In particular, we assign the item with highest picking probability to the forward location closest to the P/D, the item with the second highest picking probability to the forward location second closest to the P/D, and so on, until all items have been assigned to a forward location. Generally, it is well-established that under volume based storage assignment and traversal routing, it is a good strategy with respect
to routing costs to place items with high volume in the aisle closest to the P/D and items
with low volume far from the P/D see e.g. [2] and [18].

Having established \( s^0 \) we determine \( k_f^0 \) \( \forall f \in F \) and find \( q^0 \) by solving \( P_{(R,AP)} \) based on
the \( k_f^0 \) values. To solve \( P_{(R,AP)} \) we use the well-known Hungarian method.

**Neighbourhood structure in TS procedure**

We define \( S \) as the set of all feasible solutions to \( P_{(F,AP)} \). Each feasible solution \( s \in S \) is
characterized by a set of attributes. The set of attributes associated with solution \( s \) is
denoted \( U(s) \) and contains one attribute for each item, i.e. \( |U(s)| = |I| \). An item \( i \) is
included in exactly one attribute. In the same way a forward location \( f \) is included in
exactly one attribute. An attribute has the format \( (i,f) \) and describes which item \( i \) is
allocated to forward location \( f \). Note that each attribute \( (i,f) \in U(s) \) corresponds to
an \( x_{iy} \) with value 1 in \( P_{(F,AP)} \). The neighbourhood \( N(s) \) of solution \( s \) consists of all solutions
that can be obtained by removing two attributes \( (i,f) \in U(s) \) and \( (i',f') \in U(s) \) from \( U(s) \)
where \( i \neq i', f \neq f' \) and replacing them by the two attributes \( (i,f') \) and \( (i',f) \).

The size of the complete neighbourhood is then \( |I|(|I|−1)/2 \). Even for a relatively small
number of items the neighbourhood becomes very large. Therefore, we apply a partial
evaluation of \( N(s) \). For this purpose we need to construct a restricted neighbourhood
\( n(s) \subset N(s) \) which is then evaluated exhaustively.

However, before explaining the construction of \( n(s) \) we need a few additional definitions.
Denote by \( G \) the set of pick rows in the forward area and by \( C \) the set of pick columns in
the forward area. A pick column contains all locations at one side of a pick aisle, whereas
a pick row contains all locations placed on a horizontal line across the entire forward
area. This is illustrated in Figure 4. We denote by \( F_{Row}(h) \) the set of forward locations in
pick row \( h \) and by \( F_{Col}(h) \) the set of forward locations in pick column \( h \).
We then construct a restricted neighbourhood by selecting $g$ pick rows, randomly. For notational convenience we number the selected pick rows as $(1,...,g)$ and let

$$F_{Sel} = \bigcup_{h=1}^{h=g} F_{Row}(h)$$

denote all forward locations contained in the selected pick rows.

The restricted neighbourhood of $s$ then contains all solutions which can be reached from $s$ by one of two possible swaps. We call these a ‘within pick row swap’ (RS) and a ‘within pick column swap’ (CS), respectively. Both swaps are obtained by removing two attributes $(i,f) \in U(s)$ and $(i',f') \in U(s)$ where $i \neq i', f \neq f'$ and replacing them with the attributes $(i,f')$ and $(i',f)$. When performing a RS within pick row $k$ we add the restriction that $f, f' \in F_{Row}(h)$, and when we perform a CS within column $h$, we add the restrictions $f, f' \in F_{Col}(h)$ and $f, f' \in F_{set}$. The size of a restricted neighbourhood is then

$$gC((C-1)+(r-1))/2.$$

**Evaluation of neighbourhood**

From the neighbourhood structure we know that all neighbours in $n(s)$ are feasible solutions to $P_{(F,AP)}$. This allows us to evaluate the objective value $z'$ associated with the neighbour $s' \in n(s)$ simply by using (11). In particular, for each $s' \in n(s)$ we translate the attributes in $U(s')$ into the corresponding positive $x_{i,f}$ values and then apply (11) to find the corresponding $z'$. We then choose the non-tabu solution in $n(s)$ with the lowest
objective value and denote this solution and its objective value by \( s^0 \) and \( z^0 \), respectively. One exception from this is constituted by our aspiration criterion. If the solution in \( n(s) \) with the lowest objective value is tabu, we choose the solution anyway, if its objective value is lower than \( z^* \).

