Positioning automated guided vehicles in a loop layout

Citation for published version (APA):

Document status and date:
Published: 01/01/1997

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.

Download date: 13. Jun. 2023
A Working Paper in the Management Report Series is intended as a means to communicate the results of recent research to colleagues and all others who may be interested. All papers are preliminary and subject to possibly major revisions. This applies equally to opinions expressed, theories developed, and data used. Comments and suggestions are welcomed and should be directed to the author at the address mentioned below.

POSITIONING AUTOMATED GUIDED VEHICLES IN A LOOP LAYOUT

Noud Gademann, Steef van de Velde

MANAGEMENT REPORT NO. 31(13)

August 1997

Erasmus Universiteit Rotterdam
Faculteit Bedrijfskunde/Rotterdam School of Management
PO Box 1738
3000 DR Rotterdam
The Netherlands
tel. +31 10 4082376; fax. +31 10 4526134
POSITIONING AUTOMATED GUIDED VEHICLES IN A LOOP LAYOUT

Noud Gademann
Faculty of Mechanical Engineering
University of Twente
P.O. Box 217, 7500 AE Enschede, The Netherlands
e-mail address: a.j.r.m.gademann@wb.utwente.nl

Steef van de Velde
Rotterdam School of Management
Erasmus University
P.O. Box 1738, 3000 DR Rotterdam, The Netherlands
e-mail address: s.velde@fac.fbk.eur.nl

July 1997
ABSTRACT

We address the problem of determining the home positions for $m$ automated guided vehicles (AGVs) in a loop layout where $n$ pickup points are positioned along the circumference ($m < n$). A home position is the location where idle AGVs are held until they are assigned to the next transportation task. The home positions need to be selected so as to minimize an objective function of the response times, where the response time for a pickup point is defined as the travel time to the pickup point from the nearest home location.

For the unidirectional flow system, where all AGVs can move in one direction only, we first point out that the problem of minimizing an arbitrary regular cost function can quite straightforwardly be solved in $O(n^2)$ time if $m = 1$ and in $O(mn^m)$ time if $m \geq 2$, which is polynomial for a fixed number $m$ of AGVs. For $m \geq 3$, we can do better, however: we derive a generic $O(mn^3)$ time and $O(mn)$ space dynamic programming algorithm for minimizing any regular function of the response times. For minimizing maximum response time a further gain in efficiency is possible: we show that this problem can be solved in $O(n^2)$ time if $m = 2$ and $O(n^2 \log n)$ time if $m \geq 3$. Our results improve on earlier published work, where it was not only shown that maximum response time can be solved in $O(n)$ time if $m = 1$ but also suggested that problems with $m \geq 2$ are NP-hard.

For the bidirectional flow system, where the AGVs can move in both directions, the problem of determining the home locations is inherently much more difficult. Important objective functions like average response time and maximum response time can nonetheless still be minimized in polynomial time for any number of AGVs. Indeed, the problems of minimizing average response time and minimizing maximum response time can be minimized by the same types of algorithms and in the same amount of time as their unidirectional counterparts.

Keywords: Location; Transportation; Automated Guided Vehicles; Dynamic Programming.
1 Introduction

The task of automated guided vehicles (AGVs) is to pick up parts or items at certain points and to drop them off at others. The performance of an AGV system is generally a decreasing function of the service time, that is, the time between the time that a part becomes available for transportation and the delivery time. This time consists accordingly of two components: waiting time and the travel time between the pickup point and the delivery point. In addition to AGV technology, various tactical and operational control issues affect the performance of the system, including the design of the path layout, the location of pickup and delivery points, the number of AGVs in the system, and the scheduling and routing of the AGVs. We refer to Co and Tanchoco [1991] and King and Wilson (1991) for an overview of the AGV literature on these subjects.

In this paper, we address one such operational issue: the positioning of idle AGVs. AGV idleness is unavoidable when an AGV delivers a part and there is no pickup request to which the AGV can be assigned next. The positions of idle AGVs, however, affect the empty travel times to the pickup points, and as such play a role in the waiting time of the parts. Accordingly, a clever positioning of idle AGVs may improve the performance of an AGV system. In fact, Egbelu (1993) suggests that the following three objectives may be used to determine the home locations of idle AGVs:

- minimizing the maximum response time of any pickup point, where the response time for a pickup point is the empty travel time from the nearest home location;
- minimizing the average response time;
- distributing the idle vehicles evenly in the network.

