2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Towards Optimal Lane-changing Coordination of CAVs in Multi-lane Mixed Traffic Scenarios

Yan Ding, Yijun Mao, Chongshan Jiao and Pengju Ren

Abstract— **Lane changing is a fundamental but challenging operation for moving vehicles. Connected and Automated Vehicles(CAVs) enable autonomous vehicles to cooperate with each other to accomplish the lane changing tasks, profiting from their communication ability. However, dispatching CAVs in mixed traffic remains difficult due to the stochastic behaviors and uncertain intentions of Human-Driven Vehicles(HDVs). To tackle this issue, this paper devises a coordination approach based on Conflict-Based Search(CBS) theory. Firstly, HDVs are accurately modeled as constraints to enable usage of CBS in the mixed traffic. Additionally, virtual goals are introduced to search CAVs' priority and outlets along with path finding. Furthermore, we optimize the performance of CBS in dense traffic by defining the concept of *following vehicles*. Experiments show that performance is improved by utilizing new conflict prioritizing rules and a heuristic value calculation method that derived from *following vehicles*. Finally, we introduce grouping vehicles to extend the proposed method for solving extremely dense and large instances at a scale of more than one hundred without significant loss in efficiency.**

I. Introduction

With the rapid development of vehicle-to-everything (V2X) communication, Connected and Automated Vehicles (CAVs), as well as CAV-based traffic control, offer a promising way to increase road capacities and tackle traffic congestion problems. Lane changing is an essential action for both autonomous vehicles and human-driven vehicles (HDVs) to cope with emergencies and promote efficiency. However, irrational lane changing behavior may cause lateral collisions. Vehicles need to negotiate with each other to obtain a safe and effective plan. CAVs can share information with neighboring vehicles and road infrastructure, enabling them to carefully plan by receiving guidance instructions from these sources. Despite the fact that CAVs will dominate transportation market in the future, HDVs will continue to exist for a long time. However, coordinating CAVs and HDVs in a safe and effective manner is challenging due to the stochastic behaviors and uncertain intentions of HDVs. Therefore, CAVs must learn to coexist with HDVs in mixed traffic during the transition to an all-CAV era.

The authors are with the National Key Laboratory of Human-Machine Hybrid Augmented Intelligence, National Engineering Research Center for Visual Information and Applications, and Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, Xi'an, China. E-mail: dingyan@xjtu.edu.cn, oraclemao@stu.xjtu.edu.cn, jchongshan@163.com and pengjuren@xjtu.edu.cn.

This work was supported in part by the Key Research and Development Program of Shaanxi No.2022ZDLGY01-08, National Natural Science Foundation of China under Grant 62088102, and Fundamental Research Funds for the Central Universities under Grant xtr072022001 (Corresponding author: Pengju Ren).

A bi-level tightly coupled framework has been designed in [1] to minimize delay when lane changing is inevitable. At the upper level, Monte Carlo Tree Search (MCTS) is used to decide the right-of-way of vehicles. At the lower level, predefined trajectories are selected during the search process. However, this framework is sub-optimal and cannot be applied in mixed traffic. Some research combined formation assignment with lane changing methods [2] [3] [4] based on multi-vehicle motion planning(MVMP). The key problem of MVMP is that most existing methods cannot handle unexpected behaviors of HDVs.

A bi-level formation control framework is proposed in [5] to tackle obstacles in multi-lane scenarios. The authors establish a relative coordinate system (RCS) to describe and regulate the motion of vehicles. While the framework can be applied in mixed traffic, its main drawback is the lack of optimality guarantee at the system level.

The Conflict-based Searching (CBS) method based on RCS, introduced in [6], is utilized to plan collision-free paths for vehicles according to given assignment results. The efficiency of the overall traffic system heavily relies on the assignment of vehicles. However, the rule-based vehicle assignment in [5] and [6] may not guarantee shortest delay.

Liu etc. [7] developed a dispatching model based on integer linear programming (ILP) to coordinate CAVs. They employed tree-based heuristic search (THS) and platoon segmentation techniques to enhance computational efficiency. The proposed method can handle up to 60 vehicles, although optimality is not guaranteed. They also assume a full penetration of CAVs in the system. Another highlight of [7] is that it proposed different objective functions, including minimum-steps, minimum-maneuvers and minimum-disturbance. However, the throughput of the traffic flow is also an important objective function. To the best of our knowledge, there is currently no effective solution available to coordinate CAVs in mixed traffic while guaranteeing optimality within a limited decision time.

Coordinating CAVs and Multi Agent Path Finding(MAPF) have a lot in common when we reformulate the problem of coordinating CAVs in RCS with a similar method proposed by [5]. MAPF offers numerous excellent solvers that can be utilized to address the coordination issue of CAVs effectively.

