CrossMark

ORIGINAL RESEARCH

A new memetic algorithm for mitigating tandem automated guided vehicle system partitioning problem

Parinaz Pourrahimian[1]

Abstract Automated Guided Vehicle System (AGVS) provides the flexibility and automation demanded by Flexible Manufacturing System (FMS). However, with the growing concern on responsible management of resource use, it is crucial to manage these vehicles in an efficient way in order reduces travel time and controls conflicts and congestions. This paper presents the development process of a new Memetic Algorithm (MA) for optimizing partitioning problem of tandem AGVS. MAs employ a Genetic Algorithm (GA), as a global search, and apply a local search to bring the solutions to a local optimum point. A new Tabu Search (TS) has been developed and combined with a GA to refine the newly generated individuals by GA. The aim of the proposed algorithm is to minimize the maximum workload of the system. After all, the performance of the proposed algorithm is evaluated using Matlab. This study also compared the objective function of the proposed MA with GA. The results showed that the TS, as a local search, significantly improves the objective function of the GA for different system sizes with large and small numbers of zone by 1.26 in average.

Keywords AGVS · Tandem configuration · Tabu search · Memetic algorithm · Genetic algorithm

✉ Parinaz Pourrahimian
parinaz.pourrahimian@gmail.com

[1] Department of Mechanical and Manufacturing Engineering, Faculty of Engineering, Universiti Putra Malaysia, Serdang, Malaysia

Introduction

Design of material handling system is among the key decision makings in designing any facility layout. The cost associated with material handling is considerable; estimates average around 20–50% of total operational costs Tompkins et al. (2010). Furthermore, a significant portion of labor cost is associated with material handling, moving or storing Groover (2007).

Literature indicates that materials spent more time either being moved or stored than being processed in manufacturing plants. Since material handling does not add any value to the product, it is desired to minimize the time spent on these activities. With the progress in technology, it is possible to utilize Automated Guided Vehicle System (AGVS) to reduce the time and cost of material handling Merchant (1977).

There are four main possible guide path designs for an AGVS:

(a) conventional/traditional layout;
(b) unidirectional single loop;
(c) segmented flow paths;
(d) tandem topology.

In the conventional configuration, an AGV's guide path is designed throughout the facility layout and meets all of the stations. An AGV is allowed to be routed all over the network. However, the routing, traffic control, and conflict resolution become a serious problem in such configurations. To solve the conflict and dead lock problem in an AGVS, there are generally three approaches: first, to avoid conflicts by special design of AGV guide paths; second, to control the system traffic by dividing the traffic area into several non-overlapping control zones; and third, to use routing and scheduling strategies to prevent deadlocks.

⚘ Springer

Unidirectional single-loop guide path design requires the simplest operational control among other topologies, since all AGVs move in the same direction. However, the drawbacks of this method are first, the blocking problem of AGVs due to stopped vehicles performing pick up or delivery operations and second, the longer distance movement of vehicles for meeting the delivery requirements.

Segmented flow path which is first introduced by Sinriech (1995) consists of one or more segments, each of them divided to non-overlapping zones served by a single AGV. There are transfer buffers at both ends of each zone. In this type of path design, the segments are not necessarily connected.

Tandem AGV system which is the concern of this paper takes the advantage of first solution. Tandem AGV configuration was first presented by Bozer and Srinivasan (1989, 1991). The concept of tandem configuration is to partition all cells of a facility layout into non-overlapping zones to be served by one dedicated vehicle. Figure 1 shows a schematic representation of a typical tandem AGV layout defined by Bozer and Srinivasan (1991). Each workstation in such configuration has an input queue for dropping off, and an output queue for picking up the loads. To make an interface between adjacent zones, there should also be an additional station to transfer the loads. The transfer of jobs between loops may be done by human operator or there may be a conveyor to facilitate the job.

The problem of partitioning of tandem AGVS is to assign a set of N workstations into particular non-overlapping single AGV loops. In addition, there must be extra pick up/drop off (P/D) stations named transfer stations to create a link between two adjacent loops. The interface between transfer points is created either manually or mechanically (using conveyors).

The objective is to assign the workstations to loops in the layout in such a way that the following constraints be provided:
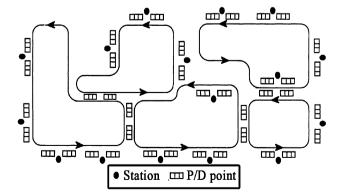
- Each station should be assigned to only one loop.
- The workload of AGVs in the system should not exceed the AGVs' capacity.
- Maximum workload of the system should be minimized.
- There should not be any overlap between adjacent loops.