**Tabu search iteration**

A tabu search iteration consists of two procedures. The first is a Set Selection Procedure (SSP) that selects a subset of \( g \) pick rows. The second is a Neighbourhood Search Procedure (NSP) that determines and evaluates the corresponding \( n(s) \). The NSP is nested within the SSP. We now discuss the procedures in turn.

In the SSP we let \( \beta \) denote the iteration number of SSP, \( \beta_{impr} \) the number of iterations of SSP since an improvement to \( z^* \). Let \( B_{impr} \) denote the upper limit of \( \beta_{impr} \), and \( B_{max} \) denote the upper limit of \( \beta \). For each iteration \( \beta \) a subset of \( g \) pick rows is selected randomly. With these rows as input we run the NSP. Once the NSP has terminated we evaluate \( \beta \) and \( \beta_{impr} \). In particular, if \( \beta < B_{max} \) and \( \beta_{impr} < B_{impr} \) we set \( \beta = \beta + 1 \) and run another iteration of SSP. Otherwise, we terminate the tabu search.

In the NSP we let \( \theta \) denote the iteration number in the Neighbourhood Search Procedure and \( \theta_{impr} \) the number of iterations of NSP since an improvement to \( z^* \). Let \( \Theta_{impr} \) denote the upper limit of \( \theta_{impr} \), and \( \Theta_{max} \) the upper limit of \( \theta \). For each iteration \( \theta \) we first determine \( n(s) \), as described in “Neighbourhood structure in TS procedure”. We then evaluate \( n(s) \) as described in “Evaluation of neighbourhood” to find \( s^0 \) and \( z^0 \). When has been found \( s^0 \) we evaluate if \( z^0 < z^* \). If this is the case we set \( z^* = z^0 \) and \( s^* = s^0 \), and we reset the counters, i.e we set \( \theta_{impr} = \beta_{impr} = 0 \). Furthermore, if \( z^0 < z^* \) we also set \( z^* = z^0 \) and \( s^* = s^0 \).

At the end of each iteration of NSP we evaluate the counters \( \theta \) and \( \theta_{impr} \). If \( \theta < \Theta_{max} \) and \( \theta_{impr} < \Theta_{impr} \) we set \( \theta = \theta + 1 \) and run another iteration in NSP. Otherwise, the NSP is terminated, and we return to the SSP.
Tabu list structure
The tabu list contains the most recent forward locations \((f\) and \(f')\) involved in a swap. Therefore, as long as a set of forward locations is tabu no items can be swapped between the locations unless the aspiration criterion is met. We choose this tabu list structure, because it allows swaps of all items at all times while the tabu on locations ensure that we do not get stuck in the same part of the search area.

In summary, we have presented a tabu search algorithm for implementing TSLAP based on our decomposition. We now turn our attention to creating a testing scheme for evaluating the performance of TSLAP.

5. Construction of test instances
To evaluate the performance of our algorithm for the TSLAP relative to the current SLAP literature, we create a number of test instances. The test instances include information about warehouse dimensions, the distance between forward and reserve locations, the picking probabilities for each item, the size of a unit load for each item, and a set of orders.

The dimension of the forward area is specified by the number of pick aisles and the number of pick rows. We assume that the number of items equals the total number of picking locations. Furthermore, we assume that the reserve area is identical to the forward area. Knowing this and the distance of travelling to an adjacent aisle we create the set of distances \(d_{fr}\) for all \(f \in F\) and \(r \in R\).

