European Journal of Operational Research 63 (1992) 173–191
North-Holland

Tandem AGV systems: A partitioning algorithm and performance comparison with conventional AGV systems *

Yavuz A. Bozer and Mandyam M. Srinivasan
Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, MI 48109, USA

Received October 1990; revised February 1992

Abstract: In an earlier paper, Bozer and Srinivasan introduced the tandem concept for automated guided vehicle (AGV) systems and presented an analytical model to evaluate the throughput performance of a basic component of the system; namely, a single vehicle serving a set of workstations under the First-Encountered-First-Served rule. In this study, using the above analytical model and certain column generation techniques, we present a heuristic partitioning scheme to configure tandem AGV systems. The partitioning scheme is based on a variation of the well-known set partitioning problem. It is aimed at evenly distributing the workload among all the AGVs in the system. We demonstrate the procedure with two numerical examples. Using simulation, the performance of the tandem configuration obtained for each example is compared to that of the corresponding conventional AGV system.

Keywords: Automated guided vehicle systems; optimization; set partitioning; heuristics; facilities

1. Introduction

The tandem concept for AGV systems was introduced in an earlier paper by Bozer and Srinivasan (see [2] or [3]). As shown in Figure 1(a), in a conventional AGV system all the vehicles are allowed to serve any workstation in the system. In contrast, a tandem AGV system (shown in Figure 1(b)) is obtained by partitioning all the workstations into *single-vehicle, non-overlapping* zones. Additional pick-up/deposit (P/D) points are provided between adjacent zones to serve as 'transfer points'.

The tandem concept offers three principal advantages. First, it dramatically simplifies the control system since only one vehicle must be controlled in each zone and the vehicles never interfere with one another. Second, it eliminates the delays caused by congestion and blocking in conventional AGV systems; and third, it offers more flexibility since zones (and workstations) can be added, removed, or modified without affecting other zones, and different types of vehicles (including bidirectional and/or self-guided vehicles) can be used in different zones. The principal limitations of the tandem concept are as follows: first, a load may have to be handled by several vehicles before it reaches its destination; this will not only require additional pick up and deposit operations but it will also induce additional delays at the transfer points. Second, additional floor space and capital investment is required to provide the transfer points.

* Both authors have been supported by a research grant (DRDA 90-908) from General Motors. The first author is also supported by a Presidential Young Investigator Award under NSF Grant DMC 8858562.
Correspondence to: Y.A. Bozer, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, MI 48109, USA.

There are other advantages and limitations of tandem AGV systems. (The reader may refer to [2] or [3] for further details.) As one can infer from Figure 1b, however, the performance of the tandem configuration is closely related to the configuration of the system; that is, the number of zones and the workstations assigned to each zone. Assuming that the flow data and the workstation locations are given, in this paper we will present a heuristic partitioning scheme (driven by analytical models) to obtain reasonably good configurations for tandem AGV systems. We will demonstrate the partitioning scheme with two numerical examples. We will also use these examples to compare certain performance measures (including the number of AGVs) obtained from simulation for tandem AGV systems and conventional AGV systems. Generally speaking, the number of AGVs required under either system is a function of many design decisions. Therefore, it is difficult to make general statements about vehicle requirements. However, we do believe that, independent of the number of vehicles required, the advantages of the tandem AGV system are likely to offset its limitations in most cases.

2. Definitions and assumptions

The definitions and assumptions introduced in this section are concerned with tandem AGV systems. (Except for empty vehicle dispatching and bidirectional AGVs, the same definitions and assumptions are also valid for conventional AGV systems which are considered later in the study.) As shown in Figure 1, each workstation (which

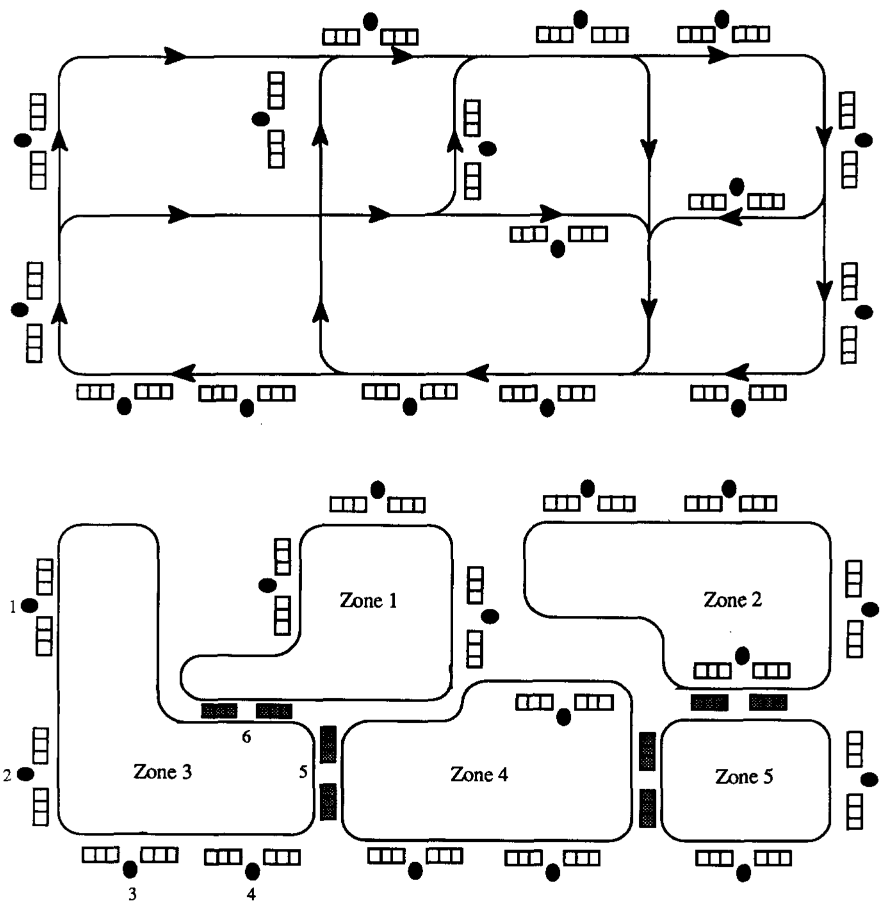


Figure 1. (upper part) Example of a conventional AGV system. (lower part) Example of a tandem AGV system

may represent one machine, or a group of machines, i.e., a cell, or a department) has an input queue and an output queue. The loads destined to a workstation are dropped at its input queue while the loads leaving that workstation are picked up from its output queue. It is assumed that the AGV travel time between the two queues is negligible. It is also assumed that all the queue capacities are such that the workstations or the vehicle seldom get blocked due to a full queue.

There are two types of workstations. The first type is an input/output (I/O) station. Loads that arrive from outside the system, that is, loads that do *not* arrive from another zone, arrive at the *output queue* of an I/O station where they wait to be picked up by the appropriate vehicle. Likewise, loads that require no further processing in the system are dropped off at the *input queue* of an I/O station where they are assumed to instantly leave the system. Note that no processing takes place at an I/O station and that a zone may have no I/O stations or several I/O stations. Also note that flow is not necessarily conserved at an I/O station since a load may enter the system from one I/O station and exit from another.

The second type of workstation is a 'processor station' where actual processing takes place. At a processor station, the loads are removed from the corresponding input queue and after a certain period of time (which represents the processing time) they are deposited at the corresponding output queue. It is assumed that flow is conserved at each processor station. The material handling requirement *within* a processor station is beyond the scope of the study.

In each zone, the vehicle travels from one workstation to another, or it travels between a workstation and a transfer point. Note that the vehicle will travel *directly* from one transfer point to another only if the load in question is a 'transit load', that is, a load which requires no processing in that zone but has to travel through that zone. We assume that the vehicle is a *bidirectional* vehicle. Using such a vehicle under the tandem approach will reduce the travel time associated with each trip without creating the usual conflicts and path contention encountered in conventional AGV systems that utilize bidirectional vehicles.

Each zone must have at least one transfer point. Transfer points are assumed to serve as 'two-way' interface points; i.e., a transfer point between zones p and q handles the flow from p to q as well as the flow from q to p. In this regard, a transfer point may be conceptually viewed as an 'I/O station'. Naturally, in a real-life setting, if two-way flow exists between two adjacent zones, then one would have to provide two one-way transfer points (such as a pair of conveyors). For ease of exposition, we will treat these two (adjacent) one-way transfer points as a single two-way transfer point.

In serving the move requests within a zone, the vehicle is assumed to follow the First-Encountered-First-Served (FEFS) empty vehicle dispatching rule. Under this rule, a vehicle which has just delivered a load will travel empty to poll (or inspect) the output queue of each workstation and transfer point in that zone in order to locate the next move request. Each station is polled exactly once according to a prespecified polling sequence. (It is implicit that, if the vehicle delivers a load at the input queue of workstation i, it first inspects the output queue of workstation i.) For example, in reference to Figure 1b, a valid polling sequence for the third zone would be 1–2–3–4–5–6. In [1] it is shown that the FEFS rule performs well with closed-loop zones such as those shown in Figure 1b. Note that, under the FEFS rule, the oldest move request in a zone is not necessarily the next move request to be served. Also note that, under this rule, the vehicle is never in the idle state; i.e., it is always traveling loaded or traveling empty. (The FEFS dispatching rule is a decentralized rule. In order to determine its state, the AGV must first travel to an output queue and inspect it. Although this may lead to unnecessary empty vehicle travel in systems with light traffic, we adopted the FEFS rule for tandem AGV systems primarily due to its simplicity, and its efficiency in closed-loop zones.)