In this paper, we consider the problem of determining the home locations of m AGVs in a loop layout with n pickup points to minimize a function of the response times. A loop layout consists of a single circuit where the pickup and delivery points are positioned along the circumference. Any positioning ω of the AGVs specifies for each pickup point j a response time Tj(ω), which is defined as the travel time from the nearest home location. Throughout we assume that the travel time is proportional to the distance travelled; in
other words, we assume that all AGVs travel at the same constant speed. When there is no ambiguity, we abbreviate $T_j(\omega)$ to $T_j$ ($j = 1, \ldots, n$). Each pickup point $j$ ($j = 1, \ldots, n$) has a response cost function $f_j$, where $f_j(t)$ denotes the cost incurred if the response time for pickup point $j$ is equal to $t$. We consider only regular cost functions, i.e., we assume throughout that each $f_j(t)$ is a non-decreasing function of $t$, for $j = 1, \ldots, n$. Furthermore, we consider only the static variant of the problem, in which all $m$ AGVs are idle at the same time and no pickup requests are available. Also, we assume to have no knowledge about future tasks, when and where they will arise. Under this assumption, it may be possible, however, that we know the probability $P_j \geq 0$ that the next task is issued by pickup station $j$ ($j = 1, \ldots, n$).

The objective is to minimize the response cost, measured either by a regular minmax objective function $f_{\text{max}} = \max_{1 \leq j \leq n} \{f_j(T_j)\}$, or by a regular minsum objective function $\sum_{j=1}^{n} f_j = \sum_{j=1}^{n} f_j(T_j)$. Specific regular objective functions that we consider are the maximum response time $T_{\text{max}}$, defined as $T_{\text{max}} = \max_{1 \leq j \leq n} \{T_j\}$, and average response time $T$, defined as $T = \frac{1}{n} \sum_{j=1}^{n} T_j$. In case we know the probabilities $P_j$ ($j = 1, \ldots, n$), then relevant objective functions are expected maximum response time and expected average response time. They are in fact special cases of regular minmax and minsum objective functions, however, and for this reason we will not consider them explicitly.

To avoid trivialities, we assume that $n > m$ and that the pickup points are distinct. We consider two types of loop layouts: those with a unidirectional flow system, where all AGVs move in one and the same direction; and those with a bidirectional flow system, where the AGVs can travel in both directions. This type of system with more than one AGV may not be very practical, since AGVs moving in opposite directions may give serious interference and control problems. For a discussion on the practical complications of a multi-AGV bidirectional flow system, we refer to Egbelu (1993).

The problem of determining the home positions in a loop layout was first addressed by Egbelu (1993), who studies the objective of minimizing the maximum response time. He shows that the problem is solvable in polynomial time in case of a single AGV ($m = 1$), both in case of a unidirectional and a bidirectional flow system. He suggests that the problem is NP-hard if $m \geq 2$ for either type of system. He gives an integer non-linear programming formulation for the multi-AGV unidirectional problem and presents a heuristic for its solution, while he proposes two heuristics for the multi-AGV
bidirectional problem.

Kim (1995) considers the problem of positioning a single idle AGV to minimize the average response time. He shows that both the static version of the problem, in which \( m = 1 \), as well as the dynamic version of the problem, where \( m - 1 \) AGVs have already been located, can be solved quite straightforwardly in \( O(n^2) \) time.

In this article, we establish that the static problems of minimizing maximum response time and minimizing average response time are solvable in polynomial time for any number of AGVs, in either type of flow system, thereby refuting Egbelu’s suggestion that these problems are NP-hard if \( m \geq 2 \).

More specifically, our contribution is as follows. For the unidirectional flow system, we first point out in Section 2.1 a property of a class of optimal solutions for minimizing an arbitrary regular function of the response times and observe that as a result of this property any such function can quite straightforwardly be minimized in \( O(n^2) \) time if \( m = 1 \) and in \( O(mn^m) \) time if \( m \geq 2 \), which is polynomial when \( m \) is fixed.

We can do better for \( m \geq 3 \), however. In Section 2.2, we derive a generic \( O(mn^3) \) time and \( O(mn) \) space dynamic programming algorithm for minimizing any regular function of the response times. A still further gain is possible for minimizing maximum response time. In Section 2.3, we show that this problem is solvable in \( O(n^2 \log n) \) time for \( m \geq 3 \).

