Collaborative Collision Avoidance for CAVs in Unpredictable Scenarios

Dhruvkumar Patel
Department of Computer Science
University of Texas at Dallas
Richardson, Texas
dhruv@utdallas.edu

Rym Zalila-Wenkstern
Department of Computer Science
University of Texas at Dallas
Richardson, Texas
rymw@utdallas.edu

2020 IEEE 3rd Connected and Automated Vehicles Symposium (CAVS) | 978-1-7281-9001-3/20/$31.00 ©2020 IEEE | DOI: 10.1109/CAVS51000.2020.9334661

***Abstract*—Modern connected and automated vehicles (CAV) are capable of making informed decisions in unexpected situations. CAVs can achieve this by collaborating with other CAVs using communication and sensing capabilities. This work discusses a partially-decentralized collaborative decision making approach for a coalition of CAVs in the presence of a misbehaving vehicle. A novel algorithm based on Monte Carlo Tree Search (MCTS) is presented for the CAV's planning problem of deriving mitigation action plans. This algorithm reduces the size of the search tree exponentially to overcome the computational limitations of MCTS for large action-agent sets. V2V communication is used to ensure that mitigation action plans chosen by coalition members are conflict-free when possible. The proposed method is evaluated for several conflict scenarios showing that the system can effectively avoid collisions in diverse situations.**

***Index Terms*—Collaborative collision avoidance, Cooperative collision avoidance, Monte Carlo tree search, Cooperative decision making**

I. Introduction

In the autonomous driving domain, it is not uncommon that Connected and Automated Vehicles (CAVs) misbehave due to technical problems or unexpected environmental changes (e.g., roadblocks, torrential rains, etc). Such misbehavior may result in collisions with other vehicles. Although CAVs are equipped with various sensors and systems that allow them to have knowledge about their surroundings and the road network, taking action to avoid collision individually may result in new conflicts.

In this paper, we consider the problem of collaborative action planning for a coalition of CAVs in the presence of a misbehaving vehicle to avoid collisions. A vehicle coalition is a group of CAVs driving together for mutual support and information sharing. Vehicle coalitions have several benefits: they improve roadway safety, decrease traffic congestion and reduce energy consumption. The problem of cooperative action planning with collision avoidance has been studied in the literature [1]. AI-based and non-AI based methods have been proposed. AI-based approaches generally formulate the planning process as a Markov Decision Process (MDP). MDPs require enumeration over states and actions making them impractical for large state spaces such as the CAV planning problem. The Monte Carlo Tree Search (MCTS) has recently shown promising results in problems with very large state spaces [2].

We present Approximate Simultaneous Move MCTS (ASM-MCTS), an MCTS-based algorithm for CAV planning. A plan is defined as a *sequence of actions* for a *planning horizon* and takes into account information provided by other CAVs in the coalition. We propose a two-step decision-making method. In the first step, each CAV executes ASM-MCTS to derive mitigation action plans that avoid collisions. In the second step, the coalition leader checks the individual coordinated action plans to confirm that they are conflict-free. The proposed ASM-MCTS algorithm improves on existing MCTS-based algorithms for CAVs by exponentially reducing the size of the search tree.

In the next section, we review related work. In Sections III and IV, we formalize the problem. In Section V, we present ASM-MCTS. In Section VI, we give an overview of a multi-agent traffic simulation system used to implement our algorithms, and provide illustrative scenarios.

II. Related works

Many CAV collision avoidance methods have been proposed in the literature. We classify these methods according to the decision-making (i.e., planning) approach, namely *non AI-based optimization* and *AI-based optimization* and further categorize them as *centralized* or *decentralized*.

A. Non AI-based approaches

Centralized: Most conventional optimization approaches formulate the trajectory planning problem for CAVs as a single, global optimization problem solved by a centralized computer. [3] presents a Mixed Integer Linear Programming (MILP) approach to find a non-colliding CAV action sequence for each CAV. First, each CAV derives a *motion tree*, i.e., a structure of possible trajectories and shares it with the centralized computer. The centralized computer formulates the trajectory generation problem as a MILP optimization problem for all vehicles' motion trees by including collision constraints. The result is an *optimal edge sequence* for each vehicle referring to a trajectory.

Decentralized: In [4] each CAV derives all possible maneuvers, and associates a cost for each maneuver using a cost function. The list of maneuvers and associated costs are shared with all other CAVs using V2V communication.

978-1-7281-9001-3/20/$31.00 ©2020 IEEE

Each CAV then applies a brute force algorithm that performs an exhaustive search on all maneuver combinations for all CAVs to find the collision-free combination with the lowest cost. Similarly to centralized solutions, given the exhaustive computations carried out by each CAV, this approach is not scalable.