Consider a *single zone* with *known* workstations and transfer points. (For notational convenience, in the remainder of the paper we will treat transfer points as I/O stations.) Let M denote the number of workstations (in the given zone). Without loss of generality, we assume that the polling sequence is given by $1, 2, \ldots, M$. Further let ϑ and Ω denote the set of processor stations and the set of I/O stations, respectively.

Suppose we are also given the flow data for the zone; i.e., the flow in and out of the zone as well as the flow within the zone. (It is straightforward to obtain such flow data for a given zone from the overall flow data given for the entire system.) Let λ_i denote the rate at which loads arrive at the output queue of workstation i. Likewise, let Λ_i denote the rate at which the vehicle delivers loads to the input queue of workstation i. We assume that $\Lambda_i = \lambda_i$ in steady state for $i \in \vartheta$. (This also implies that a processor station may never be a bottleneck in a zone.) For I/O stations, on the other hand, recall that Λ_i need not equal λ_i in general. However, from conservation of flow within a zone, we must have

$$\sum_{i \in \Omega} \Lambda_i = \sum_{i \in \Omega} \lambda_i,$$

provided that the vehicle is able to meet the required throughput.

Let f_{ij} denote the number of *loaded* trips per time unit that the vehicle must perform from (the output queue of) workstation i to (the input queue of) workstation j. (It is implicit that $f_{ii} = 0$.) Given the f_{ij}-values, we have

$$\lambda_i = \sum_{j=1}^{M} f_{ij} \text{ and } \Lambda_i = \sum_{j=1}^{M} f_{ji}, \tag{1}$$

by definition.

Consider next the travel time associated with each trip. Suppose the empty vehicle picks up a load from the output queue of workstation i, transports it to workstation j, unloads it at the input queue of workstation j, and subsequently inspects the output queue of workstation j. The total time required to perform the above operation – including the pick-up/deposit times and the time required to inspect workstation j – is assumed to be a random variable with mean τ_{ij} time units.

In contrast, if the vehicle has just delivered a load at workstation i and upon inspection has found its output queue empty, then it must perform an empty trip from workstation i to workstation $i+1$ due to the FEFS rule. The resulting empty travel time is assumed to be a random variable with mean $\sigma_{i,i+1}$ time units, which includes the time taken to inspect workstation $i+1$. We let $\sigma_{j,i}$ denote the expected empty vehicle travel time from workstation j to workstation i $(i \neq j)$ *following the polling sequence*. That is,

$$\sigma_{j,i} = \begin{cases} \sum_{k=j}^{i-1} \sigma_{k,k+1} & \text{if } j < i, \\ \sum_{k=j}^{M} \sigma_{k,k+1} + \sum_{k=1}^{i-1} \sigma_{k,k+1} & \text{if } j > i \end{cases} \tag{2}$$

where $k+1 = 1$ when $k = M$.

If all of the above information is given for a particular zone, then following the approach presented in [2] we can determine whether the vehicle will be able to meet the throughput requirement (of that zone) as follows. Let α_f denote the expected proportion of time the vehicle has to travel loaded. (This includes the time required to pick up and deposit loads, by definition.) Since the flow data and the travel times are given, it is straightforward to compute α_f from the following expression:

$$\alpha_f = \sum_{i=1}^{M} \sum_{j=1}^{M} f_{ij} \tau_{ij}. \tag{3}$$

Note that α_f is independent of the empty vehicle dispatching rule. Of course, if $\alpha_f \geq 1$, the vehicle will not be able to meet the required throughput (regardless of the empty vehicle dispatching rule used) and α_f can no longer be defined as a proportion.

Given α_f, in [2] it is shown that under the FEFS empty vehicle dispatching rule, the vehicle will meet the required throughput (in a zone) if

$$\omega = \alpha_f + \phi < 1 \tag{4}$$

where

$$\phi = \max_{i \in \Omega} \phi_i = \max_{i \in \Omega} \left[\sum_{j \neq i} (\Lambda_j - \lambda_j) \sigma_{j,i} \right]. \tag{5}$$

As remarked in [2], ϕ (> 0) acts as a 'correction factor' and it can be interpreted as 'mandatory' empty vehicle travel. (For further discussions on 'mandatory' empty vehicle travel, the reader is referred to [2].) Note that we need to check the ϕ_i-values only for the I/O stations – this includes the transfer points.

For a given zone, ω $(0 < \omega < 1)$ can be viewed as the 'workload' factor. A 'good' configuration can be obtained by letting the analyst control the number of zones (i.e., the number of vehicles), while the partitioning model is used to define each zone (i.e., to determine the workstations to be served by each vehicle) so that the total workload is evenly distributed among all the zones. In other words, the range of the ω-values in the final partition should be reasonably small so that no bottleneck zone is created. (There are additional properties one can state to characterize a 'good' configuration. We will point out these properties in Section 5.)

3. The partitioning algorithm

The partitioning algorithm consists of three phases. In the first phase we employ certain geometric techniques to generate 'promising' subsets of workstations. In the second phase, we evaluate each subset for feasibility; i.e., we use the results given by (2)–(5) to determine whether one vehicle will meet the throughput requirement imposed by the workstations in the subset. (As shown below, the first two phases actually proceed in parallel; i.e., we generate subsets while we check their feasibility.) If a subset is feasible, we save it as a potential zone (or a 'column') for the third phase, which is based on solving a variation of the well-known set partitioning problem using the above potential zones (or 'columns'). Recall that, in the set partitioning problem, a 'column' is defined by its cost coefficient and the set of 'nodes' (or workstations) that it 'covers'. The objective is to cover each 'node' exactly once by using those columns which yield the minimum total cost.

3.1. Generating subsets of workstations

Using brute force to generate and examine the feasibility of all possible subsets of workstations would not only be very time consuming for moderately large problems with, say, 20 workstations but it is also very likely to yield an unnecessarily large number of potential zones which would make the solution of the model used in phase three considerably more difficult.

Furthermore, the brute force approach may generate a potential zone that would be 'difficult' to interface with other zones if it happens to be part of the final solution. For example, consider the layout shown in Figure 2. If workstations 1, 4, and 6 form a zone, it would be difficult (or impractical) to assign workstations 3 and 8 to other zones. (The guidepaths of two zones may not cross each other.) Generally speaking, one must avoid generating potential zones which 'isolate' one or more workstations. Also, to avoid unnecessary vehicle travel, the workstations in a zone should be reasonably close to each other. The first phase is further complicated by the fact that, for a given subset of workstations, we do not know where the transfer point(s) are located until we know the configuration of the rest of the system. Obviously, we do not know the configuration of the rest of the system until we solve the partitioning problem in the third phase.

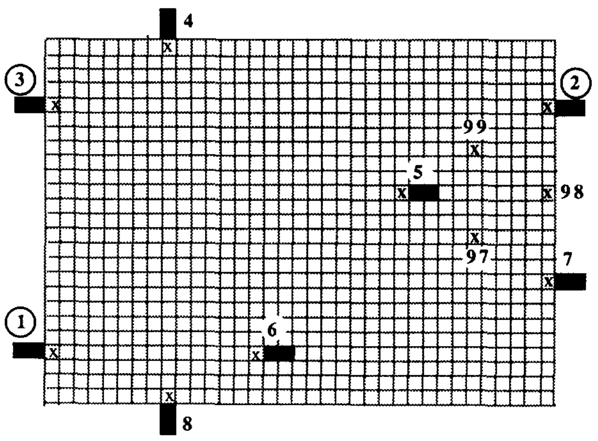


Figure 2. Station locations for layout 1

To circumvent the above difficulties, we developed the following heuristic approach to generate 'promising' subsets of workstations while we concurrently test them for feasibility. (The details related to feasibility checking will be presented in Section 3.2.) Suppose the total number of workstations in the system to be partitioned is denoted by N. In the following algorithm, we first solve a Traveling Salesman Problem (TSP) over N workstations. We let $p(i)$, $i = 1, 2, \dots, N$, denote the workstation number that corresponds to the ith position on the tour. For example, for $N = 5$, if the TSP tour is given by 3–1–4–5–2–3, then $p(1) = 3$, $p(2) = 1, \dots, p(5) = 2$.

Algorithm I.

Step 0. Read the number of workstations, N. Read the (x, y)-coordinate of each workstation.

Step 1. Solve the *Euclidean* TSP over all the workstations to determine the optimum sequence of workstations. Let $p(i)$ denote the workstation number corresponding to the i-th position in the above sequence.

Step 2. Set $i = 1$, $j = 2$ and $S = \{p(1)\}$.

Step 3. Add $p(j)$ to the set S; i.e., $S \leftarrow S \cup \{p(j)\}$.