We address the bidirectional flow system in Section 3. First of all, we point out that it is in general not possible to determine the optimal home locations in polynomial time, even if \( m = 1 \). We show further that two important criteria can be minimized in polynomial time for any number of AGVs by the same types of algorithms as their unidirectional counterparts: average response time can still be minimized in \( O(mn^3) \) time and \( O(mn) \) space and maximum response time can still be minimized in \( O(n^2 \log n) \) time.

Section 4 concludes our paper with some final remarks.
2 The unidirectional flow system

2.1 Notation and preliminaries

Without loss of generality, we assume that the \( n \) pickup points have been consecutively numbered in the direction of the flow. It is immaterial at which pickup point the numbering was started. For matter of notational convenience, we also introduce pickup points 0 and \( n+1 \) and let by definition pickup point 0 be equal to pickup point \( n \) and pickup point \( n+1 \) be equal to pickup point 1. We let \( t_{ij} \) denote the travel time between pickup points \( i \) and \( j \) (\( i = 1, \ldots, n, \quad j = 1, \ldots, n \)). By default, we let \( t_{ii} = 0 \) for \( i = 1, \ldots, n \). Since we have assumed that all AGVs travel at the same constant speed, we have that \( t_{ij} = t_{ik} + t_{kj} \) for any pickup point \( k \) between \( i \) and \( j \) (\( i = 1, \ldots, n, \quad j = 1, \ldots, n \)). In a unidirectional flow system we have that \( t_{ij} + t_{ji} = L \) (\( j \neq i \)), where \( L \) denotes the length of the circuit — so, the distance matrix is not symmetric.

Let \( q \) be the number of distinct positive travel times. Note that \( q \leq n(n-1) \). Furthermore, let \( t_{[k]} \) denote the \( k \)-th smallest of these distinct travel times (\( k = 1, \ldots, q \)). Accordingly, \( t_{[k]} \) corresponds to the travel time between at least one pair of pickup points, say, \((s_k, e_k)\) (\( k = 1, \ldots, q \)). If \( t_{[k]} \) corresponds to more than one pair, then without loss of generality we let \((s_k, e_k)\) be the pair of pickup points with smallest indices.

For minimizing the maximum response time, Egbelu (1993) observes that there exists an optimal solution in which the home positions of the AGVs coincide with \( m \) distinct pickup points. This observation, however, applies to any regular function of the response times. To see this, consider any optimal solution in which some AGV is located in between two pickup points, say, \( j \) and \( j+1 \) — relocating this AGV to pickup point \( j+1 \) will definitely improve the response time of this AGV, and hence never decrease the objective function value. Furthermore, consider any optimal solution in which two or more AGVs are positioned at the same pickup point. Relocating one of them to a pickup point where no AGV is stationed will never decrease the objective function value.

For the special case \( m = 1 \), we can accordingly minimize any regular function of the response times in \( O(n^2) \) time by evaluating the objective function for all possible \( n \) home locations. For \( m \geq 2 \), the above property implies that we may restrict ourselves in our search for an optimal solution.
to all positionings \( \omega \) that can be represented by a string of \( m \) pickup points \((w_1, w_2, \ldots, w_m)\) with \( w_i < w_{i+1} \) for \( i = 1, \ldots, m \). We say then that the AGV positioned at pickup point \( w_i \) (\( i = 1, \ldots, m \)) covers the zone from \( w_i \) up to and including \( w_{i+1} - 1 \). For any given assignment \( \omega \) of AGVs to pickup points, we can compute the objective function value in only \( O(m) \) time, if we have evaluated and stored the partial sums \( \sum_{j=1}^k f_j(t_{i,j}) \) and partial maxcost coefficients \( \max_{1 \leq j \leq k} \{ f_j(t_{i,j}) \} \) for \( k = 1, \ldots, n \), \( l = 1, \ldots, k \) in a preprocessing step. This preprocessing requires only \( O(n^2) \) time altogether.

These properties imply that any regular function can be minimized in \( O(mn^m) \) time, which is polynomial for fixed \( m \). This is achieved by simply enumerating all \( \binom{n}{m} \) possible ways of assigning \( m \) AGVs to \( n \) pickup points, evaluating the objective value of each assignment, which takes \( O(m) \) time, and storing the best solution. Actually, this naive procedure is the best we can do for \( m = 2 \). For \( m \geq 3 \), however, we are able to derive a faster algorithm, as we show in Section 2.2.