CBS [8] and its enhancements [9] are among the ideal methods in MAPF problems to coordinate agents in shared environment. Furthermore, most of them are optimal and complete. However, there is still a long way to go when adapting CBS to the CAV domain. Firstly, a CBS instance to be solved should contain a map, start points and goal

points. While map and start points are easy to get when building RCS, determining goals becomes challenging because assigning targets on a grid map means deciding priority sequences and outlets for CAVs. Determining priorities and outlets in advance may negatively impact system efficiency. Secondly, CBS is designed for cooperative agents. Therefore, we need to find ways to handle the influence of HDVs and obstacles. Lastly, CBS becomes inefficient when there are dense agents on the map. However, dense traffic flows are common in real-life traffic. The proposed method uses an improved CBS to solve the problem of coordinating a large fleet of CAVs in mixed traffic at an acceptable computational cost. The contributions of this paper are threefold:

- A framework based on CBS theory is proposed for coordinating CAVs in mixed traffic, which provides guidance for multi-agent behavior planning.
- Virtual goals are designed to achieve optimal results without determining priority and outlet in advance. HDVs and obstacles are considered as initial constraints for each CAV to accommodate mixed traffic.
- We introduce a new method for calculating admissible heuristic values and prioritizing conflicts in order to speed up the high-level search of CBS. Additionally, we utilize grouping to enhance the performance of the proposed method for large fleets in dense traffic.

II. RELATED WORK

Some studies [10] [11] have designed dedicated lanes for CAVs, allowing only CAVs to occupy the dedicated lanes. However, appropriate lane changing plan is still necessary, it has been revealed that dedicated lanes may reduce capacity and cause congestion in [12].

A cooperative lane changing strategy is proposed in [13] using a transferable utility game framework. However, the framework only considers two vehicles and the overall traffic densities are only improved slightly. The Cooperative Adaptive Cruise Control (CACC) modeling framework is extended in [14] with new algorithms to describe the interactions among the CACC vehicles and HDVs in mixed traffic. This framework does not model the overall efficiency of the traffic, and the lane changing actions can only be performed after a gap appears in the target lane. Sheng etc. [15] proposed a learning-based cooperation-aware lane changing method and primarily focused on ego-car decision making.

CBS [8] is a two level Multi Agent Path Finding algorithm. At the high level, conflicts between individual path of agents are searched and a Conflict Tree(CT) is built. Nodes in the CT represent solutions with or without conflicts and include constraints on the motion of the agents. At the low level, different single agent path finding algorithms can be performed with the constraints imposed by the high level CT node. The search process repeats the operation of generating node, choosing node with minimal cost, choosing conflict and generating children nodes. CBS has been proven to be optimal due to minimal-first node selection policy.

The performance of CBS varies in different scenarios. The main issue with CBS is that each conflict can only be solved by extending the CT, which may slow down the solving process. To address this problem, several approaches have been developed. Meta-agent CBS(MA-CBS) [9] merges agents into small groups of agents called meta-agents. Bypass [16] tries to find alternative paths instead of expanding node. The remaining approaches can be classified into three categories: 1) Distinguishing and prioritizing different types of conflicts. 2) Add heuristics to improve search efficiency. 3) Identifying and resolving symmetries.

Arbitrary selection of conflict for splitting may increase the size of CT. Improved CBS (ICBS) [9] classifies conflicts into three types according to the cost increase of two children node compared to the node split by the conflict: Cardinal conflicts have increased cost in both children nodes. Semi-cardinal conflicts have only one child with increased cost. Non-cardinal conflicts' both children nodes have the same cost with the parent node. Splitting cardinal conflict first results in a smaller CT and thus reduces search time. Bypassing is applied only to semi-cardinal and non-cardinal conflicts since there is no alternative path without cost increase.

CBS selects the node in the CT with the lowest cost, where the cost is determined by summing the lengths of all paths. This value can be viewed as g-value of high level search. Felner et al. [17] chose to add admissible h-value to the node selection according to the cardinal conflicts that will be resolved in future splits. Li et al. [18] enhanced the calculation of admissible h-value by reasoning whether semi-cardinal and non-cardinal conflicts may result in being cardinal due to pairwise symmetry. The literature called the heuristic in [17] CG and proposed DG and WDG as reformed heuristic. DG considers semi-cardinal and non-cardinal conflicts with pairwise symmetry, incorporating them into the conflict graph during the calculation of the minimum vertex cover(MVC). WDG further improved upon DG by considering that some conflicts with pairwise symmetry may result in a cost larger than 1. It constructs a weighted pairwise dependency graph with varying weights on edges based on the conflict graph. WDG also extends MVC to edge-weighted minimum vertex cover(EWMVC) in order to obtain larger but admissible h-value.