Step 4. Solve the *rectilinear* TSP over the workstations in S. Given the optimum sequence of workstations, if the subset defined by the workstations in S is feasible, that is, if the corresponding ω-value for S is less than 1.0, go to Step 5. Otherwise, go to Step 7.

Step 5. Store the subset defined by S as a potential zone (or column) which 'covers' the workstations in S with an objective function coefficient of ω.

Step 6. $j \leftarrow j + 1$. (If $j = N + 1$, set $j = 1$.) Go to Step 3.

Step 7. $i \leftarrow i + 1$. If $i = N + 1$, STOP. Otherwise, set $j = i + 1$. (If $j = N + 1$, set $j = 1$.) Let $S = \{p(i)\}$ and go to Step 3.

In Step 1 of the above algorithm, we use the Euclidean metric to solve the initial TSP since we find it to be a more realistic measure of 'closeness' between the workstations. On the other hand, in Step 4 we solve the rectilinear TSP over the workstations in S since AGV travel is assumed to be better represented by the rectilinear metric.

The above algorithm can be demonstrated by an example. Consider the workstations numbered 1 through 8 in Figure 2. Solving the Euclidean TSP we obtain the optimum tour 1–3–4–5–2–7–6–8–1. We start with the subset given by workstations {1, 2}. Assuming that this subset is feasible, we store it as a column and we subsequently test the subset {1, 3, 4}. Assuming that this subset is also feasible, we store it as a column and we next consider workstations {1, 3, 4, 5}. Assuming that this last subset is not feasible, we 'move' the pointer i to workstation 3 and test the subset {3, 4} for feasibility. Assuming that the subset {3, 4} is feasible, we next check the subset {3, 4, 5}. If the latter is feasible, we next check the subset {3, 4, 5, 2}; otherwise, we check the subset {4, 5}, and so on. (Note that we may consider {8, 1}, {8, 1, 3}, {8, 1, 3, 4}, and so on, as potential zones as well.)

We assume that a subset defined by two adjacent workstations on the Euclidean TSP tour is always a feasible subset. If a particular workstation cannot be feasibly paired with either one of its neighbors, then this workstation can be removed from the partitioning problem. (It can be later added as a single workstation zone after the remaining workstations have been partitioned.) We are also implicitly assuming that the 'subset' defined by all the workstations $(1, 2, \ldots, N)$ cannot be a feasible subset.

The columns generated by Algorithm I for the example problem shown in Figure 2 seemed to provide a sufficient number of 'promising' potential zones. However, in applying the algorithm to larger problems, it became evident that some potentially good zones will not be 'recognized' by Algorithm I. Consequently, we used the 'band' technique to generate additional columns. (The 'band' technique has been used in a variety of sequencing problems, see [4,5], among others.) To apply this technique we first sort all the workstations in a non-decreasing order according to their x-coordinate values. In reference to Figure 2, we would obtain the sequence 1–3–8–4–6–5–7–2. (Ties are broken by selecting the workstation with the smallest y-coordinate value.) We subsequently use the above sequence – instead of the Euclidean TSP tour – in Step 1 of Algorithm I to obtain additional columns. The same procedure is repeated in the y-direction. That is, the workstations are sorted in a non-decreasing order according to their y-coordinate values and the resulting sequence (8–1–6–7–5–3–2–4) is used in Step 1 of Algorithm I. (Ties are broken by selecting the workstation with the smallest x-coordinate value.)

Lastly, we generate more columns by first dividing the workstations (horizontally) between an upper band and a lower band of equal width. Subsequently, we use the above procedure to obtain the sequence 1–8–6–7 for the lower band, and the sequence 3–4–5–2 for the upper band. Likewise, dividing the workstations (vertically) and applying the band technique yields the following two sequences: 8–1–6–3–4 (for the left-hand band) and 7–5–2 (for the right-hand band). Each one of the above four sequences is used, one at a time, in Step 1 of Algorithm I to generate additional columns. Obviously, the value of N must reflect the total number of workstations in a particular sequence.

The TSP tour and alternative sequences ob-

Table 1a
The (x, y)-coordinates of the workstations for layout 1

Workstation	1	2	3	4	5	6	7	8
(x, y)-coordinate	(1, 4)	(35, 21)	(1, 21)	(9, 25)	(25, 15)	(15, 4)	(35, 9)	(9, 1)

tained from the band technique may result in duplicated columns. However, after sorting the workstation numbers within each column, it is straightforward to identify and eliminate duplicate columns. Since such an elimination technique is straightforward to apply, we did not concern ourselves with repetitions during column generation. (More information on the total number of columns, the number of repetitions, and the resulting number of unique columns obtained for the two example problems will be presented in Section 4.) In addition, the above procedure is not guaranteed to generate potential zones that are 'easy' to interface with other zones. However, to a large extent, the TSP tour and the band technique reduce the possibility of generating zones such as the one defined by workstations {1, 4, 6} in Figure 2. As we attempt to demonstrate through the numerical examples, as long as such zones are avoided, interfacing the resulting zones in a tandem system is fairly straightforward.

3.2. *Checking for feasibility*

Consider a subset of workstations obtained from the first phase. To check for feasibility, we have to first identify (approximate) transfer point locations, and define all the appropriate travel times and flow volumes for the subset of workstations in question. Once the above steps are completed, it is straightforward to check for feasibility by computing the value of ω from (4) and (5) where M denotes the number of workstations in the subset.

The approach we used to check for feasibility can perhaps be best described by an example. Recall the workstation locations shown in Figure 2. The (x, y)-coordinate for each workstation is shown in Table 1a. Suppose the AGV system is required to handle four types of jobs (labeled A through D) through the system. The workstations visited by each job type and the number of jobs that must be processed per hour are shown in Table 1b. Note that workstations 1, 2, and 3 are I/O stations while the remaining workstations are processor stations.

Suppose we are given the subset $S = \{5, 2, 7\}$. Recall that, for a given subset of workstations, we solve the *rectilinear* TSP. Since the vehicle is bidirectional, all the distances are symmetric; therefore, for the subset {5, 2, 7}, any tour is optimum. Nevertheless, suppose the optimum tour is given by the sequence 5–2–7–5. We first determine the (approximate) location of each transfer point by computing the centroid of the rectangle formed by two adjacent workstations. As shown in Figure 2, for the subset {5, 2, 7}, we obtain three transfer points: the first one, transfer point 99, is between workstations 5 and 2, and it is located at

$$x = \tfrac{1}{2}(25 + 35) = 30 \text{ and } y = \tfrac{1}{2}(15 + 21) = 18.$$

The second one, transfer point 98, is between workstations 2 and 7, and it is located at $(x, y) = (35, 15)$, and so on. (Due to the rectilinear metric, a transfer point can be actually located anywhere within the rectangle formed by the two adjacent workstations without increasing the travel distance of the vehicle from one workstation to the other. However, since the vehicle is bidirectional, travel to/from a particular transfer point may affect the overall workload on the vehicle. Recall that transfer points are treated as I/O stations.)

Consider now the flow associated with the subset {5, 2, 7}. It is straightforward to transform the data shown in Table 1b (for each job type) to an overall flow matrix for the system as shown in Table 1c. Since the vehicle is assumed to move one job at a time, each entry in the above flow

Table 1b
Workload and routing data for layout 1

Job type	Jobs/hr.	Production routing (by workstation number)
A	1.5	1–4–5–7–1
B	1.5	3–4–6–1
C	3.0	1–7–5–4–2
D	3.0	3–4–5–6–8–1

Table 1c
Overall flow matrix (from-to chart) for layout 1 (given in trips/hour)

	1	2	3	4	5	6	7	8
1	–	0.0	0.0	1.5	0.0	0.0	3.0	0.0
2	0.0	–	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	–	4.5	0.0	0.0	0.0	0.0
4	0.0	3.0	0.0	–	4.5	1.5	0.0	0.0
5	0.0	0.0	0.0	3.0	–	3.0	1.5	0.0
6	1.5	0.0	0.0	0.0	0.0	–	0.0	3.0
7	1.5	0.0	0.0	0.0	3.0	0.0	–	0.0
8	3.0	0.0	0.0	0.0	0.0	0.0	0.0	–

matrix is assumed to represent the number of (loaded) trips the vehicle must perform per hour from one workstation to another. The flow matrix that we need to construct is shown in Table 1d. Consider first the flow among workstations 2, 5, and 7. As shown in Table 1d, this flow data is simply obtained from the flow matrix shown in Table 1c. Consider next the flow that occurs from workstations {1, 3, 4, 6, 8} to workstations {5, 2, 7}. Obviously, this flow must be handled through one of the transfer points (shown in Figure 2). In order to determine the appropriate transfer point, we measure the *Euclidean* distance from a workstation to each transfer point and select the closest one.

