A practical model of routing problems for automated guided vehicles with acceleration and deceleration

Tatsushi NISHI*, Susumu MATSUSHITA*, Takeshi HISANO** and Masashi MORIKAWA**

*Graduate School of Engineering Science, Osaka University
1-3 Machikaneyama-cho, Toyonaka-shi, Osaka 560-8531, Japan
E-mail: nishi@sys.es.osaka-u.ac.jp
**Daifuku Co., Ltd.
1225 Nakazaiji, Hino-cho, Gamo-gun, Shiga 529-1662, Japan

Received 15 January 2014

Abstract
We consider an optimization of conflict-free routing problems for automated guided vehicles (AGV) with acceleration and deceleration. A continuous time model is developed to represent the dynamics of vehicles. In the proposed model, the transportation model is discretized into several regions. A network model is created by taking into account the acceleration and deceleration motions. The acceleration and deceleration are represented at curve locations. Column generation heuristic is used to find a near-optimal solution. In this algorithm, we construct a heuristic rule to generate a feasible solution with acceleration and deceleration of vehicles after the column generation. The pricing problem is represented by a resource constrained shortest path problem, which is effectively solved by a labeling algorithm. The dominance relation for acceleration and deceleration is addressed. In the proposed model, the dynamics of real speed AGV model are reflected into the routing problems. By comparing the performance of the conventional method, the effectiveness of the proposed method is demonstrated.

Key words: Automated guided vehicle, Column generation heuristics, Routing, Labeling algorithm

1. Introduction

In transportation systems such as semiconductor plants, container terminals, and flexible manufacturing systems (FMS), multiple automated guided vehicles (AGVs) are widely used. Transportation tasks are performed at loading points and/or unloading points. In practice, it is required to generate conflict-free routes for multiple vehicles, which minimizes the total transportation time.

In previous research, there are a number of studies on vehicle scheduling problems (Bunte and Kliewer, 2009). A branch-and-cut-and-price algorithm for vehicle routing problem using column generation approach was proposed by Bettinelli, et al. (2011). Reveliotis and Roszkowska (2011) regarded the vehicle route planning problem as a resource allocation system and they developed a deadlock avoidance policy. In Feillet et al. (2004), vehicle routing problem (VRP) is formulated as an elementary shortest path problem with resource constraints and a labeling algorithm and column generation approach are used. Recently, routing problems for AGVs have been widely studied (Le-Anh and Koster, 2006). Nishi et al. (2005) developed a Lagrangian decomposition technique for routing problem for AGVs. Tanaka et al. (2010) proposed a Petri net decomposition approach with deadlock avoidance policy to dynamic routing of AGV systems with unidirectional lanes. Nishi et al. (2012) presented a Petri net decomposition approach for bidirectional AGV systems. Desaulniers et al. (2003) applied a column generation approach for routing problems of AGVs. By utilizing the column generation approach, a tight lower bound can be derived by solving the restricted master problem which is a linear programming problem with huge number of columns. In previous studies, for simplicity, the vehicle speed is constant. However, in practical transportation systems, each vehicle has its acceleration and deceleration at traveling, stopping and curving. It is therefore important to take into account the acceleration and
deceleration during the route planning phase. In our previous work, we consider only the acceleration and deceleration of vehicles at straight lanes (Matsushita, et al., 2013). In this paper, we propose a practical routing model that considers the acceleration and deceleration at curve locations. In real transportation systems, there is a speed constraint at curve locations. If the vehicle travels at curve location at the same speed at straight location, curving may fail on the side. Therefore, the modeling of acceleration and deceleration is one of the crucial problems for routing AGV systems.

We also propose a column generation heuristic to solve the routing problem for AGV systems with acceleration and deceleration at curving. The transportation model is divided into several regions. Each region is represented by a node where it can occupy only one vehicle. A continuous time model is used to represent the timing chart with real acceleration and deceleration motions. A network model is created by taking into account the acceleration and deceleration motions. The collision avoidance constraints are included at a given interval during the time horizon. Column generation heuristic is applied to solve the proposed practical model. In the column generation approach, a tight lower bound with a good accuracy is derived by solving the restricted master problem and the pricing problem repeatedly. The solution of the pricing problem with a minimum reduced cost is added to the set of columns in the restricted master problem. The pricing problem can be solved effectively by a labeling algorithm using dominance relation. We obtain a near-optimal solution efficiently by removing non-optimal labels by the dominance relation for the proposed model. The solution obtained by the column generation method is generally infeasible because it is equivalent to the solution of the continuous relaxation problem. In order to create a feasible solution, heuristics are used to modify the infeasible solution to a feasible one. The proposed model is applied to represent real dynamics of AGV systems. The routing results derived by the proposed method are compared with that of a detailed AGV simulator representing real dynamics of vehicles. Computational results are executed to evaluate the validity of the proposed model.