2.2 Minimizing an arbitrary regular function

In this section, we derive a polynomial-time dynamic programming algorithm for minimizing an arbitrary regular function of the response times, which runs in \( O(mn^3) \) time and \( O(mn) \) space. The algorithm uses a forward enumeration scheme in which AGVs are successively assigned to the current partial assignment. Consider any assignment of \( k \geq 1 \) AGVs to the pickup points \( i, i+1, \ldots, j-1, j \) subject to the condition that the first AGV is positioned at pickup point \( i \) and the \( k \)-th AGV is positioned at pickup point \( j \), that is, a partial assignment with \( w_1 = i \) and \( w_k = j \). We define such an assignment to be in state \((i, j, k)\). Of course, to assign the remaining \( (m-k) \) AGVs to the pickup points \( j+1, \ldots, n-1, n, 1, \ldots, i-1 \), we need to consider only an assignment with minimum objective value among all assignments in state \((i, j, k)\).

Let \( \omega \) be an assignment with minimum objective value in state \((i, j, k)\) with \( k \geq 2 \). To achieve this state from a previous state, we must decide which pickup points are covered by the \((k-1)\)-th AGV to create \( \omega \). Accordingly, the previous state must be \((i, l, k-1)\) for some \( i+k-2 \leq l < j \), in which case the \((k-1)\)-th AGV covers the pickup points \( l, \ldots, j-1 \). The result is then that the total cost increases by \( \sum_{h=i}^{j-1} f_h(t_{i,h}) \) in case of an arbitrary regular
minsum cost function. If we have an arbitrary regular minmax function, then
the maximum cost of the pickup points 1, . . . , j − 1 is \( \max_{t \leq h < j-1} \{ f_h(t, h) \} \).

This optimality principle leads to a polynomial-time dynamic pro-
m erjing algorithm for minimizing any regular objective function. Let \( F_i(j, k) \)
be the minimum objective value for assigning \( k \) AGVs to the zone \( i, \ldots, j \)
subject to \( w_1 = i \) and \( w_k = j \). We are now ready to give the dynamic
programming recursion. The initialization is

\[
F_i(j, k) = \begin{cases} 
0, & \text{if } j = i \text{ and } k = 1, \\
\infty, & \text{otherwise.}
\end{cases}
\]

If the objective function is of the minmax type, then the recursion for \( k = 2, \ldots, m \), \( j = i + k - 1, \ldots, n \) is given by

\[
F_i(j, k) = \min_{i+k-2 \leq l < j} \left\{ \max \left\{ F_i(l, k-1), \max_{t \leq h < j} \{ f_h(t, h) \} \right\} \right\}.
\]

If the \( m \)-th AGV has been assigned to pickup point \( j \) \( (j = i + m - 1, \ldots, n) \),
then the maximum cost of the pickup points \( j, \ldots, n, 1, \ldots, i - 1 \) is equal to
\( \max_{j \leq h \leq n, 1 \leq h < t} \{ f_h(t, j, h) \} \). We have then that the optimal solution value for
given \( i \), that is, given \( w_1 = i \), is equal to

\[
F_i^* = \min_{i+\sum_{j=1}^{m-1} j \leq n} \left\{ \max \left\{ F_i(j, m), \max_{j \leq h \leq n, 1 \leq h < t} \{ f_h(t, j, h) \} \right\} \right\}.
\]

If the objective function is of the minsum type, then the recursion for \( k = 2, \ldots, m \), \( j = i + k - 1, \ldots, n \) is given by

\[
F_i(j, k) = \min_{i+k-2 \leq l < j} \left\{ F_i(l, k-1) + \sum_{l=1}^{j-1} f_h(t, l, h) \right\}.
\]

If the \( m \)-th AGV has been assigned to pickup point \( j \) \( (j = i + m - 1, \ldots, n) \),
then the total response cost for pickup points \( j, \ldots, n, 1, \ldots, i - 1 \) is equal to

\[
\sum_{h=j}^{n} f_h(t, j, h) + \sum_{h=1}^{i-1} f_h(t, j, h).
\]

Accordingly, the optimal solution value given \( w_1 = i \) is then equal to

\[
F_i^* = \min_{i+\sum_{j=1}^{m-1} j \leq n} \left\{ F_i(j, m) + \sum_{h=j}^{n} f_h(t, j, h) + \sum_{h=1}^{i-1} f_h(t, j, h) \right\}.
\]
For both types of objective functions, the overall optimal solution value is equal to \( \min_{1 \leq i \leq n-m+1} F_i^* \) and the corresponding optimal home locations are found by backtracing.