Pairwise symmetry is a phenomena that two agents have many promising paths, but every combination results in a collision [19]. This phenomena is common and can cause the CT to expand exponentially without contributing to the search procedure. Three types of pairwise symmetry have been discovered so far: Rectangle reasoning was proposed firstly in [19]. Target symmetry and corridor symmetry subsequently identified in [20]. New generalized versions of rectangle and corridor reasoning techniques are then provided [21]. Mutex propagation is a more useful and general technology for detecting and resolving symmetries [22].

All methods described above are dedicated to find obvious and latent cardinal conflicts based on the existing paths in the node. The main reason for CBS and its variants' inefficiency in dense traffic flow is that when we resolve a conflict, only agent in the chosen conflict or agents violating the positive

constraints replan their paths. The new paths may cause more sequential conflicts, but only few conflicts, sometimes only one, can be detected by expanding just one node. Numerous nodes are expanded when the sequence is long.

In the following sections, we design virtual goal points to avoid prioritizing agents and choosing goals, create a conflict prioritizing criterion to reduce node expansions, calculate a more precise but admissive h-value, and introduce grouping to solve problems in dense traffic.

III. METHODOLOGY

To apply CBS to the coordination of moving vehicles, we construct a grid map in RCS from lane net generated by the physical simulator mentioned in [23]. The velocity of the RCS is set to be average velocity of all vehicles. As depicted in Fig.1, the smallest X coordinate is set at the rear-most vehicle. The direction of movement for the vehicle formation is aligned with the X axis, while the Y axis points vertically towards right from this direction. Moving CAVs are represented as agents. The velocity choices of CAVs are set to be high, medium and low. The medium velocity is equal to that of RCS, which means CAVs take longitudinal wait action in RCS when using medium velocity. Thus, high velocity and low velocity correspond to forward and backward actions respectively. Medium velocity is used when CAVs are taking lane changing action. Moving HDVs and static obstacles such as broken vehicles are considered as constraints since they are not connected or controllable. Their states can only be observed and predicted through perception. We do not impose any restrictions on the velocity of HDVs but assume that they are reasonable drivers who will pursue higher velocity while ensuring their safety under the Responsibility-Sensitive Safety(RSS) rules. Our objective is to find paths for all CAVs with minimal overall time steps.

A. Adding virtual goals to map

The grid map built in RCS is modeled as a undirected graph $G(V, E)$, where V are vertexes representing grids and E are edges linking grids. The map consists of m columns and n rows $(m, n \in \mathbb{Z})$, where n is equal to the number of lanes and m is derived from the quotient of formation length L_f and grid length L_g as :

$$m = \lceil L_f / L_g \rceil \tag{1}$$

A set of k CAVs are labeled as $a_1, ..., a_k$. Each CAV has a start vertex $start_i \in V$. $start_i$'s position in RCS is calculated by its initial position as:

$$start_i^x = \lfloor (x_i - x_0)/L_g + 0.5 \rfloor \tag{2}$$

$$start_i^y = \arg\min_j \{y_i - lane_j\} \quad j \in \{1, ..., n\} \tag{3}$$

where x_i and y_i are the x and y position of ith CAV in the lane net coordinate. x_0 is the x position of the rear most vehicle, $lane_j$ is the position of the leftmost lane.

Each CAV should have a goal $goal_i$. Thus, k virtual vertexes $\{g_1, ..., g_k\}$, which is equal to the number of CAVs, are added to the grid map. Let $v_i, i \in \{1, ..., n\}$ be the vertexes with largest column number. If goal $g_j, j \in \{1, ..., k\}$ corresponds to vehicle $a_j, j \in (1, .., k)$ which can exit from a set of lanes $\{l_i\}, i \in (1, ...n)$. Then we add edges $\{e_{i,j}\}$ between $\{v_i\}$ and $\{g_j\}$. All edges linking virtual goals have no conflicts with each other. So, ransition from vertexes with the largest column number to virtual goals is never blocked. It is similar to removing agents when they reach their goals, but virtual goals offer additional benefits. We do not need to predefine target lanes or prioritize agents. Furthermore, integrating traffic rules such as left-changing vehicles occupying the leftmost lane becomes easier.

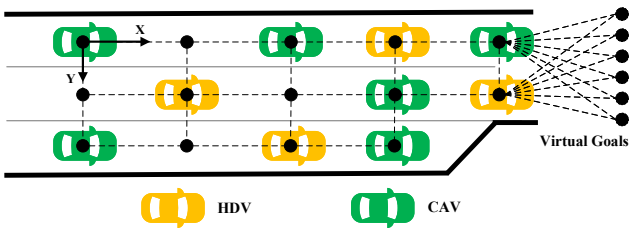


Fig. 1. Adding virtual goals to RCS in multi-lane mixed traffic scenario.