The paper is organized by follows. Section 2 describes the problem definition for the routing problem of AGVs with acceleration and deceleration. Section 3 explains the modelling of the problem. Section 4 explains the algorithm of column generation and the heuristic algorithm for generating a feasible solution. In Section 5, we provide the computational results of a case study. The proposed model is applied to a real speed model in Section 6. Section 7 concludes our study and states future works.

2. Routing problem for automated guided vehicles with acceleration and deceleration

We define the routing problem for AGV systems. Consider the situation that multiple vehicles $k \in K$ are traveling in a transportation system which consists of unidirectional lanes. Each vehicle $k \in K$ has a uniformly accelerated motion. Each vehicle has acceleration and deceleration when stopping, starting and curving. The time to change its direction can be ignored. An initial position is assigned to each vehicle in advance. The loading and unloading points of tasks are also given in advance. A task time is defined as the constant time required for a vehicle at the loading and unloading points. Each vehicle can have only one task at the same time. The routing problem for AGV systems is to determine the routing from an initial position to the loading point, and the routing from the loading point to the unloading point to minimize the total transportation time without collisions among vehicles. The transportation time means the sum of the time from the starting point to the loading point (pickup time) and from the loading point to the unloading point (delivery time). To avoid collisions, we should exclude the situations that multiple vehicles are at the same point at the same time or that a vehicle overtakes another vehicle by accelerating or decelerating. It is assumed that tasks are assigned to vehicles by a nearest neighbor method (Eda, et al., 2012). The method is designed such that the task is assigned to the vehicle which has the least estimated traveling time. The estimated completion time of task $i$ by vehicle $k$, can be obtained by the sum of the estimated completion time in which vehicle $k$ completes all assigned tasks without considering collisions with other vehicles, and the minimum traveling time from the unload point of final task of vehicle $k$ to the load point of task $i$, without considering collisions with other vehicles.

3. Mathematical model

The mathematical model of the routing problem is explained. The transportation system has unidirectional lane where a vehicle can travel only in one direction. First, we divide the transportation system into several regions. A node is defined in each region. The collision avoidance condition must be satisfied in all regions. In order to define the
problem, we define the traveling motions in those regions by using the following 4 patterns (Fig. 1). Acceleration (i) means that vehicle accelerates and then travels at a constant speed. Deceleration (ii) means that vehicle travels at a constant speed, then decelerates and stops. Constant speed (iii) means that vehicle travels at a constant speed. Acceleration and deceleration (iv) means that vehicle accelerates, travels at a constant speed, decelerates and stops. Additionally, we define 2 motions: wait and task. Wait indicates that the vehicle waits on the same node. Task indicates that the vehicle is performing a loading or unloading task on the same node. Each task has the constant task time. These motions are used to represent routing models with acceleration and deceleration.

Fig. 1 Definition of AGV motions with acceleration and deceleration.

The time required to perform the following 4 motions can be calculated by Eqs. (1)-(4). We assume that each vehicle follows a uniformly accelerated motion.

**Acceleration**

\[ \text{SP} / \text{a} + (L - \text{a}(\text{SP} / \text{a})^2 / 2) / \text{SP} \]  

**Deceleration**

\[ \text{SP} / \text{d} + (L - \text{SP}(\text{SP} / \text{d}) + \text{d}(\text{SP} / \text{d})^2 / 2) / \text{SP} \]  

**Constant Speed**

\[ L / \text{SP} \]  

**Acceleration and Deceleration**

\[ \text{SP} / \text{a} + \text{SP} / \text{d} + (L - \text{a}(\text{SP} / \text{a})^2 / 2 - \text{SP}(\text{SP} / \text{d}) + \text{d}(\text{SP} / \text{d})^2 / 2) / \text{SP} \]  

\(L\) is the length of each region. \(SP\) is the maximum speed of the vehicle. \(a\) is the acceleration of the vehicle. \(d\) is the deceleration of the vehicle. The time necessary for wait and task can be set freely.