To implement these algorithms efficiently, the partial sums \( \sum_{h=1}^{l} f_h(t_{i,h}) \) and the maxcost coefficients \( \max_{l \leq h \leq j} \{ f_h(t_{i,h}) \} \) are evaluated and stored for \( l = 1, \ldots, n \) and \( j = l + 1, \ldots, n \) in a preprocessing step, which takes \( O(n^2) \) time. Each application of the recursion equation then takes \( O(n) \) time, and thus the entire algorithm requires \( O(mn^3) \) time and \( O(mn) \) space. In Section 2.3, we show that a further gain in efficiency is possible for the problem of minimizing maximum response time.

### 2.3 Minimizing maximum response time

In this section, we present a quite straightforward \( O(n^2 \log n) \) algorithm for minimizing maximum response time in a unidirectional flow system for any \( m \geq 2 \). Note that for any assignment of AGVs to pickup points we can easily compute the maximum response time \( T_{\text{max}} \) as

\[
T_{\text{max}} = \max_{1 \leq j \leq m} \{ t_{w_j, w_{j+1}-1} \}.
\]

If we denote the minimum maximum response time by \( T_{\text{max}}^* \), then we may conclude that \( T_{\text{max}}^* = t_{[k]} \) for some a priori unknown \( k \) (1 \( \leq k \leq q \)). (Recall that \( t_{[k]} \) is the \( k \)-th smallest travel time and that \( q \) is the number of distinct positive travel times.) Since we do not know \( k \) beforehand we need to guess it — this guessing proceeds in a systematic manner.

First of all, note that the problem of minimizing the maximum response time can be viewed as a finite series of decision problems of the type "is \( T_{\text{max}}^* \leq t_{[k]}?" \) for a given \( k \), where \( k \) is repeatedly adjusted by binary search over the values 1, \ldots, \( q \). If this decision problem can be solved in polynomial time, then we can solve the optimization problem in polynomial time, since \( k \) need be adjusted no more than \( \lceil \log n(n-1) \rceil \) times. The notation \( \lceil x \rceil \) refers to the smallest integer greater than or equal to \( x \).

We now show that the decision problem for any given \( k \) can be solved in \( O(m \log n) \) time. We position then the first AGV at pickup point \( s_k \) and the second AGV at pickup point \( e_k + 1 \) — hence, this first AGV covers a zone of length exactly \( t_{[k]} \). Each next AGV is assigned to the pickup point that is as far away from the last assigned AGV as possible but no further away than \( t_{[k]} \). Now three cases need to be distinguished:
• The circuit can be covered with less than \( m \) AGVs to guarantee the maximum response time \( t_{[k]} \) — hence, the question "is \( T^*_{\max} \leq t_{[k]}?"\) has an affirmative answer.

• The travel time between the home location of the last assigned AGV and pickup point \( n \) if \( s_k = 1 \), or pickup point \( s_k - 1 \) if \( s_k > 1 \), is less than or equal to \( t_{[k]} \) — in this case, the question has an affirmative answer as well.

• The travel time between the home location of the last assigned AGV and pickup point \( n \) if \( s_k = 1 \), or pickup point \( s_k - 1 \) if \( s_k > 1 \), is larger than \( i_{[k]} \) — in this case, the answer to the question is no.

Since each next pickup point can be determined in \( O(\log n) \) time by binary search over the remaining pickup points and \( m - 2 \) pickup points need to be assigned, it takes \( O(m \log n) \) time to solve the decision problem. The entire procedure requires \( O(m \log^2 n) \) time, since we need to solve no more than \([\log n(n - 1)]\) decision problems, once the travel times have been sorted in non-decreasing order. The sorting requires \( O(n^2 \log n) \) time, however, and it is therefore more time-consuming than solving the series of decision problems. Accordingly, this algorithm solves the problem to optimality in \( O(n^2 \log n) \) time for any \( m \geq 2 \). Recall, however, that we have shown in Section 2.1 that the case \( m = 2 \) can be solved in \( O(n^2) \) time by explicit enumeration.