B. Adding HDVs and obstacles as constraints

In CBS, vertex constraint can be described as $\langle a_i, v, timestep \rangle$, which means agent a_i is prohibited from occupying vertex v at $timestep$. In our work, HDVs and obstacles are viewed as initial constraints for all CAVs. Given the velocity of RCS v_R. The state vector of ith HDV is $(a_{i,h}, v_{i,h})$. The position of HDVs in RCS at time step t can be represent in:

$$a_{i,h,R}^{x,t} = \lfloor (a_{i,h}^x + (v_{i,h}^x - v_R) \times t - x_0)/L_g + 0.5 \rfloor \tag{4}$$

$$a_{i,h,R}^{y,t} = \arg\min_j \left\{ a_{i,h}^y + v_{i,h}^y \times t - lane_j \right\} \quad j \in \{1, ..., n\} \tag{5}$$

Vertex constraints are then derived according to the positions of HDVs at each time step as $v_t = (a_{i,h,R}^{x,t}, a_{i,h,R}^{y,t})$. Vertex constraints are described as :

$$\{\langle a_i, v_t, t \rangle\} \quad i \in \{1, ..., p\} \quad t \in \{0, ..., r\} \tag{6}$$

where p is the number of HDVs and r is the makespan of the solution.

Edge constraints are derived when HDVs are moving in RCS. If the position of HDV $v_{t-1} \neq v_t$, we add edge constraints between vertexes from v_{t-1} to v_t as constraints in :

$$\langle a_i, v_{t-1}, v_t, t-1 \rangle \tag{7}$$

The obstacles are dealt with using a similar method, except that vertexes passed by the obstacles are permanently blocked in RCS due to the large velocity gap.

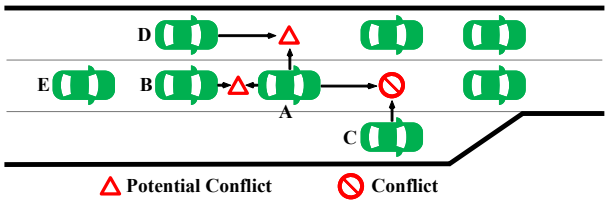


Fig. 2. Potential cardinal conflicts between following vehicles.

C. Exploiting Potential Cardinal Conflicts

Estimating the cost of the expanded nodes as close as possible is crucial to reduce the search time. Li et al. [21] exploited all the cardinal conflicts in current node and merged the *Multi-Valued Decision Diagrams*(MDDs) to find pairwise symmetry cardinal conflicts to build a *conflict graph*. The *minimum vertex covert*(MVC) of the *conflict graph* is set to be h-value of the node. Although the h-value has been accurately estimated, conflicts that may arise in the subsequent node are disregarded. These conflicts are typically challenging to identify in general MAPF problems, but they can be detected in our specific application.

When we solve a cardinal conflict between two agents, two child nodes are generated and one of the two agents replans its path in each child node. The path of the re-planned agent may conflict with other agents. Among all the conflicts, special attention should be given to conflicts with the following car. Fig. 2 illustrates the significance of following cars. Once vehicle C merges, CBS imposes a constraint on vehicle A, prohibiting it from moving forward. If Vehicle A chooses to wait, it will create a conflict with vehicle B. When resolving the conflict between vehicle A and vehicle B, we generate two child nodes. In the first node, vehicle A turns left and vehicle B moves forward, resulting in an increase in cost by 1; In the second node, vehicle B replans its path while vehicle A waits. Regardless of the path taken by vehicle B, the cost of the solution also increases by 1. Previous literature considered these two child nodes to have equal costs. However, if we choose the first node with fewer *following vehicles*, only two levels of nodes will be expanded. Without considering the number of *following vehicles*, there is a one in seven chance that the number of levels will be two. This probability decreases further when the formation length is longer due to every *following vehicle* in two child nodes wavering between waiting for a time step and changing lanes. By taking into account the number of *following vehicles*, CBS can select a node with fewer future conflicts.

We define *following vehicle* as a vehicle sharing the same path segments from the time step of the conflict to the time step arriving at the goal and are singleton at the time step in conflict. We also extend *following vehicle* to a *following vehicle sequence* if vehicles behind the detected *following vehicle* will encounter cardinal conflicts and increase paths length. We use the pseudo code in algorithm 1 to show the search process.