Bozer and Srinivasan (1989, 1991) presented an analytical model to evaluate the performance of AGVs dedicated to each zone. This analytical model was initially used to define the work load of each zone. Later, Bozer and Srinivasan (1992) proposed a heuristic algorithm based on divide and conquer principle to partition stations in each zone. Hsieh and Sha (1996) introduced the idea of concurrent design of machine layouts and AGV guide paths to minimize the number of loops in tandem layout to minimize the need for transfer the loads between the loops. Huang (1997) designed a new layout in which a transfer center connects all the transfer points in tandem AGV system. Lin et al. (1994) designed a two-phase algorithm to route AGVs in a tandem topology. Later, Liu and Chen (1997) proposed a layout similar to tandem AGV system in which each AGV only serves one zone. In their suggested layout, the zones could have had overlaps with one another. Arab et al. (1999) proposed their algorithm based on the concept of hierarchical clustering and used Tabu Search (TS) as a subroutine. Their algorithm first defines unidirectional single loops on the layout based on geometric shape and then partitions the stations into these loops. Ventura and Lee (2001) evaluated tandem AGV system with the advantage of using more than one AGV for serving each loop and Farling et al. (2001) evaluated tandem layout efficiency in terms of system size, machine failure rate, and unload/load time. Yu and Egbelu (2002) presented a heuristic algorithm to design a tandem layout by having a unidirectional conventional configuration as an input. Their algorithm was based on variable path routing concept and the objective was to define the transfer station's location in design of system. Kim et al. (2003) proposed a design for a tandem AGV system with multi-load AGVs, similar to Bozer and Sirinivasan. They evaluated their model efficiency by comparing it with conventional multi-load system in a simulated environment. BOZER (2004) suggested using one of the existing stations in each zone of tandem layout as transfer station to eliminate the need of using conveyors. Ho and Hsieh (2004) presented a design methodology for tandem multiple load AGV systems. Their objective was to have work load balance, minimize inter-loop flow, and minimize flow distance. Shalaby et al. (2006) presented a 0–1 integer programming model for designing tandem paths which enables users to select one or a combination of objective

Fig. 1 Schematic of a typical tandem AGV system Bozer and Srinivasan (1991)

parameters for: minimizing the total handling cost, minimizing the maximum workload in the system, and minimizing the number of between-zone trips. Laporte et al. (2006) introduced their algorithm based on TS concept. Kim and Chung (2007) proposed a design for tandem AGV system allowing tow AGVs in each zone. Zanjirani Farahani et al. (2007) proposed two partitioning algorithms based on TS and Genetic Algorithm (GA). ElMekkawy and Liu (2009) presented their Memetic Algorithm (MA) which consists of a GA combined with a local search. Salehipour et al. (2010) proposed a heuristic algorithm for portioning of AGVS with objective of minimizing total cumulative flow among workstations. In their work, they tried to minimize the waiting time of workstations to be served by an AGV. In 2011, Rezapour et al. (2011) proposed a method for problem of designing tandem AGV systems with single-load vehicles. They followed three objectives: (1) maximize the workload balance between loops; (2) minimize the inter-loop flow; and (3) minimize the total flow distance. Despite the advantages of using Tandem AGV layout and its vast usage in manufacturing systems and the strength of metaheuristic methods, the literature in this area of study is not sufficient.

In this study, a new TS has been developed and combined with a GA. The performance of proposed MA is also compared with pure GA. In this research, the assumptions for operational conditions of systems are presumed as follows:

- The workstations in system are of two types: input/output (I/O) stations and process stations. Transfer points also defined as I/O stations.
- The system uses single-load vehicles and bidirectional routes.
- Each station should be assigned to only one loop.
- Each loop should have minimum two stations.
- Workload of AGVs is calculated based on both loaded and unloaded vehicle trips.
- When loaded, the vehicles follow the shortest rectilinear route.
- When empty, the AGVs follow the shortest travel time first (STTF) dispatching policy.
- The number of loops is given in the beginning of algorithm.
- Intersection and overlap between zones are not allowed.
- Each loop has one transfer location which is attached to the station that has most transfer with adjacent loops.
- The speed of AGVs is assumed to be steady all the time.

Development of MA

Design and development of the proposed MA are described in this section. Figure 2 illustrates the schematic architecture of proposed MA. The following subsections elaborate on each part of the illustrated architecture in detail.