If we have an arbitrary regular minmax function, then essentially the same algorithm applies to solve the problem to optimality. The only difference is that binary search for the optimal solution value takes place over a different interval. Nonetheless, if the optimal solution value is an integer whose logarithm is polynomially bounded in the size of the input, then the problem is still solvable in polynomial time.

3 The bidirectional flow system

3.1 Preliminaries and notation

Minimizing a regular function of the response times in a bidirectional flow system is in general much harder than in a unidirectional flow system. The reason is that we may no longer restrict ourselves to home locations that
concur with pickup points. Consider for example the situation with a single AGV and two pickup points — since the AGV may move in either direction, we may have to position the idle AGV somewhere in between to minimize the objective function. Optimally positioning this AGV proceeds then by minimizing a convex function of a single variable, i.e., the position on the loop. This may be achievable only by methods that converge in an infinite number of steps, such as the Golden Section Method (Wilde and Beightler, 1967). Hence, in general it is not possible to compute the optimal home position of an AGV in polynomial time, even if $m = 1$.

We show, however, that for important criteria like average response time and maximum response time the positioning problem can still be solved to optimality in polynomial time for any number of AGVs. To this end, we identify first some structural properties of a class of optimal solutions. We start by analyzing the special case $m = 1$, although Egbelu (1993) and Kim (1995) have already shown that this is a well-solvable case for both criteria. The case $m = 1$ serves as a stepping stone for analyzing the general case. Furthermore, we believe that our approach is more insightful.

Before we proceed we introduce some notation and terminology. We assume for the remainder that the $n$ pickup points have been numbered in the clockwise direction. Again, we introduce for matter of notational convenience pickup points 0 and $n + 1$ and let by definition pickup point 0 be equal to pickup point $n$ and pickup point $n + 1$ be equal to pickup point 1. We let $t_{ij}$ denote the travel time between pickup points $i$ and $j$ and we assume that the travel times are symmetric, that is, $t_{ij} = t_{ji}$ ($i = 1, \ldots, n$, $j = 1, \ldots, n$). There are hence at most $\frac{1}{2}n(n-1)$ distinct positive travel times. Let $q$ be the number of distinct positive travel times, where $q \leq \frac{1}{2}n(n-1)$, and let $t_{[k]}$ be the $k$-th smallest travel time ($k = 1, \ldots, q$). Each $t_{[k]}$ corresponds therefore to the travel time between at least one pair of pickup points, say, $(s_k, e_k)$ ($k = 1, \ldots, q$). If $t_{[k]}$ corresponds to more than one pair of pickup points, then without loss of generality we let $(s_k, e_k)$ be the pair of pickup points with smallest indices.

### 3.2 A single AGV

Any relevant positioning of the single AGV can be represented by a pair of adjacent points $(j, j + 1)$ ($j = 1, \ldots, n$). This representation indicates that the AGV is located somewhere in the zone between $j + 1$ and $j$ such that
it covers the pickup points between its home location and pickup point \( j \) by moving in the clockwise direction and between its home location and pickup point \( j + 1 \) by moving in the anti-clockwise direction. Hence, the AGV does not traverse the path between \( j \) and \( j + 1 \).

For a given pair \((j, j + 1)\), the crux is of course to find the optimal home location in between. As pointed in Section 3.1, we can in general not solve this problem in polynomial time. But if we could, then the overall problem is solvable in polynomial time, since there are only \( n \) such pairs to consider. For both maximum response time and average response time, we can optimally locate the AGV for any given pair \((j, j + 1)\) \((j = 1, \ldots, n)\). For maximum response time, the AGV need be located exactly halfway between \( j + 1 \) and \( j \). Accordingly, we can minimize maximum response time in a bidirectional flow system in \( O(n) \) time if \( m = 1 \), simply by searching the index for which \( t_{j,j+1} \) is maximal and then locating the AGV exactly in the middle of the zone \( j + 1, \ldots, n, 1, \ldots, j \). For minimizing average response time, there exists an optimal solution in which the home location concurs with one of the pickup points. Accordingly, this problem can be solved in \( O(n^2) \) time, exactly in the same fashion as its unidirectional counterpart.