In real transportation systems, there is a speed constraint at curve locations. We add the following 6 motions in Fig. 2 for curving motions. For example, (v) from zero speed to the curve speed, (vi) from the high speed to the curve speed, (vii) from the curve speed to the high speed, (viii) from the curve speed to zero speed, (ix) the constant curve speed (x) from zero speed to the curve speed, then from the curve speed to zero speed. By using these 6 motions additionally, the speed constraints can be imposed at curve locations. The time necessary for these motions is calculated as the same as the above traveling motions.

Fig. 2 Definition of AGV motions with acceleration and deceleration at curve location.
4. Column generation approach

The column generation approach is an effective decomposition method based on the simplex method. This method solves the linear programming problem of the restricted master problem with huge columns. The derived dual variables are used in the pricing problem. The pricing problem is to derive a column which has the minimum reduced cost.

4.1 Column generation heuristic

The algorithm of column generation heuristic consists of the following steps:

Step 1: Generate an initial feasible solution
   - Generate an initial feasible solution which is a conflict-free routing for all vehicles. Create a set of initial columns \( \tilde{R} \), which is a set of the tentative routes for vehicles.

Step 2: Renew dual variables
   - Solve the restricted master problem and the dual variables are updated.

Step 3: Solve the pricing problem
   - Solve pricing problem and generate the routing for each vehicle with a minimum reduced cost.

Step 4: Evaluation of convergence
   - If there are no negative values among the reduced costs of all the vehicle routes, go to Step 6. Otherwise go to Step 5.

Step 5: Add the derived columns to the restricted master problem
   - Add the column generated at Step 3 with the negative reduced cost to \( \tilde{R} \), and go to Step 2.

Step 6: Derive a lower bound
   - We obtain a lower bound from the restricted master problem.

Step 7: Use a heuristic algorithm to generate a feasible solution with a limited set of columns \( \tilde{R} \). Generally, the branch-and-bound method is used for solving the restricted master problem with integer constraints.

4.2 Generation of an initial feasible solution

In order to execute the column generation, an initial feasible solution is required to ensure the feasibility of the dual problem of the restricted master problem. After the shortest paths from the starting point to its destination are generated by Dijkstra’s algorithm for all vehicles to execute the assigned task, an initial feasible solution is generated by heuristics to avoid collisions (Tanaka et al., 2010). Firstly, the states of each vehicle are classified into traveling, temporary stopping and final stopping. If there are multiple vehicles in a region at the same time, the routing for the vehicle is changed to avoid collision. If the state of one of the vehicles is the final stopping, a temporary destination is set for the vehicle. Otherwise the routing of the vehicle arrived into the region is delayed according to the priority of the vehicle by using the heuristic rule with acceleration and deceleration (Fig. 3). By utilizing the heuristic rule, the motion of the vehicle is changed in order to satisfy the speed constraint. We explain the heuristic algorithm that can generate a feasible solution with acceleration and deceleration in Fig. 3.

In cases (i) and (ii) in Fig. 3, we add the waiting motion before the acceleration and deceleration, and before the acceleration motion. In cases (iii) and (iv), the vehicle is decelerated. In case (iii), the acceleration and deceleration is changed to the acceleration, deceleration, acceleration and deceleration. In case (iv), the constant speed and deceleration is changed to deceleration, acceleration and deceleration. In cases (v) and (vi), the vehicle travels with a constant speed. In case (v), the acceleration and constant speed is changed to the acceleration, deceleration, and acceleration. In case (vi), the constant speed is changed to deceleration and acceleration. All the patterns and changes of the motions are prepared to satisfy the speed constraint, and we can generate a feasible solution by delaying the motions of vehicles. For the motions at curving, we define heuristic rules as same as the above heuristic rules.

4.3 Restricted master problem

The restricted master problem is a continuous relaxation of the set partitioning formulation of the original problem. The restricted master problem can be formulated as a linear programming problem.
Fig. 3 Steps (i)-(vi) to construct a feasible solution in the heuristics with acceleration and deceleration.