Algorithm 1: Finding following vehicle sequence

Data: chosen conflict in parent node $\langle a_i, a_j, v_2, v_2, t\rangle$, new constraint $\langle a_i, v_1, v_2, t\rangle$, current node $node$
Result: heuristic value of current node $h(n)$

Replan the $path$ of a_j;
$s = 0$;
while *following vehicle exist* **do**
  foreach $agent$ *in* $node$ **do**
    if $(path_n^{(t+s)\to(end-1)} = path_{a_j}^{t\to(end-1)}) \wedge (path_n^{(t+s)}$ *is single*$)$ **then**
      add n to *following vehicle list*;
      $a_j = n$;
      $s = s + 1$;
      following vehicle exist;
      break;
    end
    else
      following vehicle not exist;
    end
  end
end
Add following vehicle list to *conflict graph*;
$h(n) \leftarrow$ ComputeMVC(*conflict graph*);
return $h(n)$;

D. Prioritizing Conflicts

Former literature resolves cardinal conflicts first. When there are multiple cardinal conflicts, the preference is given to conflicts that can incur higher costs. In our work, we introduce two additional criteria to reduce node expansion operations: 1) resolving conflicts between preceding vehicles and 2) addressing conflicts that will result in greater cost increments during subsequent node expansions.

For the first criteria, we use the instance in Fig. 3 to explain. Two vehicles encounter lane drop and need to merge into the dense traffic flow on the left lane. There will be two conflicts if both vehicles turn left at the same time, namely vehicle A with B and vehicle C with vehicle D. If we choose conflict between vehicle C and D instead of conflict between vehicle A and B, the conflict between vehicle C and D will reoccur when we solve the conflict between vehicle A and B. For the second criteria, We use the f-value as the cost. This criteria is strongly associated with h-value calculation method described in previous section. Our work takes cost increase due to the cardinal conflicts appearing in the descended node expansion into consideration, profiting from the idea of exploiting *following vehicles*.

CBS-RTC [21] gave target conflicts the highest priority, followed by corridor conflicts and rectangle conflicts. In our work, we also consider these pairwise symmetry. Target conflicts do not exist due to the virtual goals. Corridor conflicts do not exist because all goals are on the same side. Rectangle conflicts exist but rare. We give rectangle conflicts

highest priority.

To summarize the priority rules: 1) We give rectangle conflicts highest priority. 2) Conflicts between preceding vehicles are solved if there are no corridor and rectangle conflicts. 3) We solve conflicts that will increase more cost in the following node expansion if there are still ties.

E. Grouping Vehicles in Extremely Dense Traffic

Though we have been trying to improve the performance of CBS in dense traffic, it is still impossible to use it efficiently in extremely congested scenarios with occupation rates as high as 80% or 90%. To the best of our knowledge, there is currently no optimal solution available. One possible but suboptimal approach is grouping CAVs into batches. In this method, after solving for one batch of CAVs, their paths are used as constraints for the next batch. Instead of segmenting platoons based on position, vehicles are randomly selected to reduce agent density within each batch. The performance of this grouping strategy falls between that of optimal solvers and prioritizing all agents before path finding. Experimental results demonstrate that dividing CAVs into two batches significantly reduces CPU time without a significant loss in efficiency or success rate. If necessary, more batches can be generated if the solving process remains challenging.

IV. EXPERIMENTS AND RESULTS

The proposed algorithm is implemented in C++14 and ROS Melodic. All the simulation experiments are executed on a desktop computer with an Intel i9-9900K CPU and a 32GB RAM running Ubuntu 18.04. The interactive multi-agent simulation platform is built from [23]. Lane drop scenarios are taken here and all HDVs and CAVs are randomly-distributed. A CAV-only case is shown to test the scalability of our algorithm. Large cases and small cases with different occupation rate and CAV percentage are used to verify the effectiveness of modifications. All experiments were repeated 100 times. Success rate and CPU time are used to test the performance. Time limit is set to be 1 sec for all experiments. Disjoint splitting, mutex propagation techniques proposed in [22] to speed up CBS is also adopted. Symmetry reasoning are not used because corridor and target conflicts do not exist and rectangle conflicts are rare. HDVs are set to take their optimal paths and conflicts between them are solved by rules. The rules can be described as: 1) Following vehicle give way to preceding vehicle. 2) Lane-changing vehicle give way to straight going vehicle.