For example, out of workstation 1, we have 1.5 jobs/hr flowing to workstation 4 (which we are not concerned with at the moment) and 3 jobs/hr flowing to workstation 7. Since the closest transfer point to workstation 1 is transfer point 97, we assume that all the flow from workstation 1 to workstation 7 is handled through transfer point 97. That is, as far as the AGV which serves the subset {5, 2, 7} is concerned, we have 3 jobs/hr flowing from transfer point 97 to workstation 7 (due to workstation 1 only). Since no flow occurs from workstations 3, 6, and 8 to any one of the workstations in the subset {5, 2, 7}, we disregard these three workstations. However, we have 3 (4.5) jobs/hr flowing from workstation 4 to workstation 2 (5). Since the closest transfer point to workstation 4 is transfer point 99, we have 3 (4.5) jobs/hr flowing from transfer point 99 to workstation 2 (5) (due to workstation 4 only). There are no other jobs that enter the subset {5, 2, 7} through the transfer points.

Lastly, consider the flow from {5, 2, 7} to other workstations. Note that we have 3 (3) jobs/hr flowing from workstation 5 to workstation 4 (6). Since transfer point 99 (97) is the closest one to workstation 4 (6), we have 3 (3) jobs/hr flowing from workstation 5 to transfer point 99 (97). In contrast, no flow occurs out of workstation 2, while 1.5 jobs/hr flow from workstation 7 to workstation 1. This flow must be handled through transfer point 97. Hence, we have 1.5 jobs/hr flowing from workstation 7 to transfer point 97.

The resulting flow data for the subset {5, 2, 7} are shown in Table 1d. Transfer point 98 is never used since it is not the closest transfer point to any of the workstations in the set {1, 3, 4, 6, 8}. Recall that transfer points are assumed to be 'two-way' interface points that handle flow in either direction. When the set partitioning model is solved and the configuration is finalized, the analyst can always replace such 'two-way' transfer

Table 1d
Flow matrix (from-to chart) for the subset {5, 2, 7} (given in trips/hour)

	2	5	7	97	98	99	λ_i	$\Lambda_i - \lambda_i$
2	–	0.0	0.0	0.0	0.0	0.0	0.0	+3.00
5	0.0	–	1.5	3.0	0.0	3.0	7.5	0.00
7	0.0	3.0	–	1.5	0.0	0.0	4.5	0.00
97	0.0	0.0	3.0	–	0.0	0.0	3.0	+1.50
98	0.0	0.0	0.0	0.0	–	0.0	0.0	0.00
99	3.0	4.5	0.0	0.0	0.0	–	7.5	−4.50
Λ_i	3.0	7.5	4.5	4.5	0.0	3.0	22.5	

points with two adjacent 'one-way' transfer points if necessary. (Although it is likely to be negligible, the AGV travel time between two such transfer points can be captured by slightly increasing the load pick up and deposit times at these stations.)

Before we can compute the workload, i.e., the ω-value for the subset {5, 2, 7}, we need to determine the appropriate travel times and the polling sequence for the empty AGV. As remarked earlier, the workstations are sequenced according to the *rectilinear* TSP tour. Adding the transfer points *does not change* the optimum TSP sequence. Assuming that the (empty or loaded) AGV travels at a speed of 15 grids/minute, and that it takes 0.20 minutes to pick up or deposit a load, from Table 1d and equation (3) we obtain $\alpha_f = 0.390$. (The time required to inspect a station is assumed to be negligible.)

We next need to determine the value of ϕ shown in (4) and (5). Due to the $\sigma_{j,i}$-values that appear in (5), the ϕ-value depends on the polling sequence. We consider two polling sequences: clockwise (5–99–2–98–7–97–5) and counterclockwise (5–97–7–98–2–99–5). Obviously, one of the above sequences is obtained directly from the rectilinear TSP tour while the other sequence is obtained by simply reversing it. Using 15 grids/minute for the vehicle speed (to determine the appropriate $\sigma_{j,i}$-values defined in (2)), and the $(\Lambda_j - \lambda_j)$-values shown in Table 1d, from (5) we obtain $\phi = 0.1467$ and $\phi = 0.0733$ for clockwise and counterclockwise polling sequences, respectively. Hence, assuming a counterclockwise polling sequence, from (4) we obtain $\omega = 0.4633$. Since ω is less than 1.0, the subset defined by the workstations 2, 5, and 7 is considered a feasible zone. Therefore, we store it as a column which 'covers' workstations 2, 5, and 7, at a 'cost' of 0.4633 units. (Note that, once the ω-value is computed, we need not include the transfer points in the definition of a column.)

Recall that, in Step 4 of Algorithm I, a prospective zone is considered feasible if its ω-value is less than 1.0. Given the approximate nature of the above proposed scheme, one should use a threshold value less than 1.0. (The partitioning scheme developed here is an approximate one since the exact locations of the transfer points are not known in advance and we do not consider possible 'transit loads' in computing the workload. Our numerical experiments indicate that, as long as the sequence of workstations remains unchanged, the workload is generally not sensitive to the exact location of a transfer point.) We suggest using threshold values between 0.70 and 0.80; i.e., any subset with an ω-value greater than or equal to 0.70 (or 0.80) should be considered infeasible. This would facilitate the interfacing task by allowing the analyst some flexibility in modifying the shape/size of the zones and the exact location of the transfer points in the final partition.

3.3. *The partitioning problem*

The partitioning model is based on a variation of the well-known set partitioning problem. Let ω_p denote the cost coefficient (i.e., the workload factor) of the p-th (unique) column obtained from the previous phase. Also, let a_{ip} equal to 1 if workstation i is 'covered' by column p, and 0 otherwise. Lastly, let L denote the desired number of zones set by the analyst. Of course, L also denotes the number of vehicles.

The decision variable is denoted by x_p, where x_p is equal to 1 if column p is used in the final partition, and 0 otherwise. The objective is to avoid generating bottleneck loops by evenly distributing the overall workload among the loops as much as possible. An indirect way to accomplish this is to minimize the maximum workload in the system. Assuming that the maximum workload (which occurs in one or more zones) is designated by z, we obtain the following 0–1 Integer Programming problem:

Minimize z

subject to

$$z - \omega_p x_p \geq 0 \quad \text{for all } p, \tag{6}$$

$$\sum_p a_{ip} x_p = 1 \quad \text{for all } i, \tag{7}$$

$$\sum_p x_p = L, \tag{8}$$

$$x_p = 0 \text{ or } 1 \quad \text{for all } p \tag{9}$$

where constraint (6) ensures that the workload in any zone used in the final partition does not exceed z. Constraint (7) requires that each workstation is assigned only to one zone. Lastly, constraint (8) forces the resulting partition to have exactly L zones.

Clearly, if L is not sufficiently large, the above model may not have a feasible solution. Depending on the problem size – primarily the number of columns – and the computational resources available, one may solve the above model for several L-values to generate alternative solutions. Although for relatively small problems this may be the most desirable approach from a design standpoint, it may not be a practical approach for large problems from a computational standpoint. (The definitions of 'small' and 'large' problems depend on the type of hardware and software available to solve the above model.) For large problems, it may be more practical to start with a relatively large value of L and gradually decrease it either until the maximum ω-value, i.e., z, exceeds a user-defined upper limit, or until the problem becomes infeasible. Since we assumed that a workstation can always be feasibly paired with an adjacent workstation on the Euclidean TSP tour, an upper bound on L is given by $\lceil \frac{1}{2}N \rceil$, where N is the total number of workstations to be partitioned and $\lceil \bullet \rceil$ denotes the ceiling function.

Also, heuristic methods can be used to estimate a near-minimum L-value. For example, we may sort the columns in a non-increasing order based on the number of workstations they cover. Starting with the column which covers the maximum number of workstations, we may select each column, one at a time, until all the workstations are covered. To avoid 'double coverage', as soon as we select a column, we disallow all the other columns that cover any of the workstations already covered by the selected column.

Other methods can be devised to generate appropriate L-values. However, it seems that in practice, the analyst can narrow down the range of reasonable L-values rather quickly. For example, if $N = 20$, then setting L equal to 10 will generate too many zones that are in all likelihood not necessary. On the other hand, setting L equal to, say, 4, will require that a zone, on the average, cover 5 workstations. By examining the above sorted list of columns and the corresponding ω-values, the analyst can quickly determine whether such an 'average' is reasonable or not. Hence, for most cases it seems that, even for moderately large problems with $N \approx 20$, it will be fairly straightforward to reduce the L-values of interest to a small range. In this study, since we are also interested in comparing the performance of tandem AGV systems with conventional AGV systems, the number of vehicles required for the conventional system was used as a benchmark value for L. (As described in Section 4, the number of AGVs required for the conventional system was obtained via simulation.)

3.4. Solving the partitioning problem

We used a generic 0–1 Integer Programming code (LINDO) on a mainframe to solve the partitioning problem defined in Section 3.3. As shown in Section 4, a fair number of (unique) columns are generated for both example problems. It is instructive to note that one may eliminate a number of columns if a heuristic solution is available. (Note that classical set partitioning heuristics will not directly apply here since the desired number of zones, that is, L, is selected a priori by the user.) Suppose the maximum workload obtained from a heuristic solution is given by z_H. Any column which has an ω-value greater than z_H cannot be in the optimum solution (due to the minimax nature of the problem). Hence, to reduce the computational effort, all such columns can be eliminated.