Let \(\gamma\) denote the slope parameter of a pareto curve, where \(\gamma\) between 0 and 1, and let \(a_1 \geq a_2 \geq \ldots \geq a_i \geq a_j\). We then calculate the pick probability for each item \(i\) as:

\[
a_i = \frac{(i + 1)^\gamma - i^\gamma}{\sum_{j=1}^i (j + 1)^\gamma - j^\gamma}.
\]

Let \(\zeta\) denote a uniform distributed random variable taking values between 1 and 0. The size of a unit load for an item is then found as \(\zeta\) multiplied by a scaling factor. This is rounded to the nearest integer, and 1 is added.
Finally, a set of orders is generated randomly based on a specification of the number of orders, a set of order sizes, and the proportion that each order size constitutes of the orders. In $P_{(FAP)}$ we let $N$ be the average order size, based on the orders generated.

6. Computational results

In this section we present an evaluation of our heuristic. We evaluate our TSLAP procedure against four benchmark settings. Each of the four settings is found by combining one of two forward location assignment strategies with one of two reserve location strategies.

The first forward location assignment strategy (referred to as down) places items at the forward area such that items with the highest turnover are placed in the aisle closest to the P/D and with the highest turnover item closest to the front aisle. The second forward location assignment strategy (referred to as up) also places items with the highest turnover in aisles closest to the P/D, however, within the aisles items with the highest turnover are placed closest to the back aisle.

In the first of the two reserve location assignment strategies (Reserve strategy 1) the items are assigned to the reserve locations that minimize the replenishment costs given the forward location assignment. This is found by solving an assignment problem similar to $P_{(RAP)}$. In the second reserve location assignment strategy (Reserve strategy 2) items are randomly assigned to reserve locations.

When evaluating a setting which includes reserve strategy 2, we generate 20,000 different random assignments at the reserve area and evaluate the cost for each of these. We then benchmark our algorithm against the average total cost of these 20,000 observations.

We have tested TSLAP on 32 randomly generated test instances. We have tested all combinations of the factors; number of items (80, 120, 128 and 192), order size (5,000 and 10,000), scaling factor (20 and 100), and number of order sizes (1 and 3). In the case of an order size 1, the size is 4 items per order. When 3 order sizes are used 30 p.c. of
orders have 4 items, 40 p.c. have 5 items, and 30 p.c. have 6 items. Finally, a shaping factor of 0.2 has been used to create the pareto curve.

The results of the test instances are shown in table 1. The first column states the name of the test instance. Column two the warehouse dimensions. The third column shows the total cost based on the solution to our TSLAP algorithm. Column four and five show the results obtained by using the literature assignment strategies at the forward area combined with reserve strategy 1. Column six and seven show the average results obtained by using the literature assignment strategies at the forward area combined with reserve strategy 2, and the columns eight and nine show the associated standard deviations. The values in columns four through nine are indexed by the total cost reported in column three.

Table 1: Comparison of TSLAP versus other location assignment strategies
From the table it is seen that our TSLAP algorithm outperforms each of the four benchmark settings for all test instances. On average, the best assignment based on literature strategies “down” and “up” has a travel distance which is 7.1 p.c. higher for “down” and 5.2 p.c. higher for “up”. The corresponding numbers for “down” and “up” when combined with reserve strategy 2 is 13 p.c. and 11.1 p.c., respectively. From table 1 it can also be seen that a higher scaling factor results in smaller improvements. This is because the relative importance of replenishments diminishes when the scaling factor increases. However, our results show that coordination between the reserve area and forward area may lead to substantial savings on routing and replenishment costs.