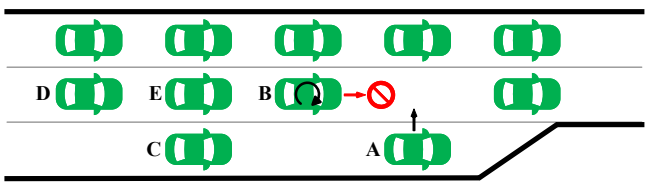


Fig. 3. Solve conflicts between preceding vehicles

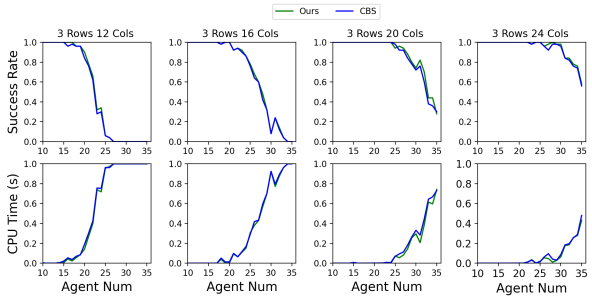
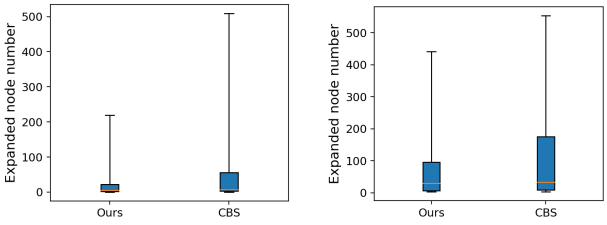


Fig. 4. Success rate and CPU time in CAV only scenarios



(a) Small scale cases (b) Large scale cases

Fig. 5. Average expanded node number in mixed traffic cases

A. CAV Only Cases

We use CAV only scenarios to illustrate the performance of our algorithm compared to existing methods because there are few references on coordinating CAVs or formation control in mixed traffic to our knowledge. On a three-lanes map with first lane dropped, experiments on different number of grids each lane show that optimal method becomes inefficient when the occupation rate is near or over $^2/_3$, as shown in Fig. 4. The reason for this is that the two lanes available for outlet have a capacity of 24. When the number of agents on road exceed this number, two available lanes will be filled in and conflicts grow rapidly. Exploring potential conflicts also loses its effectiveness when agents take full occupation of the two available lanes because the number of following vehicles are same.

B. Different Scale Cases with Different Occupation Rate

The small and the large scale cases both use a map of 3 lanes. The first lane is blocked at the endpoint. Small cases' lane consists of 6 grids. Large cases' lane consists of 12 grids. Based on the observation in CAV only experiments, we pay our attention mainly on the dense traffic. Fig. 6 shows that our method always work better than the original CBS. When the traffic is semi dense, the margin is larger because exploring potential conflicts help the solver choose the less occupied lane without expanding extra nodes. Fig. 5 shows that exploring *following vehicle* greatly reduces the number of nodes expanded. Further more, the distribution of nodes expanded number is more concentrated.

C. Grouping vehicles in Extremely large and dense map

CBS-based method failed to tackle with large and dense traffic scenarios. As shown in Fig. 7, a fleet of 80 vehicles

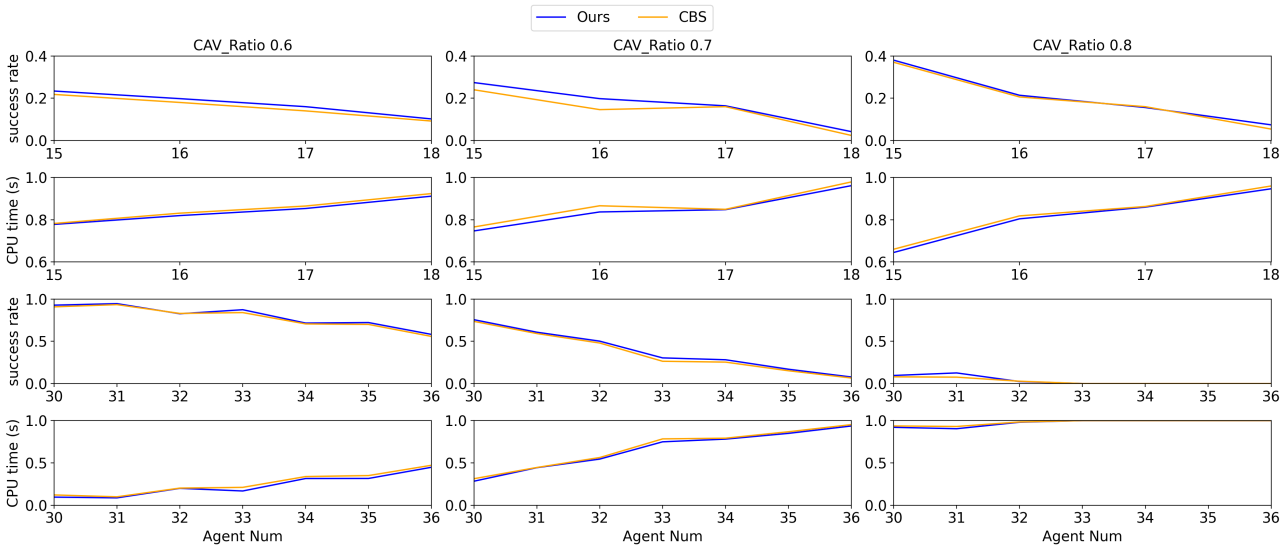


Fig. 6. Success rate and CPU time of varying occupation rate in mixed traffic. Top two rows are small map with 6×3. Botttom two rows are large map with 12×3. Improvement are more obvious in semi dense scenarios