3.3 Multiple AGVs

For any objective function, we know that there is an optimal solution in which each AGV covers a certain zone between two pickup points and that these zones have no pickup points in common. We can therefore restrict our search for an optimal positioning of AGVs to those that can be represented by a string of \( m \) pickup points \((w_1, \ldots, w_m)\) with \( w_i < w_{i+1} \) for \((i = 1, \ldots, m)\). This string indicates that the \( i \)-th AGV covers the zone \( w_i, \ldots, w_{i+1} - 1 \) for each \( i \) \((i = 1, \ldots, m)\). In the remainder, we refer to such a string of \( m \) distinct pickup points as a partitioning of the loop layout into \( m \) distinct zones.

In analogy to the single AGV case, the crux is to find the optimal location of each AGV within its designated zone. If it can be found in polynomial time, then the positioning problem can be solved in polynomial time for a fixed number of AGVs. This is done by enumerating all \( \binom{n}{m} \) partitionings, optimally locating the \( m \) AGVs within each zone, evaluating the cost of each partitioning, and store the best one. For the problem of minimizing maximum response time, the location for the \( i \)-th AGV is obvious: it is precisely halfway in between the pickup points \( w_i \) and \( w_{i+1} - 1 \). For minimizing av-
verage response time, there exists an optimal solution in which the optimal location of the i-th AGV concurs with one of the pickup points between \( w_i \) and \( w_{i+1} - 1 \) for each \( i = 1, \ldots, m \). Accordingly, both problems can be solved in \( O(n^m) \) and \( O(mn^m) \) time, respectively, which is polynomial for fixed \( m \), and this is the best we can do if \( m = 2 \). For \( m \geq 3 \), we can solve these problems by essentially the same algorithms as their unidirectional counterparts in the same amount of time, as we briefly indicate in the next subsections.

### 3.3.1 Minimizing a special class of regular functions

Consider now any arbitrary regular function for which we can optimally position the AGVs in polynomial time once their zones have been designated. We will show that such objective functions can be minimized in polynomial time by a slightly modified version of the dynamic programming algorithm for minimizing an arbitrary regular function in a unidirectional system, which was presented in Section 2.2.

Consider a partial partitioning of the loop in which \( w_1 = i \) and \( w_k = j \), i.e., a partitioning where \( k \geq 1 \) AGVs cover the zone between \( i \) and \( j \), that is, the pickup points \( i, i + 1, \ldots, j - 1, j \). We define such a partitioning to be in state \((i, j, k)\). To position the remaining \( m - k \) AGVs we need to consider only a partitioning with minimum objective value among all partitionings in this state. In analogy to the unidirectional case, we can derive a partitioning in state \((i, j, k)\) (\( k \geq 2 \)) only from some partitioning in a previous state \((i, l, k - 1)\) for some \( i + k - 2 \leq l < j \), where \( w_1 = i \) and \( w_{k-1} = l \). We only give the dynamic programming recursion for an arbitrary regular minsum objective function \( \sum_{j=1}^{n} f_j \) — the recursion for an arbitrary regular minmax objective function proceeds in a similar fashion.

Let now \( F_i(j, k) \) be the minimum objective value for positioning \( k \) AGVs such that \( w_1 = i \) and \( w_k = j \). Furthermore let \( c_{i,j} \) denote the minimum total cost if a single AGV covers the zone \( i, i + 1, \ldots, j - 1, j \) \((i = 1, \ldots, n, j = i + 1, \ldots, n)\); by default, we let \( c_{1,0} = 0 \). The initialization of the dynamic programming recursion is

\[
F_i(j, k) = \begin{cases} 
0, & \text{if } j = i \text{ and } k = 1, \\
\infty, & \text{otherwise},
\end{cases}
\]

and the recursion for \( k = 2, \ldots, m, j = i + k - 1, \ldots, n \) is given by

\[
F_i(j, k) = \min_{i+k-2 \leq l < j} \{ F_i(l, k-1) + c_{l,j-1} \}.
\]
If the $m$-th AGV has been assigned to pickup point $j$ ($j = i + m - 1, \ldots, n$), then the total response cost for pickup points $j, \ldots, n, 1, \ldots, i - 1$ is equal to $c_{j,n} + c_{1,i-1}$. Accordingly, the optimal solution value given $w_1 = i$ is then equal to

$$F_i^* = \min_{i+m-1 \leq j \leq n} \{F_i(j, m) + c_{j,n} + c_{1,i-1}\}.$$  

The overall optimal solution value is equal to $\min_{1 \leq i \leq n-m+1} F_i^*$ and the corresponding optimal home locations are found by backtracing. Hence, once the cost coefficients $c_{i,j}$ have been computed, the entire algorithm requires $O(mn^3)$ time and $O(mn)$ space — accordingly, if the coefficients $c_{i,j}$ can be computed in polynomial time, then the overall problem can be solved in polynomial time.