Initial population

The algorithm begins with clustering workstations into L loops in order to create the first generation of population for GA. Clustering is defined as partitioning a data set composed of n points which are located in m-dimensional space into K sets in a way that each set has the most 'similar' data points. Similarity is defined as the degree of closeness of data sets considering some similarity measures. K means clustering method which is used in this study first introduced by Macqueen (1967) is a common method used for partitioning. The number of clusters is an input to this algorithm. Assume that $D = \{d_i | i = 1, \ldots, n\}$ is the data set with K clusters, $C = \{c_i | i = 1, \ldots, K\}$ is a set of K centers and $S_j = \{d | d$ is member of cluster $k\}$ is the set of instances of the kth cluster. The objective of the k means method is to minimize the value of cost which is defined by

$$\text{Cost} = \sum_{i=1}^{n} \text{dist}(d_i, c_k) \tag{1}$$
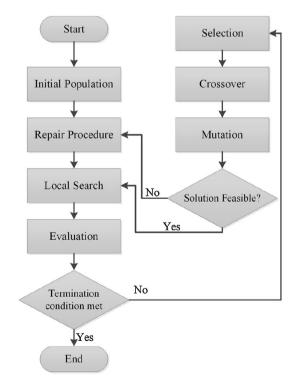

Fig. 2 Flowchart of the proposed MA

where $\mathrm{dist}(d_i, c_k)$ represents the Euclidean distance between d_i and c_k (the cluster center). Cluster centers are defined by the following these steps:

Step 1. Set the centers in c_k using random sampling.
Step 2. Assign patterns (di) to each K clusters based on minimum distance from cluster center criteria.
Step 3. Calculate new c_k using

$$c_k = \frac{\sum_{d_i \in S_k} d_i)}{|S_k|}. \tag{2}$$

Step 4. Repeat steps 2 and 3 until there is not any alteration in cluster centers.

Using this method, a group of workstations are selected randomly as the centers of clusters. This procedure ensures that all the clusters will have at least one station. Then, in the next step, the distance of all unassigned stations to the cluster centers is calculated. Stations will be assigned to the nearest loops and the new cluster centers are defined using Eq. 2.

Employing k means clustering for obtaining first generation of solutions yields some advantages. First, it guarantees prevention of intersections and overlaps between loops due to the fact that the closeness measure is based on the distance of workstations. Second, the workstations assigned to each cluster are logically close to each other, so that it prevents excessive AGV travel within each loop.

Genetic algorithm

Solution representation

Solutions are presented by chromosomes which determine the allocating of stations to partitions. A chromosome is an N-element vector, where N is number of stations. The length of chromosomes which is called index is representative of number of stations in the tandem AGVS. The value of elements which are the position of genes (x_i) represents the loop number that station i is assigned. An example of a chromosome is shown in Fig. 3.

This example indicates that station 1 and 2 are assigned to loop 1, stations 3 and 6 to loop 2, stations 5 and 7 to loop 3, and finally station 4 and 8 to loop 4. This representation guarantees that each work station is assigned to only one loop.

Infeasibility types in generating new solutions

During the process of producing new generations of solutions, three kinds of infeasibility conditions are taken into account, namely:

(a) intersections and overlapping between loops;
(b) empty loops;
(c) singleton loops.

The proposed GA deals with each kind of infeasibilities in different ways using repair operators.

Solutions with overlapping or singleton loops go through a repair procedure, whereas the solutions including empty loops are rejected.

The proposed algorithm deals with the routing problem within loops (connecting stations assigned to each loop) using the nearest neighbor algorithm which is one of the first solutions proposed for traveling salesman problem (TSP). Figure 4 shows the flowchart of the nearest neighbor algorithm.

The loops produced by k means clustering method do not have intersections and overlaps in most cases. However, if any overlap between loops occurs, the proposed algorithm tries to solve it using repair procedure.
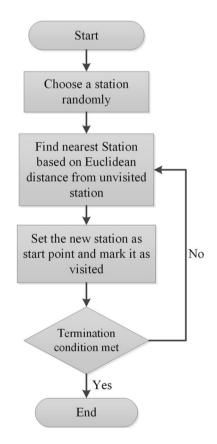
Station number	1	2	3	4	5	6	7	8
Loop number	1	1	2	4	3	2	3	4

Fig. 3 Example of chromosome

Fig. 4 Flowchart of the nearest neighbor algorithm

Repair procedure

The repair procedure goes through the following steps to fix the overlaps between stations of different clusters:

Step 1. Calculate the positions of loop centers (cluster centers).
Step 2. Relocate the stations to loops with the nearest center.
Step 3. Determine new loop centers.
Step 4. End the repair process if the positions of new loop centers are not changed, otherwise go to step 2.

If the number of generated loops is less than the predefined number which was selected at the beginning of algorithm, it shows that some loops do not have any workstations and are empty. In the proposed repair procedure, these solutions are rejected.