Developing a heuristic procedure to solve the partitioning problem is beyond the scope of this study. However, using a 'quick-and-dirty' method (based on the Euclidean TSP tour), we were able to obtain a reasonably good solution for the second example problem. (We did not have to apply the heuristic to the first example problem since we were able to solve it without eliminating any columns.) Obviously, one would eliminate many columns with a good heuristic. However, if a good heuristic procedure were available, there would be less incentive to obtain an exact solution.

In terms of solving the partitioning problem, there is a special case which requires further explanation. Recall that we assumed a workstation can be feasibly paired with at least one of its neighbors on the Euclidean TSP tour. For an odd number of workstations, this does not necessarily guarantee a feasible partition even if L is sufficiently large. For example, for $N = 5$, if the TSP tour is given by the sequence 1–2–3–4–5–1, and the *only* feasible subsets are given by {1, 2}, {2, 3}, {3, 4}, {4, 5}, and {5, 1}, then no feasible solution exists to the partitioning problem. In this case, one may add N single-workstation columns. Letting s denote the set of all *single-workstation* columns and Q denote the maximum number of single-workstation zones allowed in the final partition, the workload for each single-station column can be simply set equal to zero, as long as we add the constraint

$$\sum_{p \in s} x_p \leq Q$$

to the model given by (6)–(9).

3.5. *Summary of the overall approach*

The heuristic procedure to obtain a partition for tandem AGV systems can be summarized as follows:

1. Use Algorithm I to generate *subsets of workstations* from the *sequence of workstations* which are obtained from: (a) the Euclidean TSP tour, (b) the single-band technique applied in the x- and y-directions, and (c) the two-band technique applied in the x- and y-directions.
2. Check each subset for feasibility as they are generated. If a subset is feasible, store it as a potential zone (i.e., column) with the objective function coefficient set equal to the workload in the zone, i.e., ω_p.
3. Scan the list of all potential zones and eliminate duplicates. Solve the resulting minimax set partitioning problem by Integer Programming.

In the next section we will demonstrate the above procedure.

4. Numerical examples and conventional AGV systems

In this section we will use two numerical examples to demonstrate the partitioning algorithm and to compare the performance of the resulting tandem configurations with their conventional counterparts. All the assumptions and definitions stated for tandem AGV systems also apply to conventional AGV systems, with the following exceptions:

1. Conventional AGVs are assumed to operate on a unidirectional basis. Using bidirectional vehicles in conventional AGV systems further complicates the control system, and it potentially creates a significant increase in path contention.
2. The conventional AGV system is assumed to have only I/O and processor stations; that is, no transfer points are needed.
3. The empty vehicles in the conventional system are assumed to be dispatched according to the Shortest-Travel-Time-First (STTF) rule. Under this rule, if a vehicle becomes empty at its last delivery point, it is dispatched to the closest workstation which contains an unassigned move request.
4. If a vehicle becomes empty at its last delivery point and no unassigned move requests are available in the system, the empty vehicle is sent to a parking area (so that it will not block other vehicles in the system). If a new move request arrives while the empty vehicle is traveling to the parking area, the vehicle is assigned to that move request and rerouted accordingly.
5. Under the STTF rule, a vehicle will be in one of three possible states: traveling loaded, traveling empty, or idling at the parking area. The expected vehicle utilization, say, ρ, is given by $\alpha_f + \alpha_e$, where α_f and α_e designate the expected proportion of time the vehicle is traveling loaded and traveling empty, respectively.

We used a commercial simulation/animation program (WITNESS) to simulate both the tandem and the conventional system. This program captures all vehicle interactions (such as blocking at intersection points and P/D points) as well as delays caused by 'zone blocking' in conventional AGV systems. Each simulation run was first warmed up for 10000 minutes followed by 5 replications that were 6000 minutes long each. In all the simulation runs, to ensure that they do not become a bottleneck, the average processing time for each processor was set equal to that value which yields an expected processor utilization of 75%. Also, the (empty or loaded) vehicle is assumed to travel at a speed of 15 grids/minute, and it takes 0.20 minutes to pick-up or deposit a load.

The data for the first numerical example (that is, layout 1) were given earlier in Table 1 and Figure 2. The configuration used for the conventional system is shown in Figure 3a. Using the simulation model, we observed for the conventional system that the workload given in Table 1b can be satisfied with 4 vehicles that are, on the average, approximately 75% utilized (i.e., $\rho = \alpha_f + \alpha_e \approx 0.75$).

For the tandem configuration, assuming that a prospective zone is feasible only if its estimated workload does not exceed 0.90, the column generation algorithm generated a total of 69 columns with layout 1. After eliminating duplicate columns, there were 39 (unique) columns remaining for the partitioning model. Since this number is well within the range of problems we can solve exactly, we set $L = 4$ and solved the resulting model (without eliminating any columns) to obtain the configuration shown in Figure 3b. The estimated ω-values (obtained from the partitioning algorithm) and the actual ω-values (obtained from (3)–(5) for the zones shown in Figure 3b) are shown in Table 2. As one would anticipate, the actual values exceed the estimated values since each loop was expanded in order to reduce the length of the interface conveyors. There are no transit loads in the tandem configuration shown in Figure 3b.

We simulated the conventional system and the tandem system under three throughput levels, say, 'low', 'medium', and 'high'. The first level (i.e., the low level) is given in Table 1b where we have 1.5 jobs/hr (or 40 minutes between job arrivals) for job types A and B, and 3.0 jobs/hr (or 20 minutes between job arrivals) for job types C and D. The medium throughput level was obtained by increasing the arrival rate of all job types by 33.33%. The high throughput level was obtained by increasing the arrival rate of all job types by 60% relative to the low throughput level. The results obtained from the simulation model – except for the $(\alpha_f + \phi)$-values, which were obtained analytically – for the *low* and *high* throughput levels are presented in Tables 3a and 3b, where 'TIS' (time-in-system) represents the average time a job spends in the system, including the processing times. Both the tandem system and the conventional system meet the required throughput at both throughput levels. (At the medium throughput level, the $(\alpha_f + \phi)$-values for the tandem system range from 44.93 (for AGV1) to 65.73 (for AGV2), and the average vehicle utilization for the conventional system is equal to 87.38%.)

We must stress that, although the $(\alpha_f + \phi)$-values and the ρ-values shown in Table 3 reflect the workload on each vehicle under the tandem (FEFS) and conventional system, respectively, their direct comparison will lead to the incorrect conclusion that the tandem system has more slack. This is primarily because, as the throughput level is increased, the $(\alpha_f + \phi)$-values for the tandem system will increase proportionally. In contrast, for the conventional system, only loaded vehicle travel will increase proportionally with the throughput level, whereas vehicle utilization due to empty travel, that is, α_e, will generally decrease as the throughput is increased. (Under the STTF rule, as the throughput is increased, an arriving loaded vehicle is more likely to find a new load at its delivery point.)

The above observation can be made in Table 3 where the $(\alpha_f + \phi)$-values for the tandem system increase at the same rate as the throughput level, while in the conventional system the ρ-values increase at a slower rate. Also note that, as the throughput level is increased, the average and maximum queue lengths for both systems increase. However, the conventional AGV system appears to perform slightly better. In Table 3, although the tandem (FEFS) system 'theoretically' still meets the required throughput at the highest throughput level, the maximum queue lengths are approaching unacceptable values for most practical applications. In fact, if the throughput level is increased by 100% (relative to the low level), the conventional system still meets the required throughput with an average vehicle utilization of approximately 98% and an overall maximum queue length of 24 jobs (at the output queue of workstation 3). For the tandem system, on the other hand, we were not able to obtain a result with the above throughput level since the number of 'active' jobs in the system reached an

Table 2
Estimated and resulting workloads in each zone

Zone	Layout 1[a]		Layout 2	
	Estimate	Result	Estimate	Result
1	0.218	0.337	0.292	0.307
2	0.300	0.493	0.307	0.297
3	0.333	0.453	0.228	0.297
4	0.370	0.483	0.293	0.260
5	n/a	n/a	0.284	0.313
6	n/a	n/a	0.248	0.243

[a] In layout 1, the resulting workload in each zone should exceed our estimate since we 'stretched' each loop after obtaining the final partition. Although it was not necessary, we stretched each loop in order to reduce the length of the interface conveyors.

upper limit imposed by the software. When the simulation model stopped due to this upper limit, there were 375 jobs in the output queue of workstation 3.

To provide a more direct comparison between the two systems we also simulated the tandem system with the STTF rule. The results are shown in Table 3. As anticipated, using the STTF rule in place of the FEFS rule, improves the performance of the tandem system. This is perhaps due to the fact that, under FEFS, the vehicle always inspects the workstations in a unidirectional fashion, whereas, with STTF, the empty vehicle is allowed to travel in a bidirectional fashion to reach the closest unassigned move request. Using the ρ-values shown for both the tandem and the conventional system in Table 3, one can now directly compare the workload imposed on the vehicles. At the low throughput level, the maximum ρ-value for the tandem system is equal to 0.7169 (due to AGV4). This value is less than 0.75, which is the average vehicle utilization for the conventional system. However, at the high throughput level, the maximum ρ-value for the tandem system is equal to 0.9654, which exceeds the overall vehicle utilization of 0.9357 in the conventional system. With the medium throughput level (which is not shown), the maximum ρ-value for the tandem system is equal to 0.8822 (due to AGV4) while it is equal to 0.8738 for the conventional system.