Sets
- $K$: the set of vehicles
- $\overline{R}^k$: the set of possible routes for vehicle $k \in K$
- $R^k$: the limited set of possible routes for vehicle $k \in K$ ($\overline{R}^k \subseteq R^k$)
- $N$: the set of nodes
- $T$: the set of time periods during planning horizon
- $D$: the set of tasks

Parameters
- $c_r$: delivery time of route $r$ of vehicle $k \in K$, $r \in \overline{R}^k$
- $e_{n,t}^r$: binary parameter that is equal to 1 if a vehicle using route $r$ is on node $n \in N$ at time $t \in T$, and to 0 otherwise
- $\tau^d$: binary parameter that is equal to 1 if a vehicle using route $r$ has task $d \in D$, and 0 otherwise
- $\Delta t$: time duration to avoid collisions in each region

Decision variables
- $\theta_{r,k}$: binary variable that is equal to 1 if route $r \in R^k$ is selected by vehicle $k \in K$, and 0 otherwise

Problem formulation

$$\min \sum_{k \in K} \sum_{r \in \overline{R}^k} c_r \theta_{r,k}$$  (5)
Equation (5) is the objective function to minimize the total delivery time. 
subject to

$$\sum_{k \in K} \sum_{r \in R^k} e^{i,j}_{k,R} \theta_{r,k} \leq 1 \quad (\forall n \in N, \forall i = 1, 2, \ldots, T / \Delta t)$$

(6)

The constraints (6) ensure that there is at most one vehicle existing on a node at each time. The conflicts among vehicles can be avoided by the constraints.

$$\sum_{k \in K} \sum_{r \in R^k} x^{d,j}_{k,R} \theta_{r,k} = 1 \quad (\forall d \in D)$$

(7)

The constraints (7) ensure that all tasks are executed by the set of all vehicles.

$$\sum_{r \in R^k} \theta_{r,k} = 1 \quad (\forall k \in K)$$

(8)

The constraints (8) enforce the selection of exactly one route for each vehicle.

$$\theta_{r,k} \geq 0 \quad (\forall k \in K, \forall r \in R^k)$$

(9)

The constraints (9) specify the domain constraints of variable $\theta_{r,k}$.

4.4 Pricing problem

The pricing problem is regarded as a resource constraint shortest path problem. The labeling method is applied to solve the problem. The labeling algorithm can find a path from source to sink with a minimum cost with resource constraints. The method stores the information of the states such as the cost and the resource consumption as labels. The states and the labels are renewed during the search. A label is defined by Eq. (10).

$$L = (c, s, t, h_1, \ldots, h_i, t)$$

(10)

$L$ is a label on node $n$, $l$ is the number of tasks, $c$ is the reduced cost from the start node to node $n$, $s$ is the speed of the vehicle on node $n$ at time $t$, $t$ is the task number of the vehicle at time $t$, $h_i$ is binary variable that is equal to 1 if the vehicle finishes task $i$ by time $t$ and 0 otherwise, and $t$ is the time of the label. A label represents one state from start node to the node. There may be multiple labels at the same node.

4.4.1 Renewing labels

We renew a label $L = (c, s, t, h_1, \ldots, h_i, t)$ on node $n$, and create label $L' = (c', s', t', h_1', \ldots, h_i', t')$ on node $n'$. Node $n'$ can be reached from node $n$.

The reduced cost $c'$ can be renewed by the following equation:

$$c' = c - \pi_{n'}$$

(11)

$\pi_{n'}$ is the cost necessary for the transition from node $n$ to node $n'$. If the motion is acceleration or constant speed, the speed $s'$ is 1. If the motion is deceleration, acceleration and deceleration, wait or task, the speed $s'$ is 0. If the motion is task and the task is loading, the task number $t'$ is the number of the task. If the motion is task and the task is unloading, the task number $t'$ is 0. Otherwise $t' = t$. If the motion is task and the task is unloading task $i$, $h_i'$ is 1. Otherwise $h_i' = 0$. Time $t'$ is renewed by the following equation.

$$t' = t + t_{\text{transit}}$$

(12)

where $t_{\text{transit}}$ is the time for the vehicle to travel in a region.
4.4.2 Dominance relation

If two labels on node \( n \), \( L_1 = (c^1, s^1, r^1, h^1_1, \ldots, h^1_l) \) and \( L_2 = (c^2, s^2, r^2, h^2_1, \ldots, h^2_l) \) satisfy the following equations,
\[
c^1 \leq c^2, s^1 = s^2, r^1 = r^2, h^1_1 = h^2_1, \ldots, h^1_l = h^2_l, t^1 = t^2
\]
the reduced cost of \( L_2 \) cannot be less than the reduced cost of \( L_1 \) in the future. If \( L_1 \) and \( L_2 \) are not the same, \( L_1 \) dominates \( L_2 \), and there is no need to renew \( L_2 \).