Loops having just one station within them are called singleton zones. In case of singleton zones, the repair algorithm tries to fix the problem by following these steps:

Step 1. Determine the nearest station to the singleton zone which belongs to a loop with more than two stations (to avoid a new singleton).
Step 2. Add the station to the singleton zone.
Step 3. Update the solution if there are no overlapping loops, otherwise reject the solution.

Figure 5 depicts the repair procedure for a hypothetic data set. As illustrated in Fig. 5a, there is an intersection between loops 1 and 2. The positions of loop centers are determined as c1 and c2 using the method which was introduced in 2.1.

In Table 1, the rectilinear distances of stations of each loop to the loop centers are presented. Examining the data from Table 1, it is observed that all work stations are closer

Table 1 Distances between stations and loop centers

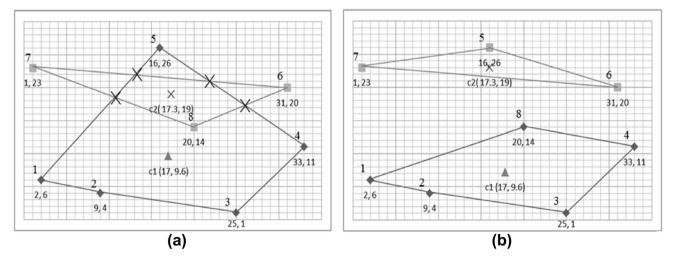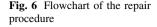| Center	1	2	3	4	5	6	7	8
1	15.43	9.77	11.75	16.06	16.43	20.87	17.44	5.33
2	20.10	17.16	19.56	17.59	7.13	16.82	13.70	5.67

to their own loop center but stations 5 and 8. Hence, these two stations are removed from their loops and added to the neighbor ones. Figure 5b shows the new loop created by repair procedure. Since there are no overlapping loops, the solution is updated.
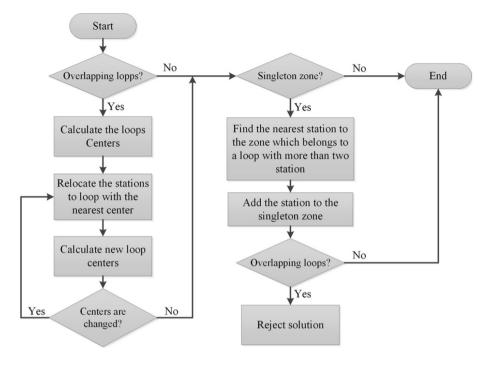
Flowchart of repair procedure is presented in Fig. 6. In case a solution is rejected by repair procedure the k means clustering is applied to reproduce the new generation of solution.

Selection

Selection in GA is the process of choosing particular chromosomes from a population of solutions to produce the next generation of solutions. To apply selection method, a fitness function, which will be described in this section, is evaluated for each loop based on the workload of the loops. Then, the fitness function of chromosomes will be calculated. The fitness functions of chromosomes then should be normalized. Normalization means to divide the fitness value of each chromosome by the sum of fitness values of all individuals in the solution space, so that the sum of all resulting fitness values equals 1. Then, the population is sorted in descending normalized fitness value order. A random number R between 0 and 1 is chosen, and then, the first chromosome with normalized fitness value greater than R is selected for the next generation.

Fig. 5 Example of repair procedure for overlapping loops

Fig. 6 Flowchart of the repair procedure

Due to the fact that one of the objective functions of the proposed algorithm is to minimize the maximum workload of the system, the fitness function is defined based on the workload of loops.

The proposed algorithm applies the model presented by Shalaby et al. (2006) to determine the fitness function of produced solutions. Both loaded trips (α_p) and unloaded trips (φ_p) are taken into account for the workload of loop p (η_p). Equation 3 is used to calculate the value of α_p:

$$
\alpha_p = \left(\sum_{i \in p} \sum_{j \in p} f_{ij} \times \left(2T + \frac{d_{i,j}}{S} \right) + \sum_{i \in p} \sum_{j} f_{ij} \right.
$$
$$
\times \left(2T + \frac{d_{i,tp}}{S} \right) + \sum_{j} \sum_{i \in p} f_{ij}
$$
$$
\left. \times \left(2T + \frac{d_{i,tp}}{S} \right) \right) / 60,
$$

$$(3)$$

where f_{ij} is the flow between stations i and j per unit time (hour), d_{ij} is the rectilinear distance between station i and transfer point t of loop p, S is the AGV speed (unit distance/min), and T is the P/D time (minutes).