From Table 3 one can also observe that, at both throughput levels, the average time a job spends in the system (i.e., 'TIS') is longer for the tandem configuration. With 4 vehicles, very limited blocking and congestion occurs in the conventional system, while additional delays are induced at the transfer points in the tandem sys-

Table 3a
Simulation results for layout 1. Interarrival times [a] = 40, 40, 20, 20

Output buffers	Tandem (FEFS)			Tandem (STTF)			Conventional (STTF)		
	Average queue size	1/2-width 95% CI	Max queue size	Average queue size	1/2-width 95% CI	Max queue size	Average queue size	1/2-width 95% CI	Max queue size
1	0.15	0.01	4	0.09	0.02	3	0.24	0.01	4
2	0.00	0.00	0	0.00	0.00	0	0.00	0.00	0
3	0.42	0.12	10	0.24	0.07	6	0.28	0.02	5
4	0.54	0.04	8	0.23	0.04	5	0.58	0.01	7
5	0.41	0.03	6	0.19	0.03	4	0.39	0.01	5
6	0.39	0.03	5	0.23	0.02	4	0.22	0.02	4
7	0.45	0.05	7	0.28	0.05	4	0.27	0.01	4
8	0.12	0.01	4	0.07	0.01	3	0.15	0.01	3
Conveyors									
C1	0.07	0.03	3	0.07	0.01	2	*	*	*
C2	0.16	0.01	3	0.11	0.02	3	*	*	*
C3	0.15	0.01	2	0.15	0.02	3	*	*	*
C4	0.12	0.02	5	0.08	0.02	2	*	*	*
C5	0.12	0.01	4	0.08	0.02	3	*	*	*
C6	0.26	0.02	4	0.24	0.03	4	*	*	*
C7	0.46	0.08	6	0.25	0.07	4	*	*	*
C8	0.15	0.01	5	0.10	0.02	3	*	*	*

TIS	Average	1/2-width 95% CI	Average	1/2-width 95% CI	Average	1/2-width 95% CI
	137.27	22.46	131.33	26.77	127.70	19.70

	α_f	$\frac{1}{2}$-w	$\alpha_f + \varphi$	α_f	$\frac{1}{2}$-w	ρ	$\frac{1}{2}$-w	α_f	$\frac{1}{2}$-w	ρ	$\frac{1}{2}$-w
AGV1	30.39	0.84	33.70	31.51	1.38	46.16	1.86	37.35	0.63	74.71	0.50
AGV2	41.06	2.05	49.30	40.54	2.18	59.57	3.04	37.17	0.80	74.80	1.52
AGV3	35.86	1.42	45.30	38.51	1.97	58.27	2.75	37.68	0.78	74.97	1.38
AGV4	44.82	1.13	48.30	45.97	2.91	71.69	3.59	37.24	1.50	74.50	1.91

[a] Interarrival times for job types A, B, C, and D (minutes/job).

tem. Also note that, for both systems, the average time a job spends in the system *decreases* when the throughput level is increased. The decrease observed in TIS is due to shorter processing times at the processor stations. (Recall that, in order to maintain an expected processor utilization of 75% in all the simulation runs, we increase the processor capacity as we increase the throughput level.) The average time a job spends in the system may not always decrease as the throughput (and the processor speeds) are increased. In some cases, when throughput is increased, TIS may also increase. This is due to the fact that, as the throughput level is increased, the total time a job spends in the system becomes dominated by the time it spends waiting for a vehicle. That is, reductions obtained in TIS due to faster processors are offset by long waiting times at the output buffers or transfer points.

For the second example problem (i.e., layout 2) we used six job types and 20 workstations. The data for each job type are shown in Table 4. The configuration of the conventional AGV system is shown in Figure 4a. The direction of each arc in the conventional system was determined by a simple heuristic procedure aimed at minimizing loaded vehicle travel. For the workload given in Table 4 (that is, 1.5 jobs/hr of each job type), the simulation model for the conventional system indicates that 8 vehicles are required to meet throughput. The average AGV utilization is equal to approximately 84%.

For the tandem configuration, assuming that a prospective zone is feasible only if its estimated workload does not exceed 0,75, the column generation algorithm generated a total of 417 columns. (This operation required approximately 17.5 minutes on a 25 MHz 386 personal computer

Table 3b
Simulation results for layout 1. Interarrival times [a] = 25, 25, 12.5, 12.5

Output buffers	Tandem (FEFS)			Tandem (STTF)			Conventional (STTF)		
	Average queue size	1/2-width 95% CI	Max queue size	Average queue size	1/2-width 95% CI	Max queue size	Average queue size	1/2-width 95% CI	Max queue size
1	0.34	0.05	8	0.23	0.06	5	0.44	0.02	8
2	0.00	0.00	0	0.00	0.00	0	0.00	0.00	0
3	2.35	0.87	24	1.91	0.87	14	0.71	0.14	14
4	2.00	0.37	19	0.69	0.11	7	1.09	0.08	13
5	1.45	0.36	14	0.72	0.12	8	0.83	0.08	9
6	1.58	0.29	12	0.64	0.08	6	0.49	0.04	6
7	1.86	0.52	16	1.44	0.44	10	0.52	0.03	6
8	0.28	0.03	5	0.18	0.02	4	0.37	0.06	8
Conveyors									
C1	0.17	0.08	6	0.40	0.15	4	*	*	*
C2	0.37	0.04	5	0.30	0.09	4	*	*	*
C3	0.27	0.02	3	0.37	0.05	5	*	*	*
C4	0.42	0.08	6	0.36	0.14	4	*	*	*
C5	0.28	0.06	5	0.22	0.05	3	*	*	*
C6	0.84	0.15	5	1.03	0.26	11	*	*	*
C7	1.77	0.30	6	1.26	0.44	9	*	*	*
C8	0.45	0.12	6	0.25	0.05	4	*	*	*
TIS	Average	1/2-width 95% CI		Average	1/2-width 95% CI		Average	1/2-width 95% CI	
	133.70	23.42		117.36	20.53		88.27	8.84	

	α_f	$\frac{1}{2}$-w	$\alpha_f+\varphi$	α_f	$\frac{1}{2}$-w	ρ	$\frac{1}{2}$-w	α_f	$\frac{1}{2}$-w	ρ	$\frac{1}{2}$-w
AGV1	48.86	1.76	53.92	50.46	1.73	70.24	1.87	55.55	2.05	93.65	1.98
AGV2	63.70	2.85	78.88	64.89	2.49	90.97	2.79	55.33	1.85	93.57	1.92
AGV3	59.24	3.48	72.48	61.65	2.23	86.93	2.82	55.44	2.28	93.51	2.05
AGV4	71.96	2.76	77.28	73.68	3.28	96.54	1.74	55.42	2.47	93.56	1.57

[a] Interarrival times for job types A, B, C, and D (minutes/job).

equipped with a 387 coprocessor.) After eliminating duplicate columns (which required only 11 seconds), there were 325 (unique) columns remaining for the partitioning model. Given the fairly large problem size, we used the heuristic mentioned in Section 3.4 and found a 6-zone partition with an objective function value of $z = 0.428$. Eliminating each column that had an ω-value greater than 0.428, we were able to reduce the number of columns to 153. (Note that the above elimination does not exclude any column which may be in the optimum solution for $L = 6$.) Solving the reduced problem with $L = 6$, we obtained the optimum configuration shown in Figure 4b for which $z^* = 0.307$ (The reduced problem was solved by LINDO on an IBM 3090/600E mainframe in 4.38 minutes.)

The estimated ω-values (obtained from the partitioning algorithm) and the actual ω-values (obtained from (3)–(5) for the zones shown in Figure 4b) are shown in Table 2. The maximum

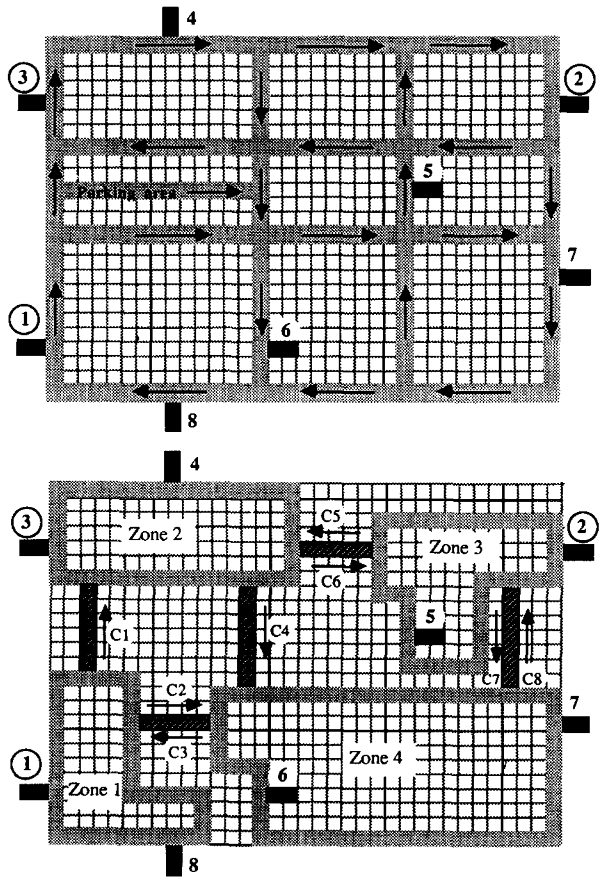


Figure 3. (a) Conventional AGV system, layout 1, (b) Tandem AGV system, layout 1

estimated ω-value is equal to 0.409 (due to zone 2) while the maximum actual workload in the system is equal to 0.313 (due to zone 5). Note that the difference between the maximum and minimum estimated ω-values is fairly small. The same holds true for the actual ω-values.