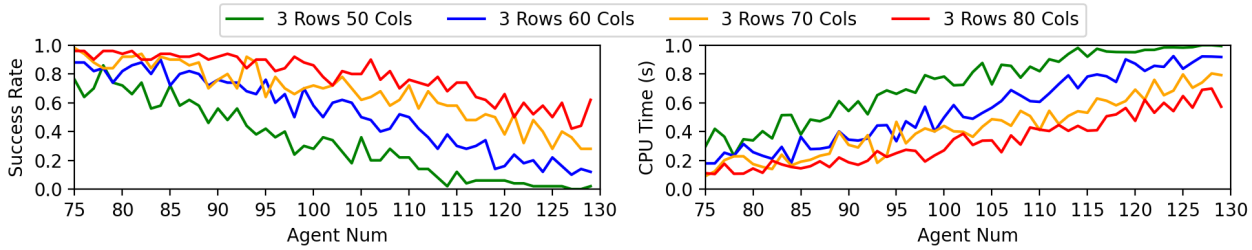


Fig. 7. Success rate and CPU time in large and dense scenarios. Occupation rate and agent number still matter. Grouping vehicles enable larger fleet.

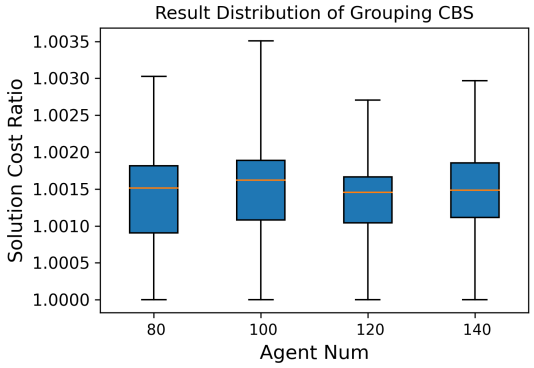


Fig. 8. Cost ratio distribution of different number of agents with grouping CBS on a 80×3 map. Cost are compare with optimal solution cost if the instance is solvable and minimal cost of grouping CBS otherwise.

on a large map with 50×3 failed to get coordinating results in 1 second. However, the occupation rate is only 50%. Vehicle number is another limiting condition for solving CBS problems in real time. Fig. 8 shows the cost of the final solution of grouping vehicles into two batch. For CPU time and success rate shown in Fig. 7, grouping CBS can solve an instance of 130 agents with considerable success rate. For the optimality side, we repeat instances with same $\langle starts, goals \rangle$ 3000 times and divide the agents into two batch randomly each time. If the times of repetition is enough, the minimal solution cost will approximate the optimal cost. Therefore, Fig. 8 proved that efficiency can be maintained.

V. Conclusion

In this paper, we utilized a modified CBS approach to solve the problem of CAVs coordination in mixed traffic. The lane allocation and passing priorities of vehicles are determined concurrently. Obstacles and HDVs are effectively modeled to facilitate the coordination of CAVs in mixed traffic. The results obtained from our framework serve as guidance for behavior planning. We improve the performance of CBS in large and dense traffic scenarios by exploring potential conflicts between following car sequences. Numeric experiments show that we can get optimal results in 1 second when agents number is up to dozens and occupation rate is bellow $2/3$ in designed lane drop scenarios. We can tackle instances of 130 agents by grouping vehicle into two batches, which is sub optimal with a factor near 1 empirically. In this work, we demonstrate that MAPF methods have great potential in transportation applications.

REFERENCES

[1] H. Xu, Y. Zhang, C. G. Cassandras, L. Li, and S. Feng, A bi-level cooperative driving strategy allowing lane changes, Transportation research part C: emerging technologies, vol. 120, p. 102773, 2020.

[2] M. Cai, Q. Xu, K. Li, and J. Wang, Multi-lane formation assignment and control for connected vehicles, in 2019 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2019, pp. 1968–1973.

[3] X. Qian, A. De La Fortelle, and F. Moutarde, A hierarchical model predictive control framework for on-road formation control of autonomous vehicles, in 2016 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2016, pp. 376–381.

[4] R. Du, S. Chen, Y. Li, M. Alinizzi, and S. Labi, A framework for lane-change maneuvers of connected autonomous vehicles in a mixed-traffic environment, Electronics, vol. 11, no. 9, p. 1350, 2022.