The value of φ_p can be calculated using a probabilistic approach:

$$
P_{ij} = \alpha_p \times \left(1 - \sum_{k \in K} P(W)_k \right) \times P(W)_j +
$$
$$
\left(1 - \frac{3}{2} \alpha_p \right) \times P(W)_j
$$

$$(4)$$

$$
E_{ij} = E_i \times P_{ij}
$$

$$(5)$$

$$
\varphi_p = \frac{\sum_i \sum_j E_{ij} \times \frac{d_{i,j}}{S}}{60}, \quad \forall i, j \in Z,
$$

$$(6)$$

where P_{ij} is the probability of assigning an empty trip from stations i to j, K is the set of stations or transfer points in loop p closer to station i than station j, E_i is the number of empty trips emerging from a station i, E_{ij} is the number of empty trips assigned from stations i to j, and Z is the set of all stations and transfer points in loop p. $P(W)_j$ is the probability that station j has a waiting job in its output queue and can be defined as

$$
P(W)_j = \frac{\sum_j f_{ij}}{\sum_i \sum_j f_{ij}}, \quad \forall i, j \in Z.
$$

$$(7)$$

An assumption which has been considered in Shalaby et al. (2006) is also taken into account in this model that is the number of empty AGV travels from a station i (E_i) is the same with the number of loaded AGV travels to that station.

The workload (η_p) of loop p can be calculated using

$$
\eta_p = \alpha_p + \varphi_p.
$$

$$(8)$$

In the selection level of the proposed algorithm, this cycle takes place over and over until the defined number of selected solutions is available; this is called fitness proportionate selection or roulette-wheel selection.

Crossover

From the previous section, the algorithm has selected some solutions to represent the first population for producing the next generation. Crossover in terms of GA is a type of genetic operator which is exploited to alter the chromosomes to mate them for producing the offspring. It is similar to the crossover takes place in nature for procreation which GA has inspired from. The proposed GA uses one-cut-point crossover operator.

The crossover point on each of the parent chromosomes is defined randomly. Just like the selection model, a crossover rate (a number between 0 and 1) which is associated with the chance of a chromosome to be selected for crossover is assigned to the solutions. Then, a random number between 0 and 1 is selected and the first chromosome with a crossover rate smaller than the assigned number is selected. Using this method, the probability for all the chromosomes to be selected for crossover is equal. The crossover operator switches a couple of genes on the parent chromosomes to generate the offspring. Figure 7 shows an example of a typical crossover used for the problem of partitioning.

Mutation

The final level for the proposed GA is mutation. Mutation is a type of a genetic operator which modifies a defined number of genes in a chromosome to produce totally new genes in a set of chromosomes. These new genes in chromosomes may help the GA to attain new better solutions which, otherwise, at the absence of such mutation would not be possible.

Mutation, in GA process, plays an important rule, as it prevents the solutions from cycling around local optima. Mutation takes place throughout evolution process based on an agreement set by the user which is called probability of mutation (P_{mut}). P_{mut} should not be set too high, because it may cause the whole population change and the solutions become random and unreliable.

The proposed algorithm uses P_{mut} of 0.05 and 0.07, and the next step is to choose the number of genes to be mutated. The number of genes in the solution then should be multiplied by 0.01 to attain the number of mutations. A number between 0 and 1 is assigned to each gene showing their mutation rate. A number between 0 and 1 is randomly chosen and a gene with the smaller mutation rate will be chosen and changed with a random integer between 1 and total numbers of loops. For example, in a four loop tandem configuration, the third gene on the chromosome 11**2**43234 is chosen for mutation, the integer 3 is generated between 1 and 4, so the result of the mutation will be the chromosome 11**3**43234.

Local search

As mentioned in the previous sections, MAs are different from GAs because of their applied local search. The local search is employed to refine the solutions and enhance their fitness. Local search moves from one solution to another in a neighborhood of solutions to find the local optimal. The process of search continues until a solution considered optimal is acquired, or the time limit for search is passed. The proposed MA in this study uses the TS defined by Zanjirani Farahani et al. (2007) as the local search.

Neighborhood structure In a tandem model, the neighborhood of a solution is acquired by transferring one workstation from one loop to another in a solution space while taking into account the feasibility of moves. A move is called feasible in the defined neighborhood structure if it does not result in intersections between two loops, generation of singletons or cause the workload of the loop which the workstation is transferred to, exceeds 1. A typical infeasible solution is shown in Fig. 8. The neighborhood of a solution consists of all possible solutions that can be attained by acquiring feasible moves.