We simulated the conventional system (with 8 vehicles) and the tandem (FEFS) system (with 6 vehicles) under three throughput levels as before. The low level is given in Table 4 where we have 40 minutes between job arrivals for each job type. The medium throughput level was obtained by increasing the arrival rate of all job types by 33.33%, and the high throughput level was obtained by increasing the arrival rate of all job types by 60% (relative to the low throughput level). The results obtained from the simulation model – except for the $(\alpha_f + \phi)$-values, which were obtained analytically – for the low and high throughput levels are presented in Tables 5a and 5b. Both the tandem (FEFS) system and the conventional system meet the required throughput at the low level. (The same holds true for the medium throughput level which is not shown.) However, judging by the expected and maximum queue lengths shown in Table 5, when throughput is increased to the high level, the conventional AGV system clearly begins to stall. (Recall that the $(\alpha_f + \phi)$-values and the ρ-values shown in Table 5 are not directly comparable.)

In fact, if the throughput level is increased by 100% (relative to the low level), the tandem system still meets the required throughput with a maximum $(\alpha_f + \phi)$-value of 0.6260 (in zone 5) and an overall maximum queue length of 16 jobs (at the output queue of workstation 14) although it has 6 vehicles instead of 8. For the conventional system, on the other hand, we were not able to obtain a result with the above throughput since the number of 'active' jobs in the system reached an upper limit imposed by the software. (When the simulation model stopped due to this upper limit, there were 775 jobs in the output queue of workstation 18.)

As before, we also simulated the tandem system under the STTF rule. The results are shown in Table 5. Once again, the STTF rule improves the performance of the tandem system. At the low throughput level, the maximum ρ-value for the tandem system is equal to 0.5174 (in zone 2) while the average vehicle utilization in the con-

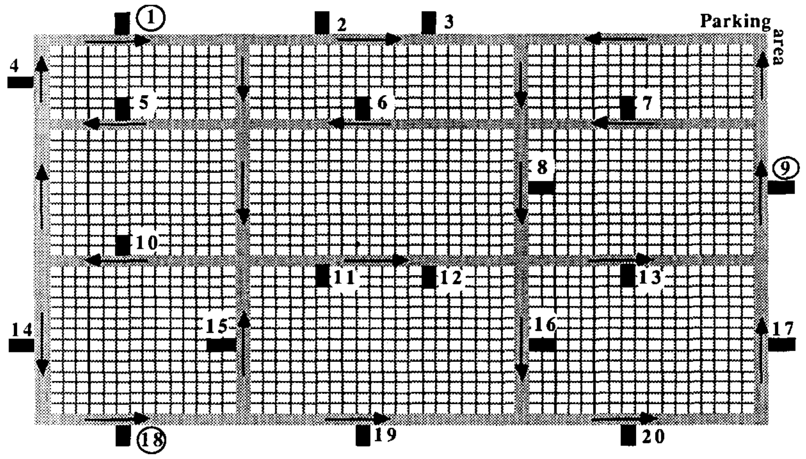
Table 4
Workload and routing data for layout 2

Job type	Jobs/hr.	Production routing (by workstation number)
A	1.5	1–3–6–2–5–4–1
B	1.5	1–6–8–7–9
C	1.5	9–7–8–16–20–17–13–9
D	1.5	18–15–11–12–16–13–17–9
E	1.5	18–15–19–12–11–10–14–18
F	1.5	18–14–10–4–5–1

ventional system is equal to approximately 0.84. For the high throughput level, the maximum ρ-value for the tandem system is equal to 0.7737 as opposed to an average vehicle utilization of almost 1.0 for the conventional system. Also note that, contrary to what we had observed for layout 1, in layout 2 the average time a job spends in the system (TIS) is generally longer for the conventional system at both throughput levels (see Table 5.) For the conventional system with 8 vehicles and 20 workstations, it is clear that blocking, congestion, and unidirectional travel are becoming critical factors. (At the low throughput level, the difference between the TIS values are fairly small because the time-in-system is dominated by the processing times and the waiting times at the *input* buffers.)

5. Conclusions and future research

In this paper we presented a partitioning algorithm based on a specialized column generation technique. The resulting partitioning model is a variation of the well-known set partitioning prob-



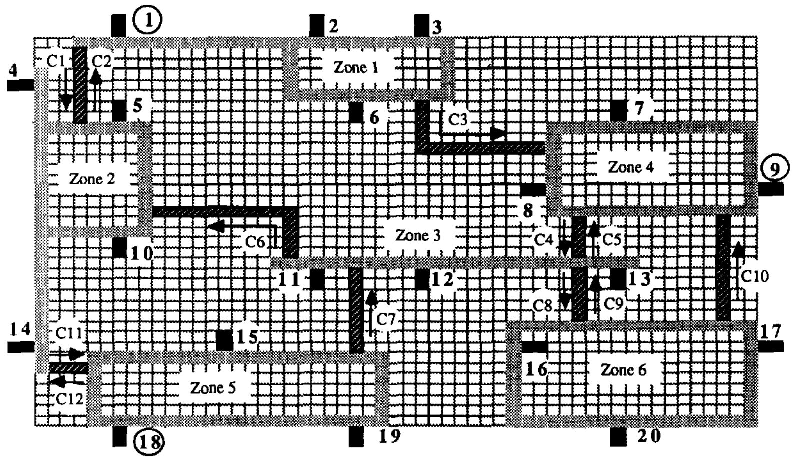


Figure 4. (a) Conventional AGV system, layout 2, (b) Tandem AGV system, layout 2

lem. Our empirical results indicate that reasonably good partitions can be obtained for tandem AGV systems using the above algorithm. Also, our simulation results indicate that, from a throughput standpoint, tandem AGV systems are very competitive with conventional AGV systems.

Table 5a
Simulation results for layout 2. Interarrival times = 40 minutes/job for all job types

Output buffers	Tandem (FEFS)			Tandem (STTF)				Conventional (STTF)			
	Average queue size	1/2-width 95% CI	Max queue size	Average queue size	1/2-width 95% CI	Max queue size		Average queue size	1/2-width 95% CI	Max queue size	
1	0.24	0.03	6	0.09	0.01	3		0.28	0.03	7	
2	0.09	0.01	3	0.04	0.01	4		0.17	0.01	3	
3	0.12	0.02	3	0.04	0.01	2		0.18	0.02	4	
4	0.20	0.01	5	0.09	0.02	3		0.26	0.01	4	
5	0.15	0.01	4	0.08	0.01	3		0.26	0.02	4	
6	0.17	0.02	4	0.08	0.02	3		0.24	0.04	4	
7	0.13	0.02	4	0.07	0.01	4		0.24	0.02	3	
8	0.13	0.01	3	0.06	0.01	3		0.21	0.02	4	
9	0.05	0.00	3	0.03	0.01	2		0.09	0.00	3	
10	0.18	0.02	3	0.09	0.02	3		0.29	0.03	5	
11	0.15	0.02	4	0.06	0.01	3		0.26	0.04	4	
12	0.12	0.01	5	0.05	0.00	3		0.27	0.03	4	
13	0.17	0.02	5	0.05	0.01	3		0.20	0.02	4	
14	0.22	0.04	4	0.12	0.02	4		0.33	0.04	5	
15	0.11	0.02	5	0.06	0.01	3		0.44	0.06	6	
16	0.12	0.01	3	0.08	0.01	3		0.20	0.02	4	
17	0.12	0.01	5	0.07	0.01	3		0.26	0.02	5	
18	0.22	0.04	5	0.13	0.02	3		0.58	0.07	6	
19	0.09	0.02	3	0.04	0.01	2		0.24	0.04	4	
20	0.06	0.01	3	0.05	0.01	2		0.12	0.01	3	
Conveyors											
C1	0.09	0.01	3	0.04	0.01	2		*	*	*	
C2	0.25	0.03	4	0.11	0.02	4		*	*	*	
C3	0.08	0.01	3	0.04	0.01	2		*	*	*	
C4	0.07	0.01	3	0.03	0.01	2		*	*	*	
C5	0.06	0.01	2	0.03	0.00	2		*	*	*	
C6	0.08	0.01	2	0.04	0.01	2		*	*	*	
C7	0.16	0.02	4	0.06	0.01	3		*	*	*	
C8	0.16	0.01	4	0.10	0.02	3		*	*	*	
C9	0.14	0.02	4	0.04	0.01	2		*	*	*	
C10	0.05	0.01	2	0.04	0.01	2		*	*	*	
C11	0.06	0.01	3	0.04	0.01	3		*	*	*	
C12	0.10	0.01	3	0.06	0.02	3		*	*	*	
TIS	Average	1/2-width 95% CI		Average	1/2-width 95% CI			Average	1/2-width 95% CI		
	360.26	53.03		379.98	39.26			378.63	53.66		
	α_f	$\frac{1}{2}$-w	$\alpha_f + \varphi$	α_f	$\frac{1}{2}$-w	ρ	$\frac{1}{2}$-w	α_f	$\frac{1}{2}$-w	ρ	$\frac{1}{2}$-w
AGV1	27.35	1.88	30.70	24.93	2.29	43.44	3.70	37.21	1.71	84.43	3.83
AGV2	27.91	1.53	29.70	27.05	2.63	51.74	4.57	36.22	2.64	84.75	3.16
AGV3	24.96	1.30	29.70	18.88	1.29	36.27	2.49	36.90	2.35	84.60	3.10
AGV4	23.02	1.55	26.00	23.36	1.54	38.59	2.64	36.78	2.23	84.59	3.30
AGV5	23.15	2.32	31.30	21.96	1.84	37.30	3.17	36.79	2.86	85.00	3.33
AGV6	21.46	1.11	24.30	23.93	2.15	39.04	3.67	36.53	2.75	84.50	3.33
AGV7	*			*		*		36.25	2.73	84.47	3.94
AGV8	*			*		*		36.64	2.28	84.80	3.26