[5] M. Cai, Q. Xu, C. Chen, J. Wang, K. Li, J. Wang, and Q. Zhu, Formation control for connected and automated vehicles on multi-lane roads: Relative motion planning and conflict resolution, IET Intelligent Transport Systems, vol. 17, no. 1, pp. 211–226, 2023.

[6] M. Cai, Q. Xu, C. Chen, J. Wang, K. Li, J. Wang, and X. Wu, Formation control with lane preference for connected and automated vehicles in multi-lane scenarios, Transportation research part C: emerging technologies, vol. 136, p. 103513, 2022.

[7] Q. Liu, X. Lin, M. Li, L. Li, and F. He, Coordinated lane-changing scheduling of multilane cav platoons in heterogeneous scenarios, Transportation Research Part C: Emerging Technologies, vol. 147, p. 103992, 2023.

[8] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, Conflict-based search for optimal multi-agent pathfinding, Artificial Intelligence, vol. 219, pp. 40–66, 2015.

[9] E. Boyarski, A. Felner, R. Stern, G. Sharon, O. Betzalel, D. Tolpin, and E. Shimony, Icbs: The improved conflict-based search algorithm for multi-agent pathfinding, in Proceedings of the International Symposium on Combinatorial Search, vol. 6, no. 1, 2015, pp. 223–225.

[10] A. Ghiasi, O. Hussain, Z. S. Qian, and X. Li, A mixed traffic capacity analysis and lane management model for connected automated vehicles: A markov chain method, Transportation Research Part B: Methodological, vol. 106, pp. 266–292, 2017.

[11] X. Hua, W. Yu, W. Wang, and W. Xie, Influence of lane policies on freeway traffic mixed with manual and connected and autonomous vehicles, *Journal of Advanced Transportation*, vol. 2020, pp. 1–20, 2020.

[12] S. Woo and A. Skabardonis, Flow-aware platoon formation of connected automated vehicles in a mixed traffic with human-driven vehicles, Transportation research part C: emerging technologies, vol. 133, p. 103442, 2021.

[13] D. Lin, L. Li, and S. E. Jabari, Pay to change lanes: A cooperative lane-changing strategy for connected/automated driving, Transportation Research Part C: Emerging Technologies, vol. 105, pp. 550–564, 2019.

[14] H. Liu, X. D. Kan, S. E. Shladover, X.-Y. Lu, and R. E. Ferlis, Modeling impacts of cooperative adaptive cruise control on mixed traffic flow in multi-lane freeway facilities, Transportation Research Part C: Emerging Technologies, vol. 95, pp. 261–279, 2018.

[15] Z. Sheng, L. Liu, S. Xue, D. Zhao, M. Jiang, and D. Li, A cooperation-aware lane change method for autonomous vehicles, arXiv preprint arXiv:2201.10746, 2022.

[16] E. Boyrasky, A. Felner, G. Sharon, and R. Stern, Don't split, try to work it out: Bypassing conflicts in multi-agent pathfinding, in Proceedings of the International Conference on Automated Planning and Scheduling, vol. 25, 2015, pp. 47–51.

[17] A. Felner, J. Li, E. Boyarski, H. Ma, L. Cohen, T. S. Kumar, and S. Koenig, Adding heuristics to conflict-based search for multi-agent path finding, in Proceedings of the International Conference on Automated Planning and Scheduling, vol. 28, 2018, pp. 83–87.

[18] J. Li, A. Felner, E. Boyarski, H. Ma, and S. Koenig, Improved heuristics for multi-agent path finding with conflict-based search. in IJCAI, vol. 2019, 2019, pp. 442–449.

[19] J. Li, D. Harabor, P. J. Stuckey, H. Ma, and S. Koenig, Symmetry-breaking constraints for grid-based multi-agent path finding, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, 2019, pp. 6087–6095.

[20] J. Li, G. Gange, D. Harabor, P. J. Stuckey, H. Ma, and S. Koenig, New techniques for pairwise symmetry breaking in multi-agent path finding, in Proceedings of the International Conference on Automated Planning and Scheduling, vol. 30, 2020, pp. 193–201.

[21] J. Li, D. Harabor, P. J. Stuckey, H. Ma, G. Gange, and S. Koenig, Pairwise symmetry reasoning for multi-agent path finding search, Artificial Intelligence, vol. 301, p. 103574, 2021.

[22] H. Zhang, J. Li, P. Surynek, T. S. Kumar, and S. Koenig, Multi-agent path finding with mutex propagation, Artificial Intelligence, vol. 311, p. 103766, 2022.

[23] Y. Mao, Y. Ding, C. Jiao, and P. Ren, UCLF: An uncertainty-aware cooperative lane-changing framework for connected autonomous vehicles in mixed traffic, in2023 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2023, pp. 1–8.