B. AI based approaches

In AI based approaches, each CAV performs decision-making by learning the value of an action in its current state. Systems in these category are decentralized, i.e., each CAV determines the value of available actions individually using implicit or explicit cooperation with other CAVs. [5] proposes a decision-making approach based on Monte-Carlo tree search (MCTS). A CAV has access to full knowledge of the road network and and acquires data about other CAVs using perception modules. Each CAV performs an MCTS algorithm which includes the classical four steps, i.e., *selection*, *expansion*, *simulation* and *backpropagation* [6]. In the search tree, a node refers to a joint state of all CAVs, an edge refers to a single CAV action, and a sequence of edges from the root node corresponds to actions taken by all CAVs. At the end of the algorithm, each CAV executes the action having the highest action value at the root node. The approach was evaluated on scenarios involving up to three cooperative vehicles. The main limitation of this approach is scalability. For larger number of CAVs, the MCTS tree grows exponentially in depth and it becomes computationally expensive to find valid solutions. [7] applies MCTS to cooperative driving formulated as a Semi Markov Decision Process (SMDP). Similarly to [5], each CAV executes the classical four steps of MCTS. A node in the tree refers to a joint state of all CAVs and an edge refers to a joint action. To reduce the tree depth, this work uses *macro actions* defined as sequences of primitive actions and represented using its policy. In the MCTS stages, all simulated CAVs select macro actions according to their policies until primitive actions are chosen and the next joint state is determined. In the *backpropagation* stage, the value of each node of the tree is updated using the cumulative reward. Each CAV chooses a macro action according to the maximum reward at the root of the tree. The approach was evaluated on scenarios involving up to three CAVs. The main limitation of this approach is scalability as the MCTS tree grows exponentially in width for larger number of CAVs.

In this paper we present a partially-decentralized collaborative collision avoidance method for CAVs in a coalition. We propose a hierarchical decision-making approach where decision-making is performed at the vehicle level and at coalition leader level. Cooperation is achieved explicitly among CAVs in a coalition using V2V communication. The unique contributions of our approach are:

1) A novel anytime MCTS-based algorithm for the planning problem of individual CAV that exponentially reduces the size of the search tree both in depth and in width.
2) A solution that finds a non-colliding *sequence of actions* for each CAV for a *planning horizon*.

III. Model Definition

In the remainder of this paper the terms *agent* and *CAV* are used interchangeably. We assume that a set of $\mathcal{N}$ CAVs are navigating on a highway and forming a coalition $\mathcal{C}$. A coalition $\mathcal{C}$ is formed when a finite set of CAVs are in close proximity. $\mathcal{C}$'s agents are called *coalition members*. One coalition member is assigned the role of a *leader* and is responsible for coalition management. Coalition formation and leader assignment algorithms are adapted from our previous work [8].

A CAV is defined by its ID $i \in \mathcal{N}$ and a state vector $s^i = [p^i, v^i, l^i, \theta^i, x^i]$ where p^i is the CAV's position coordinates, v^i is its velocity, l^i is its current lane, θ^i is its orientation, and x^i is the size of the vehicle containing length and width values. A coalition *joint state* $s = \{s^i\}_{i \in \mathcal{C}}$ is the set of states of all coalition members.

Each CAV can perform the actions listed in Table I. The effect of taking an action a_t^i at time t is determined using the trajectory of i, denoted as $\tau_{a_t^i}^i$. $\tau_{a_t^i}^i$ is computed using velocity, position and orientation. A *mitigation action plan* α_h^i is a sequence of h consecutive actions that is defined by a CAV i in case of perceived danger or warning related to a misbehaving vehicle. It is defined as $\alpha_h^i = \{a_{t_k}^i\}_{k=1}^h$, where t_k is the start of the execution of the mitigation plan's k'th action. h is known as the *planning horizon*. An action is performed over a duration denoted Δt.

Action	Condition	Description
$maintain$	-	Maintain the current velocity
$accel$	-	Accelerate with a fixed acceleration value α_{acc}
$decel$	-	Decelerate with a fixed deceleration value α_{dec}
cll	Currently not in the left most lane	Change lane to the left lane of the current lane
clr	Currently not in the right most lane	Change lane to the right lane of the current lane

TABLE I
CAV Actions

Based on current CAV technologies, we assume that each CAV is equipped with perception sensors such as ultrasonic sensors, radars and Lidar which provide short (i.e., less than 2 meters) and long range (i.e., up to 200 meters) sensing data. CAVs in a coalition use V2V communication to continuously exchange information with each other. The information exchanged depends on the CAV role in the coalition (i.e., member or leader) and includes a CAV's state s^i, the coalition joint state $s = \{s^i\}_{i \in \mathcal{C}}$, a warning or an action plan α_h^i.