**Proposition 1**

If \( c^1 \leq c^2, s^1 = s^2, r^1 = r^2, h^1_1 = h^2_1, \ldots, h^1_l = h^2_l, t^1 = t^2 \) and \( L_1 \) and \( L_2 \) are not the same, label \( L_1 \) dominates label \( L_2 \).

In this study, we consider the continuous time representation. Therefore, it is difficult to satisfy the equation \( t^1 = t^2 \), because \( t \) is real number. We relax this equation into \( t^1 - \Delta d \leq t^2 \leq t^1 + \Delta d \) taking into account the continuity of time \( t \). By relaxing the equation, we can solve the pricing problem faster.

4.4.3 Labeling algorithm

The labelling algorithm to solve the pricing problem consists of the following steps.

**Step 1:** Generate the initial labels
Generate a label on a given initial position of the vehicle, and set time \( t = 0 \) and node number \( n = 0 \).

**Step 2:** Renew the labels
If there is the label which is not renewed on node \( n \), renew the label to node \( n' \) to which the vehicle can transit from node \( n \). Generate unrenewed labels on node \( n' \). If the renewed time is over the planning horizon, we generate a renewed label instead of an unrenewed label.

**Step 3:** Delete the labels
After renewing labels, we delete the labels dominated by other labels using dominance relation.

**Step 4:** Change variables
If \( n = |N| \) and there are no unrenewed labels on any of the nodes, go to Step 5. If \( n = |N| \) and there are unrenewed labels on any node, set \( n = 0 \) and go to Step 2. Otherwise set \( n = n + 1 \) and go to Step 2.

**Step 5:** Search the route
Search a route with the least reduced cost among the routes which are not deleted. If the reduced cost is negative, add the route to the restricted set of columns and the algorithm is finished.

4.5 Construction of a feasible solution by column generation

The solution obtained by the column generation approach is the optimal solution of the continuous relaxation problem, but it may be infeasible for the original problem. We propose a heuristic algorithm to generate a feasible solution to provide a good upper bound instead of using the branch and bound method to derive a feasible solution.

**Heuristic algorithm to find a feasible solution**

**Step 1:** Generate an initial feasible solution
Generate an initial feasible solution by using the heuristics to avoid conflicts that is explained in Section 4.2.

**Step 2:** Column generation
Derive an optimal continuous relaxation solution by solving linear programming

**Step 3:** Fix a column to adopt as a feasible solution
Find the highest value of \( \theta_{r,k} \) in the optimal continuous relaxation solution. Set \( \theta_{r,k} = 1 \) in order to fix the column \( r \) to the vehicle \( k \) in the feasible solution.

**Step 4:** Judgement of the completion of the algorithm
If the columns for all vehicles are fixed, then the algorithm is finished and we can obtain an upper bound. Otherwise go to Step 5.

**Step 5:** Check feasibility of the solution
If the solution is infeasible, generate a feasible solution by using heuristics (see Fig. 3) with the fixed vehicles and infeasible vehicles, then return to Step 2.

The feasibility of the solution can be ensured by the initial feasible solution derived by the algorithm explained in Section 4.2. If the solution is infeasible at the Step 5 of the algorithm, we can generate a feasible solution by using heuristics of Section 4.2. By using this algorithm, a good feasible solution can be derived.

5. Computational results

We show the computational results to investigate the performance of the proposed method. The first one is to investigate the effectiveness of our proposed method. The second one is to investigate the dominance parameter $\Delta d$. The third one is to investigate the time duration parameter $\Delta t$. We coded the program with Microsoft Visual C++ 2008 Express Edition. The branch and bound method with IBM ILOG CPLEX12.1 was used for solving linear programming problems. An Intel(R) Core(TM) i7 2.80GHz with 3.46GB memory was used for computations. The parameters were the max speed at straight location, max speed at curve location, acceleration, deceleration, length of each region, task time, wait time (1 sec.), planning period (100 sec.), dominance parameter and time duration parameter. These parameters are based on actual information. The initial point of each vehicle and load point and unload point of each task are randomly given. The number of vehicles and the number of tasks are equal for all instances. This is because we assume the congestions when every vehicle is traveling with its loading and unloading tasks simultaneously.