It seems that, in terms of throughput performance, a conventional AGV system with only three or four vehicles would be comparable to or better than a tandem AGV system. However, unless very simple, tailored control routines are developed for small conventional AGV systems,

Table 5b
Simulation results for layout 2. Interarrival times = 25 minutes/job for all job types

Output buffers	Tandem (FEFS)			Tandem (STTF)			Conventional (STTF)		
	Average queue size	1/2-width 95% CI	Max queue size	Average queue size	1/2-width 95% CI	Max queue size	Average queue size	1/2-width 95% CI	Max queue size
1	0.59	0.14	7	0.24	0.06	4	0.66	0.14	9
2	0.19	0.03	5	0.08	0.02	4	0.40	0.08	5
3	0.30	0.08	6	0.10	0.03	4	0.39	0.02	6
4	0.48	0.04	6	0.23	0.07	4	0.50	0.05	6
5	0.29	0.01	4	0.18	0.03	4	0.53	0.05	6
6	0.33	0.02	5	0.17	0.04	4	0.47	0.02	6
7	0.28	0.03	5	0.16	0.03	5	0.39	0.02	5
8	0.29	0.03	6	0.15	0.02	4	0.39	0.03	5
9	0.11	0.01	4	0.07	0.02	2	0.17	0.02	3
10	0.42	0.04	8	0.21	0.06	4	0.77	0.09	7
11	0.36	0.05	6	0.12	0.02	4	0.63	0.05	7
12	0.26	0.02	4	0.10	0.02	3	0.65	0.05	6
13	0.53	0.09	7	0.11	0.01	3	0.41	0.03	5
14	0.54	0.02	8	0.32	0.09	4	0.91	0.17	10
15	0.20	0.01	5	0.15	0.04	4	3.45	1.62	17
16	0.27	0.03	5	0.18	0.04	4	0.46	0.03	6
17	0.27	0.03	5	0.17	0.02	3	0.55	0.06	7
18	0.53	0.06	6	0.31	0.07	4	6.48	6.27	45
19	0.22	0.02	4	0.09	0.03	3	2.80	0.91	12
20	0.14	0.01	3	0.12	0.03	3	0.27	0.02	5
Conveyors									
C1	0.19	0.02	4	0.09	0.02	2	*	*	*
C2	0.70	0.08	7	0.33	0.07	4	*	*	*
C3	0.18	0.05	4	0.10	0.02	4	*	*	*
C4	0.22	0.02	4	0.06	0.01	3	*	*	*
C5	0.12	0.02	3	0.07	0.02	3	*	*	*
C6	0.18	0.02	4	0.12	0.03	3	*	*	*
C7	0.48	0.08	7	0.13	0.03	4	*	*	*
C8	0.32	0.04	4	0.23	0.05	5	*	*	*
C9	0.33	0.06	5	0.10	0.01	3	*	*	*
C10	0.12	0.03	3	0.09	0.03	3	*	*	*
C11	0.12	0.01	3	0.11	0.03	4	*	*	*
C12	0.23	0.04	3	0.20	0.09	4	*	*	*

TIS	Average	1/2-width 95% CI	Average	1/2-width 95% CI	Average	1/2-width 95% CI
	257.09	15.69	252.57	24.11	316.71	34.07

	α_f	$\frac{1}{2}$-w	$\alpha_f + \varphi$	α_f	$\frac{1}{2}$-w	ρ	$\frac{1}{2}$-w	α_f	$\frac{1}{2}$-w	ρ	$\frac{1}{2}$-w
AGV1	44.14	2.41	49.12	39.96	3.53	66.55	5.31	58.94	1.55	99.88	0.17
AGV2	44.55	0.61	47.52	43.33	4.10	77.37	5.42	58.74	1.92	99.77	0.31
AGV3	42.05	2.32	47.52	30.26	2.14	56.30	3.39	58.96	1.57	99.74	0.37
AGV4	37.16	1.82	41.60	37.36	2.56	59.33	3.51	58.91	2.12	99.79	0.31
AGV5	36.70	1.21	50.08	35.10	3.02	58.55	4.63	59.18	3.15	99.83	0.31
AGV6	35.74	2.23	38.88	38.30	3.28	61.39	5.60	60.00	2.28	99.83	0.23
AGV7	*			*		*		59.09	2.15	99.79	0.38
AGV8	*			*		*		59.01	1.91	99.83	0.14

there will be a sizable 'fixed cost' associated with the control system even if a small number of vehicles are needed. Given their throughput performance, and the additional simplicity and flexibility they offer, the tandem AGV system is emerging as a strong contender for small systems with just three or four vehicles.

For a larger system, our empirical results indicate that the tandem AGV system (with 6 vehicles plus the interface conveyors) performs much better than the conventional AGV system (with 8 vehicles). However, more examples are required to support this observation. The layout and the production routing of the jobs can be expected to have a significant impact on the performance of both conventional and tandem AGV systems. Also, relative to loaded and empty travel times, if the load pick up and deposit times are long, the throughput performance of the tandem system is likely to be adversely effected since a load may be handled several times before it reaches its destination. (With 12 seconds for load pick-up and 12 seconds for load deposit, in layout 2 the tandem system outperformed the conventional system with a wide margin.)

We are in the process of extending this study in several directions. First, we are considering alternative schemes to account for potential "transit loads" in computing the workload for a subset of workstations. Second, we are considering new definitions for a "good partition". For example, in the partitioning algorithm presented here, we were concerned only with the workload generated within a zone, and with minimizing the maximum workload in the system. An alternative approach is to simply minimize the number of zones, subject to the constraint that the workload in any zone may not exceed an upper limit set by the user. (Note that the constraint can be imposed simply by eliminating all columns which have a workload greater than the user defined upper limit.) In the process, one may evaluate the desirability of a prospective zone not only by the workload and the number of workstations it 'covers', but also by the ratio of the flow within the zone to the total flow associated with that zone.

Minimizing zone-to-zone flow can be accomplished more effectively if the layout is not fixed. Although we assumed the layout is fixed, a third and very promising extension to our study is to solve the layout and partitioning problems concurrently. We believe that significant savings can be realized if one is allowed to change the layout (that is, interchange workstation locations) under the tandem approach. Similar savings can be realized for conventional AGV systems as well. However, given its single-vehicle zones and bidirectional travel, it appears that more dramatic savings can be realized with the tandem system if some or all workstation locations are interchangeable.

Acknowledgments

The authors would like to thank Steven J. Erlebacher for his assistance throughout the study. We would also like to thank the anonymous referees for their constructive criticism and suggestions to improve an earlier draft of the paper.

References

[1] Bartholdi, III, J.J., and Platzman, L.K., "Decentralized control of automated guided vehicles on a simple loop", *IIE Transactions* 21/1 (1989) 76–81.

[2] Bozer, Y.A., and Srinivasan, M.M., "Tandem configurations for automated guided vehicle systems and the analysis of single-vehicle loops', *IIE Transactions* 23/1 (1991) 72–82.

[3] Bozer, Y.A., and Srinivasan, M.M., "Tandem configurations for automated guided vehicle systems offer simplicity and flexibility", *Industrial Engineering* 21/2 (1989) 23–27.

[4] Daganzo, C.F., "The length of tours in zones of different shapes", *Transportation Research B* 18/2 (1984) 135–145.

[5] Iri, M., Murota, K., and Matsui, S., "Heuristics for planar minimum-weight perfect matchings", *Networks* 13/1 (1983) 67–92.