IV. Approach

We formulate the problem of decentralized collaborative planning for communicating CAVs as a Multi-agent Markov Decision Process (MMDP) [9]. Unlike a conventional MMDP

where each agent considers the immediate reward for joint actions, our approach focuses on maximizing the overall coalition reward based on CAVs' action plans for a horizon. Unlike decentralized planning with implicit cooperation using macro-actions [7], our approach uses explicit V2V communication to coordinate between all members of a coalition.

A. Extension to MMDP

We formulate our problem as an MMDP and define the planning problem as a tuple $\langle \mathcal{C}, S, A, T, R, \gamma \rangle$, where

- $\mathcal{C}$ is a coalition of n CAVs.
- S represents the joint state space for the CAVs in $\mathcal{C}$. $S = \times S^i$ where S^i is the state space for CAV i.
- A represents the joint action space for the actions of CAVs in $\mathcal{C}$. $A = \times A^i$ where A^i is the set of actions that i can perform.
- $T : S \times A \times S \rightarrow [0, 1]$ is the transition function where $T(s, a, s')$ specifies the probability of the system transitioning to state s' when performing joint action a in state s.
- $R : S \times A \times S \rightarrow \mathbb{R}$ is the reward function with $R(s, a, s')$ specifying the reward received when executing joint action a in state s and transitioning to the state s'.
- γ is a discount factor controlling the contribution of future rewards on the current state.

In our approach, in the event that a misbehaving vehicle m is detected, each CAV i independently solves MMDP using the communicated joint state s to estimate Q^* values for the next h consecutive actions, and possible start states for those actions. Each CAV i derives all possible mitigation action plans for horizon h, and ranks the plans using the Q^* values. Each CAV i sends the ranked plans to the coalition leader using V2V communication. The coalition leader analyzes the prioritized mitigation action plans, and deliberates to find, an action plan α_h^i for each CAV i such that the collision constraints are satisfied when possible. The collision constraints for any two coalition members i and j and a misbehaving vehicle m are formulated as: $\tau^i_{a^i_{t_k}} \cap \tau^j_{a^j_{t_k}} = \phi$, $\tau^i_{a^i_{t_k}} \cap \tau^m = \phi$ and $\tau^j_{a^j_{t_k}} \cap \tau^m = \phi$.

B. Extension to SM-MCTS

Monte Carlo Tree Search is a simulation-based search algorithm to find an optimal decision with the help of random samples and is commonly applied to solve MDPs. In the MCTS algorithm, a search tree is built where each node represents a state of the MDP and an an outgoing edge represents an action taken from that state. Each iteration of MCTS consists of four stages [6]: *Selection*, *Expansion*, *Simulation* and *Backpropagation*. Selection of nodes is performed starting from the root node until a leaf node is found. Expansion of this leaf node is performed using all possible actions available at that node. After selecting one of the expanded children, simulation is performed unsing a default policy until the planning horizon. Collected reward at the end of the simulation is backpropagated to the root node. Since MCTS is an anytime algorithm, it can be stopped after either a fixed number of k iterations or by checking a convergence criterion.

MCTS applied to problems in which multiple agents in the environment perform their actions at the same time is known as Simultaneous Move MCTS (SM-MCTS). In SM-MCTS, each node represents a joint state of all agents $s \in S$ and each edge represents a joint action for all agents $a \in A$. Thus the branching factor of an MCTS tree becomes the *number of agents* times the *number of possible actions per agent*. This exponential growth limits scalability of SM-MCTS [7]. We present Approximate Simultaneous Move MCTS (ASM-MCTS), an approach that reduces the branching factor of MCTS trees and is practical and scalable for MMDP.

V. Algorithms

To solve the proposed MMDP discussed in section IV-A, we present the Approximate Simultaneous Move MCTS (ASM-MCTS) algorithm. As mentioned in Section IV-B, ASM-MCTS extends and modifies SM-MCTS to make it practical and scalable for MMDP. To this effect, we propose to exponentially reduce the number of nodes in the tree by reducing the branching factor. We achieve this by only using individual actions instead of joint actions to represent an edge in the tree. We modify each of the four MCTS stages. In the remainder of this section, we consider that a CAV i in coalition C receives a warning about a misbehaving vehicle and needs to define mitigation action plans for horizon h taking into account each of the coalition member's states. A key component of the solution is the construction of a search tree.

A. ASM-MCTS search tree structure

In our approach, a node n in CAV i's search tree includes the following parameters at time t_k which represents the time at which action $a^i_{t_k}$ from the mitigation action plan α^i_h is executed (see Section III): $n.s_{t_k}$ is the joint state of the coalition at time t_k, $n.s^i_{t_k}$ is the individual state of CAV i at time t_k, $n.a^i_{t_{k-1}}$ is the individual action, $n.\nu$ stores a value that corresponds to the sum of rewards of all simulations that include node n, $n.cnt$ is visit count of node n, and $n.n_{parent}$ is the parent node of node n. The search tree's root node n_μ is initialized with state $n_\mu.s^i_{t_1}$ as well as the coalition joint state $n_\mu.s_{t_1}$. For root node n_μ, $n_\mu.\nu$ and $n_\mu.cnt$ are initialized with zero whereas $n_\mu.a^i_{t_0}$ and $n_\mu.n_{parent}$ are initialized with *null*. An edge to a node n from parent node n_{parent} corresponds to vehicle i's individual action a^i. The most important functions of our proposed algorithm are outlined in Algorithm 1. In following sections, we describe each of the four stages of a single iteration of ASM-MCTS.