5.1 Effectiveness of the proposed method

In order to investigate the effectiveness of our proposed method, we conducted the simulation on a small-scale transportation system. In this simulation, we set task time=4, $\Delta d=0.2$ and $\Delta t=0.3$. The transportation system layout and its mathematical model with 8 nodes are shown in Fig. 4. The arrows in the layout indicate the direction of unidirectional lanes. We set the length of each region 10 m. The number of vehicles is from 2 to 6. We compare the upper bound (UB), the lower bound (LB) and computation time of the Proposed Method (PM) and branch and bound method (BB). In BB method, the branch ad bound with limited set of columns is used to generate a feasible solution after the column generation is converged. The computational results are summarized in the Table 1.

![Transportation layout Model with 8 nodes](Fig. 4  Small scale transportation system with 8 nodes.)

| Method | Number of vehicles | LB       | UB       | Time [s] |
|--------|--------------------|----------|----------|----------|
| BB     | 2                  | 34.3     | 34.3     | 1.1      |
|        | 3                  | 44.1     | 44.1     | 2.6      |
|        | 4                  | 75.8     | 84.1     | 17.8     |
|        | 5                  | 97.5     | 101.9    | 47.4     |
|        | 6                  | 123.3    | 147.7    | 84       |
| PM     | 2                  | 34.3     | 34.3     | 1        |
|        | 3                  | 44.1     | 44.1     | 2.6      |
|        | 4                  | 75.8     | 84.1     | 17.8     |
|        | 5                  | 97.5     | 99.8     | 59.6     |
|        | 6                  | 123.3    | 132.9    | 99.3     |
From these computational results of Table 1, it is confirmed that the proposed column generation heuristic (PM) can generate better upper bounds than those of BB. This is because PM can generate more columns to create a feasible solution than BB. Our proposed method takes more computational time than BB, because PM fixes the routes one by one. If the number of vehicles is small, we can obtain the results with the same value of UB and LB. It means that an optimal solution can be obtained by the proposed algorithm. If the number of vehicles is large, the gap between UB and LB is increased. If there are a number of vehicles in the small scale transportation system, the interferences among them are also increased. This is the reason why it becomes difficult to generate UB and LB with good accuracy.

5.2 Effects of dominance parameter

In order to investigate the effectiveness of dominance parameter $\Delta d$, we conducted a simulation on a small-scale transportation system. In this simulation, we set the task time 12 and $\Delta t = 0.8$. We used the transportation layout and its model in Fig. 5. In the transportation system, we set curve location and consider the speed constraint at curve location. The number of nodes in the transportation system is 10. The number of vehicles is 3 and the number of tasks is the same as the number of vehicles. We compare the computation time to solve the pricing problem (PP-time), the lower bound (LB), the computation time to get lower bound (LB-time), the upper bound (UB) and the time to get upper bound (UB-time). We set the dominance parameter $\Delta d = 0, 0.1, 0.5$ and 1. The computational results are summarized in the Table 2.

![Transportation layout Model with 10 nodes](image)

**Fig. 5** Small scale transportation system with 10 nodes.

| $\Delta d$ | PP-time [s] | LB | LB-time [s] | UB | UB-time [s] |
|---|---|---|---|---|---|
| 0 | - | - | - | - | - |
| 0.1 | 26.6 | 136.6 | 2314.7 | 151.2 | 2769.3 |
| 0.5 | 1.2 | 136.6 | 108.7 | 151.2 | 130.5 |
| 1 | 1.1 | 136.7 | 88.3 | 154.1 | 105.7 |

The proposed method cannot generate a solution due to out of memory when the parameter $\Delta d = 0$ is used. From these computational results, it is confirmed that the setting of $\Delta d$ is important to improve the solution quality and total computation time. If the value of $\Delta d$ is small, it takes longer computation time to solve problem. On the other hand, if $\Delta d$ is large, LB and UB is larger. Therefore, the value of $\Delta d$ should be sufficiently small if we want to derive an exact solution of routing problem with acceleration and deceleration. Also we can set the value of $\Delta d$ larger if the computation time of the routing is much faster.