<table>
<thead>
<tr>
<th>Testinstances</th>
<th>Size</th>
<th>TSLAP</th>
<th>Literature best</th>
<th>Literature random</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Down</td>
<td>Up</td>
</tr>
<tr>
<td>80_5000_20_0_2-1</td>
<td>4X0</td>
<td>2122480</td>
<td>112</td>
<td>110</td>
</tr>
<tr>
<td>80_5000_100_0_2-1</td>
<td>4X0</td>
<td>201393</td>
<td>103</td>
<td>b</td>
</tr>
<tr>
<td>80_5000_20_0_2-3</td>
<td>4X0</td>
<td>238156</td>
<td>109</td>
<td>107</td>
</tr>
<tr>
<td>80_5000_100_0_2-3</td>
<td>4X0</td>
<td>222866</td>
<td>102</td>
<td>101</td>
</tr>
<tr>
<td>80_10000_20_0_2-1</td>
<td>4X0</td>
<td>425965</td>
<td>113</td>
<td>111</td>
</tr>
<tr>
<td>80_10000_100_0_2-1</td>
<td>4X0</td>
<td>405164</td>
<td>102</td>
<td>101</td>
</tr>
<tr>
<td>80_10000_20_0_2-3</td>
<td>4X0</td>
<td>409050</td>
<td>108</td>
<td>105</td>
</tr>
<tr>
<td>80_10000_100_0_2-3</td>
<td>4X0</td>
<td>443065</td>
<td>102</td>
<td>b</td>
</tr>
<tr>
<td>120_5000_20_0_2-1</td>
<td>6X0</td>
<td>256602</td>
<td>111</td>
<td>107</td>
</tr>
<tr>
<td>120_5000_100_0_2-1</td>
<td>6X0</td>
<td>242472</td>
<td>106</td>
<td>a</td>
</tr>
<tr>
<td>120_5000_20_0_2-3</td>
<td>6X0</td>
<td>280678</td>
<td>108</td>
<td>107</td>
</tr>
<tr>
<td>120_5000_100_0_2-3</td>
<td>6X0</td>
<td>271272</td>
<td>102</td>
<td>101</td>
</tr>
<tr>
<td>120_10000_20_0_2-1</td>
<td>6X0</td>
<td>500965</td>
<td>109</td>
<td>103</td>
</tr>
<tr>
<td>120_10000_100_0_2-1</td>
<td>6X0</td>
<td>481491</td>
<td>103</td>
<td>a</td>
</tr>
<tr>
<td>120_10000_20_0_2-3</td>
<td>6X0</td>
<td>572658</td>
<td>106</td>
<td>104</td>
</tr>
<tr>
<td>120_10000_100_0_2-3</td>
<td>6X0</td>
<td>537349</td>
<td>104</td>
<td>103</td>
</tr>
<tr>
<td>120_5000_20_0_2-1</td>
<td>6X6</td>
<td>304370</td>
<td>115</td>
<td>109</td>
</tr>
<tr>
<td>120_5000_100_0_2-1</td>
<td>6X6</td>
<td>275865</td>
<td>104</td>
<td>a</td>
</tr>
<tr>
<td>120_5000_20_0_2-3</td>
<td>6X6</td>
<td>332289</td>
<td>111</td>
<td>108</td>
</tr>
<tr>
<td>120_5000_100_0_2-3</td>
<td>6X6</td>
<td>306069</td>
<td>104</td>
<td>a</td>
</tr>
<tr>
<td>120_10000_20_0_2-1</td>
<td>6X6</td>
<td>509564</td>
<td>114</td>
<td>110</td>
</tr>
<tr>
<td>120_10000_100_0_2-1</td>
<td>6X6</td>
<td>510957</td>
<td>104</td>
<td>102</td>
</tr>
<tr>
<td>120_10000_20_0_2-3</td>
<td>6X6</td>
<td>566904</td>
<td>113</td>
<td>109</td>
</tr>
<tr>
<td>120_10000_100_0_2-3</td>
<td>6X6</td>
<td>613181</td>
<td>102</td>
<td>101</td>
</tr>
<tr>
<td>192_5000_20_0_2-1</td>
<td>8X6</td>
<td>336737</td>
<td>110</td>
<td>103</td>
</tr>
<tr>
<td>192_5000_100_0_2-1</td>
<td>8X6</td>
<td>327084</td>
<td>104</td>
<td>101</td>
</tr>
<tr>
<td>192_5000_20_0_2-3</td>
<td>8X6</td>
<td>389732</td>
<td>108</td>
<td>105</td>
</tr>
<tr>
<td>192_5000_100_0_2-3</td>
<td>8X6</td>
<td>389883</td>
<td>104</td>
<td>103</td>
</tr>
<tr>
<td>192_10000_20_0_2-1</td>
<td>8X6</td>
<td>703453</td>
<td>110</td>
<td>107</td>
</tr>
<tr>
<td>192_10000_100_0_2-1</td>
<td>8X6</td>
<td>659939</td>
<td>103</td>
<td>a</td>
</tr>
<tr>
<td>192_10000_20_0_2-3</td>
<td>8X6</td>
<td>794981</td>
<td>115</td>
<td>110</td>
</tr>
<tr>
<td>192_10000_100_0_2-3</td>
<td>8X6</td>
<td>741825</td>
<td>105</td>
<td>102</td>
</tr>
</tbody>
</table>