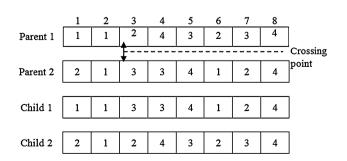Figure 9 **a** hows the feasible moves for station 11, and **b** shows the consequences of each move.
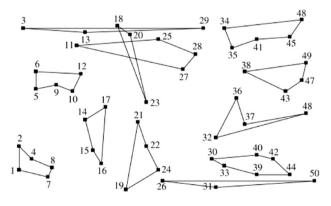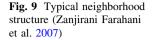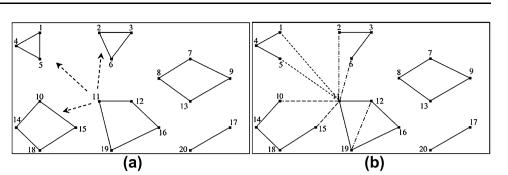
Fig. 7 Typical presentation of one-cut-point crossover

Fig. 8 Typical infeasible solution (Zanjirani Farahani et al. 2007)

Fig. 9 Typical neighborhood structure (Zanjirani Farahani et al. 2007)

Objective function of moves The objective of the partitioning problem is to minimize the workload of the system. This also applies to the moves. Hence, the idea of the local search algorithm is to target a loop with maximum workload and attempt to minimize its workload. As mentioned earlier, the workstation which is removed from one loop is transferred to the other loop in the solution. As a result, the workload of the second loop increases. The most desired move is such a move that can cause the maximum decrease in the workload of the selected loop while resulting in the minimum increase in workload of the second loop. Therefore, the evaluation criterion is: workload decrease of selected loop, minus workload increase of the second loop. Up until now, the algorithm has defined the loop with maximum workload. The problem is then to decide which move to choose. Since in most cases, transferring a workstation to the other loops may cause intersections or excessive AGV travel; therefore, it is feasible to check just a few adjacent loops. For the sake of simplicity, the proposed algorithm only checks six or seven loops based on Euclidean distance of their centers and only four stations in the selected loop will be chosen which their movement results in the largest workload decrease.

To prevent generation of singleton loops, a loop is selected only and if only it has three or more stations.

Tabu restrictions The algorithm applies a fixed size tabu list. When a station is removed from a loop, it cannot be added to that loop again for the next iterations. Although in case such a move would result in a better solution than the best solution found so far, its tabu status will be cancelled.

Diversification As a commonly applied strategy, the TS uses diversification in case all of the allowed movements result in infeasible solutions. The tabu list is cleared and the search restarts from the best known solution.

Termination In theory, the search process can go on forever until a solution found is better than the best known solution; however, in practice, it has to be a criterion to stop. The proposed TS stops after 20 iterations, since the best solution has not changed.

Results and discussion

The developed algorithms are implemented using Matlab and ran on a 2.00 GHz core two duo Intel with 2 GB RAM to solve several benchmark problems adopted from the literature to evaluate their performance. The successful implementation of similar algorithm with Matlab has been reported by Valipour et al. 2017; Valipour and Ma 2017; Valipour 2016; Rezaei et al. 2016; Valipour 2016; Valipour et al. 2013.

To adjust the indices of the GA and TS algorithms used in this study, the benchmark parameters used in the literature are adopted. Design of experiments is used to adjust the parameters of the proposed MA and GA algorithms including the population size, crossover rate, and mutation rate. Three levels have been selected for each parameter. These are 50, 100, and 150 for population sizes; 0.3, 0.5, and 0.7 for crossover rate; and 0.03, 0.05, and 0.07 for mutation rat. The summery of best combination of the parameters is shown in Table 2.

Table 2 Summary of the best combination of parameters for GA

System size	Zone loading	Number of zones	Crossover rate	Mutation rate	Population size
10	Any		0.7	0.07	150
20	Any	Large	0.5	0.07	
		Small	0.7	0.07	
30	Any		0.7	0.07	
40	Any	Large	0.5	0.07	
		Small	0.7	0.07	
50	Any		0.5	0.07	