5.3 Effects of time duration parameter

The transportation systems depicted in Figs. 5 and 6 are used. In the simulation, we set the parameters: task time=12 and $\Delta t = 0.2$. We set the length of each region to 10m. In case (i), we use the simple transportation system in Fig. 5. The number of nodes is 10. The number of vehicles is 2 to 5 and the number of tasks is the same as the number of vehicles. We compare the lower bound (LB), upper bound (UB), and computation time (Time) among three cases of time duration. The parameter $\Delta t$ is set to 0.3, 0.6, and 0.8. The computational results are shown in Table 3.
Table 3  Effects of the time duration parameter in case (i).

| ∆t | Number of vehicles | LB    | UB    | Time [s] |
|-----|--------------------|-------|-------|----------|
| 0.3 | 2                  | 63.5  | 63.5  | 10.0     |
|     | 3                  | 107.1 | 111.6 | 85.3     |
|     | 4                  | 180.8 | 180.8 | 264.0    |
|     | 5                  | 201.6 | 201.6 | 342.3    |
| 0.6 | 2                  | 63.5  | 63.5  | 10.0     |
|     | 3                  | 104.8 | 104.8 | 57.8     |
|     | 4                  | 160.8 | 173.8 | 104.2    |
|     | 5                  | 193.9 | 197.9 | 315.5    |
| 0.8 | 2                  | 63.5  | 63.5  | 10.1     |
|     | 3                  | 104.8 | 104.8 | 40.5     |
|     | 4                  | 160.4 | 160.4 | 78.9     |
|     | 5                  | 193.5 | 206.4 | 222.1    |

From these computational results, it is confirmed that the setting of ∆t is important for solution quality and total computation time. If the value of ∆t is small, UB is large and it takes much computation time but an exact solution for the routing problem of vehicles with acceleration and deceleration can be obtained. On the other hand, if ∆t is sufficiently large, UB is small and the computational time becomes much shorter. Therefore the value of ∆t should be sufficiently small if we want to derive an exact solution of routing problem with acceleration and deceleration. Also we can set the value of ∆t larger if the computation time of the routing is much faster.

In case (ii), we use the medium size transportation layout and its model in Fig. 6. The number of nodes of this transportation system is 22. The number of vehicles is 2 to 4. In this simulation, we compare the lower bound (LB) and upper bound (UB) and computational time (Time) with case (i). We set ∆t = 0.3, 0.6, 0.8. The computational results are shown in Table 4. From these computational results, it is confirmed that it takes much larger computational time to solve medium size transportation system than those of the small size transportation system.

Fig. 6  Medium scale transportation system with 22 nodes.

6. Application of real speed model

The main characteristic of our proposed model is that the model can treat real numbers for velocity. We apply the real speed model and evaluate the gap between our route plan and the simulator of the logistic company DAIFUKU. We define 10 types of motions shown in Fig. 7. These motions correspond to 4 motions in Fig. 1 and 6 motions in Fig. 3 for curving. The task motion and the wait motion are added additionally. Two types of task sets with no interference among vehicles and with interference among vehicles are prepared. We coded the program with Microsoft Visual C++ 2008 Express Edition. The branch and bound method with IBM ILOG CPLEX12.6 was used for the solving linear programming problem. An Intel(R) Core(TM) i7 2.80GHz with 3.46GB memory was used for computations.
Table 4  Effects of the time duration parameter in case (ii).

| ∆t (s) | Number of vehicles | LB [s] | UB [s] | Time [s] |
|--------|--------------------|--------|--------|----------|
| 0.3    | 2                  | 66.8   | 66.8   | 2115.6   |
|        | 3                  | 167.7  | 173.1  | 5211.1   |
|        | 4                  | 218.8  | 233.7  | 9157.8   |
| 0.6    | 2                  | 65.8   | 72.4   | 2790.7   |
|        | 3                  | 167.7  | 167.7  | 4216.4   |
|        | 4                  | 218.8  | 218.8  | 7673.8   |
| 0.8    | 2                  | 65.8   | 65.8   | 2083.8   |
|        | 3                  | 167.7  | 167.7  | 3753.0   |
|        | 4                  | 218.8  | 218.8  | 5734.1   |

Fig. 7  Definition of AGV motions for the real speed model. (i)-(iv) correspond to Fig. 1. (v)-(x) correspond to Fig. 2.