For minimizing average response time, we can easily prove that there is an optimal solution in which the AGVs are stationed at $m$ distinct pickup points. This means that these cost coefficients $c_{i,j}$ can be computed in $O(n^2)$ altogether in a preprocessing step; hence, average response time in a bidirectional flow system can be minimized in $O(mn^3)$ time and $O(mn)$ space, just as in a unidirectional system.

### 3.3.2 Minimizing maximum response time

For the problem of minimizing maximum response time, each AGV must be located exactly halfway between the two end stations of each zone. This means that we have

$$T_{\text{max}}^* = \frac{1}{2} \max_{1 \leq i \leq m} \{t_{w_i, w_{i+1}-1}\},$$

where $T_{\text{max}}^*$ is the minimum maximum response time. Hence, we have that $T_{\text{max}}^* = \frac{1}{2} t_{[k]}$ for some a priori unknown $k$ ($k = 1, \ldots, q$). The problem of finding $T_{\text{max}}^*$ and a corresponding optimal partitioning can then be found by solving a series of decision problems "is $T_{\text{max}}^* \leq \frac{1}{2} t_{[k]}"$ for given $k$, where $k$ is repeatedly adjusted over the interval $1, \ldots, q$. Since each decision problem can be solved in $O(m \log n)$ time, by a procedure similar to the one described in Section 2.3 for the unidirectional case, and no more than $\lceil \frac{1}{2} n(n-1) \rceil$ decision problems need to be solved if we apply binary search over the indices $1, \ldots, q$, the entire procedure requires $O(m \log^2 n)$ time, once the travel times have been sorted, which requires $O(n^2 \log n)$ time. Accordingly, the entire procedure requires $O(n^2 \log n)$ time.
4 Conclusions

Our results constitute a complete complexity mapping of determining the home locations of idle AGVs in a loop layout with either a unidirectional, or a bidirectional flow system; we refer to Table 1 for an overview of time complexities. We have shown for the unidirectional flow system that any regular function can be minimized in polynomial time. Determining the home locations of AGVs in a bidirectional flow system is much harder. In fact, this problem is not solvable in polynomial time, even in case of a single AGV. Important criteria like maximum response time and average response time, however, can still be minimized in polynomial time for any number of AGVs.

<table>
<thead>
<tr>
<th>Objective function</th>
<th>Unidirectional</th>
<th>Bidirectional</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_{\text{max}}$</td>
<td>$O(n^2)$ $O(n^2)$ $O(mn^3)$</td>
<td>$-$ $-$ $-$</td>
</tr>
<tr>
<td>$\sum_{j=1}^{n} f_j$</td>
<td>$O(n^2)$ $O(n^2)$ $O(mn^3)$</td>
<td>$-$ $-$ $-$</td>
</tr>
<tr>
<td>$T_{\text{max}}$</td>
<td>$O(n)^{1}$ $O(n^2)$ $O(n^2 \log n)$</td>
<td>$O(n)^{1}$ $O(n^2)$ $O(n^2 \log n)$</td>
</tr>
<tr>
<td>$\sum_{j=1}^{n} \overline{T}_j$</td>
<td>$O(n)^{2}$ $O(n^2)$ $O(mn^3)$</td>
<td>$O(n)^{2}$ $O(n^2)$ $O(mn^3)$</td>
</tr>
</tbody>
</table>

Table 1: Overview of time complexities; ‘—’ indicates ‘not solvable in polynomial time’; $^{1)}$ see Egbelu (1993); $^{2)}$ see Kim (1995).

Finally, it is likely that determining the home locations of idle AGVs in more complex layouts is more difficult — it would be interesting to see for what types of layouts these problems can still be solved in polynomial time.

References


1997