Table 3 Result of designed experiments using GA and MA

Size	Zone loading	Number of zones	GA				MA				AVG IMPROV
			Best solution	Worst solution	AVG	Run time (s)	Best solution	Worst solution	AVG	Run time (s)	
10	High	Large	0.542 (100%)		0.542	0.48	0.542 (100%)		0.542	2.79	0
		Small	0.861 (80%)	0.961 (20%)	0.881	0.45	0.861 (100%)		0.861	2.70	2.27
	Mid	Large	0.434 (100%)		0.434	0.44	0.434 (100%)		0.434	2.70	0
		Small	0.690 (20%)	0.786 (80%)	0.767	0.42	0.69 (40%)	0.783 (60%)	0.746	2.52	2.74
	Low	Large	0.315 (100%)		0.315	0.43	0.315 (100%)		0.315	2.61	0
		Small	0.520 (40%)	0.564 (60%)	0.546	0.40	0.520 (100%)		0.520	2.43	4.83
20	High	Large	0.671 (40%)	0.712 (60%)	0.696	19.04	0.671 (40%)	0.709 (60%)	0.694	29.34	0.26
		Small	0.949 (40%)	0.951 (60%)	0.950	17.00	0.932 (50%)	0.94 (50%)	0.936	26.82	1.49
	Mid	Large	0.489 (80%)	0.516 (20%)	0.494	17.94	0.489 (80%)	0.508 (20%)	0.493	27.90	0.32
		Small	0.698 (40%)	0.712 (60%)	0.706	16.83	0.698 (60%)	0.711 (40%)	0.703	25.38	0.45
	Low	Large	0.369 (60%)	0.39 (40%)	0.377	18.02	0.369 (40%)	0.375 (60%)	0.373	26.91	1.27
		Small	0.518 (50%)	0.540 (50%)	0.529	16.41	0.518 (60%)	0.536 (40%)	0.525	23.49	0.72
30	High	Large	0.590 (20%)	0.621 (80%)	0.615	37.32	0.590 (80%)	0.615 (20%)	0.595	65.34	3.22
		Small	0.910 (80%)	0.930 (20%)	0.914	27.80	0.895 (50%)	0.916 (50%)	0.906	56.79	0.93
	Mid	Large	0.463 (70%)	0.499 (30%)	0.474	31.88	0.463 (40%)	0.470 (60%)	0.467	62.82	1.39
		Small	0.729 (80%)	0.760 (20%)	0.735	25.16	0.728 (80%)	0.764 (20%)	0.735	52.47	0
	Low	Large	0.280 (50%)	0.314 (50%)	0.297	30.60	0.28 (20%)	0.296 (80%)	0.293	58.41	1.41
		Small	0.470 (80%)	0.473 (20%)	0.471	24.91	0.467 (80%)	0.471 (20%)	0.468	47.43	0.59

Results of the designed experiments using GA and proposed MA are presented in Table 3. The column run time is the CPU time in seconds taken by the Matlab to solve each instance. The column AVG shows the average of best and worst solution for each algorithm and the column AVG IMPROV represents the improvement of objective function by applying proposed MA against basic GA. The average improvement is calculated using

$$A_{\mathrm{imp}} = (A_{\mathrm{GA}} - A_{\mathrm{MA}})/A_{\mathrm{GA}} \qquad (9)$$

where A_{GA} is the average of the results obtained from pure GA and A_{MA} is the average of the results of the proposed MA. Because the objective function of the partitioning procedure is to minimize the workload, the positive values represent improvements obtained.

In small problems, the objective functions of the two algorithms are often equal to the base algorithm, which might be the value of optimal solution. The percentage of the best solutions significantly increases as a function of the instance size. The percentage of worst solutions increases with problem size. It can be inferred that the base algorithm is not successful in solving problems with high number of stations. The number of zones has a significant effect on the running time of the TS algorithm. The average run time of the algorithm becomes higher as the number of zones increases. The explanation is that as the number of stations in the zones increases, more time is required to compute the workloads.

The results indicate that although the proposed algorithm requires longer runtime in comparison with GA, but the objective function is significantly being improved by applying the TS as a local search. Indeed, the longer operation time is due to the fact that an extra local search is performed which provides an optimal selection for GA and this leads in better objective function in most cases, especially when zone loading is high and there are many unordered selections for initiating the GA. Using TS intersections among the loops are avoided, however, the base GA does not provide any mechanism to guarantee the independence of loops in the final configuration.

Conclusion

A new MA for designing tandem AGVS has been developed to minimize the maximum workload of the system hence to make effective use of resources. A new local search based on the concept of TS was defined and applied to enhance the objective function of each newly generated individual. The performance of the proposed MA was evaluated through comparison with GA. The results demonstrated that in most problem cases, the local search improves the solutions of GA in term of system workload.

To consider uncertainty, time can be interpreted as triangular number and applied an expert system to infer cost. Then, the objective would be to determine a path minimizing both the time and cost criteria.