In the simulation, we use the transportation layout depicted in Fig. 8. The number of nodes in the transportation system is 54. The number of vehicles and the number of tasks is 3. Ten cases of instances with randomly generated loading and unloading points are solved. The average gap between our routing result and the simulator is 0.34 second for 10 instances without interferences. The average gap for 10 instances with interferences among vehicles is 8.62 second. If the interferences among vehicles are increased, the gap becomes larger because the conflict-free routing method is different. In our proposed model, the collisions are avoided in fixed time interval for each region. Therefore if there is a vehicle on a node, the other vehicle cannot enter the region. This is one of the reasons why the gap becomes larger. If there is no interference between vehicles, the gap between the optimization results and the simulation results is sufficiently small with less than 1.0 second. It demonstrates that the proposed model can represent real speed model of AGV systems.
7. Summary and conclusion

In this paper, we proposed an efficient column generation approach for solving the routing problem for AGVs with acceleration and deceleration. In the pricing problem, we applied a labeling algorithm using the dominance relation. In the restricted master problem, the collision avoidance constraint was considered in the case of acceleration and deceleration. From the computational results, we demonstrated that our proposed method can generate better upper bounds than those of the conventional column generation method with BB. The effects of the parameters to the performance of the proposed method have been also investigated. The real speed model has been applied to the routing problems for AGV systems. The gap between our results and those of the simulator developed by DAIFUKU has been evaluated. Future work is to evaluate the effectiveness of the proposed method under dynamic situations.

References

Bettinelli, A., Ceselli, A. and Righini, D., A branch-and-cut-and-price algorithm for the multi-depot heterogeneous vehicle routing problem with time windows, Transportation Research - Part C: Emerging Technologies, Vol. 19, No. 5 (2011), pp. 723–740.

Bunte, S. and Kliewer N., An overview on vehicle scheduling models, Public Transport, Vol. 1, No. 4 (2009), pp. 299–317.

Desaulniers, G., Langevin, A. and Riopel, D., Dispatching and conflict-free routing of automated guided vehicles: an exact approach, International Journal of Flexible Manufacturing Systems, Vol. 15, No. 4 (2003), pp. 309–331.

Eda, S., Nishi, T., Mariyama, T., Kataoka, S., Shoda, K. and Matsumura, K., Petri net decomposition approach for bi-objective routing for AGV systems minimizing total traveling time and equalizing delivery time, Journal of Advances Mechanical Design, Systems, and Manufacturing, Vol. 6, No. 5 (2012), pp. 672–686.

Feillet, D., Dejax, P., Gendreau, M. and Gueguen, C., An exact algorithm for the elementary shortest path problem with resource constraints: application to some vehicle routing problems, Networks, Vol. 44, No. 3 (2004), pp. 216–229.

Le-Anh, T. and Koster, M.B.M. De., A review of design and control of automated guided vehicle systems, European Journal of Operational Research, Vol. 171, No. 1 (2006), pp. 1–23.

Matsushita, S., Nishi, T., Kataoka, S., Hisano, T. and Matsumura, K., Column generation heuristics to route planning problems for automated guided vehicles with acceleration and deceleration, Proceedings of International Symposium on Scheduling 2013 (2013), pp. 91-96.

Nishi, T., Ando, M. and Konishi, M., Distributed route planning for multiple mobile robots using an augmented Lagrangian decomposition technique, IEEE Transactions on Robotics, Vol. 22, No. 6 (2005), pp. 1191–1200.

Nishi, T. and Tanaka, Y., Petri net decomposition approach for dispatching and conflict-free routing of bidirectional automated guided vehicle systems, IEEE Transactions on Systems, Man and Cybernetics Part A: Systems and Humans, Vol. 42, No. 5 (2012), pp. 1230-1243.

Reveliotis, S. A. and Roszkowska, E., Conflict resolution in free-ranging multivehicle systems: a resource allocation paradigm, IEEE Transactions on Robotics, Vol. 27, No. 2 (2011), pp. 283–296.

Tanaka, Y., Nishi, T. and Inuiguchi, M., Dynamic optimization of simultaneous dispatching and conflict-free routing for automated guided vehicles - Petri net decomposition approach, Journal of Advanced Mechanical Design, Systems, and Manufacturing, Vol. 4, No. 3 (2010), pp. 701–715.