**Average of all instances:** 107.1 105.2 113.0 111.1

a: cost of “down” location assignment higher than “up” location assignment
b: cost of “up” location assignment higher than “down” location assignment
7. Concluding remarks

In this paper we have presented the Two-level Storage Location Assignment Problem for a two-level low-level picker-to-part warehouse. The problem is to simultaneously minimize the cost of order picking and replenishment of the forward area by assigning items to dedicated storage locations at the forward and reserve area. We show that this problem can be modeled as an extended 3-Dimensional Assignment Problem, which is non-linear. We then present a way to decompose the problem into two 2-Dimensional Assignment Problems, the forward assignment problem and the reserve assignment problem, respectively. The forward assignment problem finds an assignment of items to forward locations that minimizes the routing and replenishment costs, given a fixed assignment of reserve locations to forward locations. The reserve assignment problem finds an assignment of reserve locations to forward locations that minimizes the replenishment cost, given a fixed assignment of items to forward locations. Based on the decomposition we develop a tabu-search algorithm, and we develop a set of 32 test instances for evaluation of our algorithm. From our computational experiments on the test instances, we find that our algorithm shows very promising results. Indeed, our algorithm performs better than all of our developed benchmark settings, proving that coordination between forward and reserve area may lead to substantial savings in routing and replenishment costs.
References


12 J. L. Heskett, “Putting the cube-per-order index to work in a warehouse layout”, Transportation and Distribution Management, vol. 4, 1964


L-2008-08  Søren Bloch & Christian H. Christiansen: Simultaneously Optimizing Storage Location Assignment at Forward Area and Reserve Area – a Decomposition Based Heuristic.

L-2008-07  Bisheng Du & Christian Larsen: Base stock policies with degraded service to larger orders.

L-2008-06  Karina H. Kjeldsen: Classification of routing and scheduling problems in liner shipping.


L-2008-03  Christian Larsen: Derivation of confidence intervals of service measures in a base-stock inventory control system with low-frequent demand.


L-2008-01  Jens Lysgaard & Janni Løber: Scheduling participants of Assessment Centres.


L-2007-01  Christian Larsen: The $Q(s,S)$ control policy for the joint replenishment problem extended to the case of correlation among item-demands.


L-2006-09  Christian Larsen & Gudrun P. Kiesmüller: Developing a closed-form cost expression for an $(R,s,nQ)$ policy where the demand process is compound generalized Erlang.

L-2006-07  Geir Brønmo, Bjørn Nygreen & Jens Lysgaard: Column generation approaches to ship scheduling with flexible cargo sizes.


L-2006-03  Christian Larsen: Computation of order and volume fill rates for a base stock inventory control system with heterogeneous demand to investigate which customer class gets the best service.

L-2006-02  Søren Glud Johansen & Anders Thorstenson: Note: Optimal base-stock policy for the inventory system with periodic review, backorders and sequential lead times.

L-2006-01  Christian Larsen & Anders Thorstenson: A comparison between the order and the volume fill rates for a base-stock inventory control system under a compound renewal demand process.

L-2005-02  Michael M. Sørensen: Polyhedral computations for the simple graph partitioning problem.


L-2004-03  Søren Glud Johansen & Anders Thorstenson: The (r,q) policy for the lost-sales inventory system when more than one order may be outstanding.

L-2004-02  Erland Hejn Nielsen: Streams of events and performance of queueing systems: The basic anatomy of arrival/departure processes, when focus is set on autocorrelation.

L-2004-01  Jens Lysgaard: Reachability cuts for the vehicle routing problem with time windows.