References

Arab A, Chetto H, Radounace L (1999) Flow path design for AGV systems. Stud Inform Control 8(2):97–106

Bozer Y (2004) Using existing workstations as transfer stations in tandem AGV systems. J Manuf Syst 23(3):229–241

Bozer Y, Srinivasan M (1992) Tandem AGV systems: a partitioning algorithm and performance comparison with conventional AGV systems. Eur J Oper Res 63(2):173–191

Bozer YA, Srinivasan MM (1989) Tandem configurations for AGV systems offer simplicity and flexibility. Ind Eng 21(2):23–27

Bozer YA, Srinivasan MM (1991) Tandem configurations for automated guided vehicle systems and the analysis of single vehicle loops. IIE Trans 23(1):72–82

ElMekkawy T, Liu S (2009) A new memetic algorithm for optimizing the partitioning problem of tandem AGV systems. Int J Prod Econ 118(2):508–520

Farling B, Mosier C, Mahmoodi F (2001) Analysis of automated guided vehicle configurations in flexible manufacturing systems. Int J Prod Res 39(18):4239–4260

Groover MP (2007) Automation, production systems, and computer-integrated manufacturing, 3rd edn. Prentice Hall, Upper Saddle River, New Jersey

Ho YC, Hsieh PF (2004) A machine-to-loop assignment and layout design methodology for tandem AGV systems with multiple-load vehicles. Int J Prod Res 42(4):801–832

Hsieh LF, Sha D (1996) A design process for tandem automated guided vehicle systems: the concurrent design of machine layout and guided vehicle routes in tandem automated guided vehicle systems. Integr Manuf Syst 7(6):30–38

Huang C (1997) Design of material transportation system for tandem automated guided vehicle systems. Int J Prod Res 35(4):943–953

Kim K, Chung BD, Jae M (2003) A design for a tandem AGVS with multi-load AGVs. Intl J Adv Manuf Technol 22(9–10):744–752

Kim KS, Chung BD (2007) Design for a tandem AGV system with two-load AGVs. Comput Ind Eng 53(2):247–251

Laporte G, Farahani R, Miandoabchi E (2006) Designing an efficient method for tandem AGV network design problem using tabu search. Appl Math Comput 183(2):1410–1421

Lin J, Chang C, Liu W (1994) A load-routeing problem in a tandem-configuration automated guided-vehicle system. Int J Prod Res 32(2):411–427

Liu F, Chen J (1997) Analytical framework for designing the divided automated guided vehicles system. Int J Ind Eng 4(2):90–102

Macqueen J (1967) Some methods of classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, pp 281–297

Merchant M (1977) The inexorable push for automated production. Prod Eng 24:4–465

Rezaei M, Valipour M, Valipour M (2016) Modelling evapotranspi-ration to increase the accuracy of the estimations based on the climatic parameters. Water Conserv Sci Eng 1(3):197–207

Rezapour S, Zanjirani Farahani R, Miandoabchi E (2011) A machine-to-loop assignment and layout design methodology for tandem AGV systems with single-load vehicles. Int J Prod Res 49(1):3605–3633

Salehipour A, Kazemipoor H, Moslemi Naeini L (2010) Locating workstations in tandem automated guided vehicle systems. Int J Adv Manuf Technol 52(1–4):321–328

Shalaby MA, El Mekkawy TY, Fahmy SA (2006) Zones formation algorithm in tandem AGV systems:a comparative study. Int J Prod Res 44(3):505–521

Sinriech D (1995) Network design models for discrete material flow systems: a literature review. Int J Adv Manuf 10(4):277–291

Tompkins JA, White JA, Bozer YA (2010) Facilities planning. Wiley, New Jersey

Valipour M (2016) Variations of land use and irrigation for next decades under different scenarios. Irriga 1(01):262

Valipour M, Ma G (2017) Temporal analysis of reference evapotran-spiration to detect variation factors. Int J Glob Warm (**In Press**)

Valipour M (2016) How much meteorological information is necessary to achieve reliable accuracy for rainfall estimations? Agriculture 6(4):53

Valipour M, Banihabib ME, Behbahani SMR (2013) Comparison of the ARMA, ARIMA, and the autoregressive artificial neural network models in forecasting the monthly inflow of Dez dam reservoir. J Hydrol 476:433–441

Valipour M, Gholami Sefidkouhi MA, RaeiniSarjaz M (2017) Selecting the best model to estimate potential evapotranspiration with respect to climate change and magnitudes of extreme events. Agric Water Manag 180:50–60

Ventura J, Lee C (2001) A study of the tandem loop with multiple vehicles configuration for automated guided vehicle systems. J Manuf Syst 20(3):153–165

Yu W, Egbelu PJ (2002) Design of a variable path tandem layout for automated guided vehicle systems. J Manuf Syste 21(6):480–481

Zanjirani Farahani R, Laporte G, Miandoabchi E, Bina S (2007) Designing efficient methods for the tandem AGV network design problem using tabu search and genetic algorithm. Int J Adv Manuf Technol 36(9–10):996–1009