B. ASM Selection policy

In order to select an action $n_{child}.a^i_{t_k}$, i.e., an edge from node n to a child node n_{child}, we use the UCB1 algorithm [6]. $n_{child}.a^{\mathcal{C}\setminus i}_{t_k}$ are randomly sampled from the action set available to CAVs other than CAV i. $n_{child}.a^i_{t_k}$ combined with randomly sampled $n_{child}.a^{\mathcal{C}\setminus i}_{t_k}$ forms a joint action set $n_{child}.a_{t_k}$, which is then applied to the joint state set $n.s_{t_k}$ of the parent node n to derive the joint state set $n_{child}.s_{t_{k+1}}$ of the child node.

Algorithm 1 ASM-MCTS

Input: n_μ, A

while maximum number of iterations are not executed **do**
  $\langle n_{leaf}, \mathcal{R} \rangle \leftarrow$ ASM-SelectionPolicy(n_μ, A)
  if $n_{leaf}.cnt \neq 0$ **then**
    $\langle n_{leaf}, \mathcal{R} \rangle \leftarrow$ ASM-ExpansionPolicy$(n_{leaf}, A, \mathcal{R})$
    Add n_{leaf} to the MCTS Tree
  end if
  $\mathcal{R} \leftarrow$ ASM-SimulationPolicy$(n_{leaf}, \mathcal{R})$
  ASM-BackpropagationPolicy$(n_{leaf,\mathcal{R}})$
end while

Algorithm 2 ASM-Selection Policy

Input: n_μ, A
Output: $n_{leaf}, \mathcal{R}$

$n \leftarrow n_\mu$
$\mathcal{R} \leftarrow 0$
repeat
  $n_{child}.a^i_{t_k} \leftarrow$ UCTAction(n, i)
  $n_{child}.a^{\mathcal{C}\setminus i}_{t_k} \leftarrow$ RandomSelection$(A^{\mathcal{C}\setminus i})$
  $\mathcal{R} \leftarrow \mathcal{R} +$ ComputeReward$(n, n_{child}.a^i_{t_k}, n_{child}.a^{\mathcal{C}\setminus i}_{t_k})$
  $n_{child}.s_{t_{k+1}} \leftarrow$ ComputeState$(n.s_{t_k}, n_{child}.a_{t_k})$
until n is a leaf node

C. Expansion policy

When expanding a node n in the tree, individual state $n.s^i_{t_k}$ and the valid individual actions are used for expansion. Since maximum number of actions for any CAV is 5, up to five child nodes are added for each valid value of $a^i_{t_k}$. $a^{\mathcal{C}\setminus i}_{t_k}$ is initialized by performing random sampling from $A^{\mathcal{C}\setminus i}$. Combining $a^i_{t_k}$ and $a^{\mathcal{C}\setminus i}_{t_k}$ forms $n_{child}.a_{t_k}$ for the child node. Each of the child nodes n_{child} is initialized with $n_{child}.s_{t_{k+1}}, n_{child}.s^i_{t_{k+1}}, n_{child}.a^i_{t_k}, n_{child}.\nu$, and $n_{child}.cnt$, where the joint state $n_{child}.s_{t_{k+1}}$ is derived using $\langle n.s_{t_k}, n_{child}.a_{t_k} \rangle$ and the individual state $n_{child}.s^i_{t_{k+1}}$ is derived using $\langle n.s^i_{t_k}, n_{child}.a^i_{t_k} \rangle$. $n_{child}.\nu$ and $n_{child}.cnt$ are initialized with zero values.

Algorithm 3 ASM-Expansion Policy

Input: $n, A, \mathcal{R}$
Output: $n_{leaf}, \mathcal{R}$

$a^{\mathcal{C}\setminus i}_{t_k} \leftarrow$ RandomSelection$(A^{\mathcal{C}\setminus i})$
for $a^i_{t_k}$ in A^i **do**
  $a_{t_k} \leftarrow \langle a^i_{t_k}, a^{\mathcal{C}\setminus i}_{t_k} \rangle$
  $s_{t_{k+1}} \leftarrow$ nextState$(n.s_{t_k}, a_{t_k})$
  $n_{child} \leftarrow$newNode$(s_{t_{k+1}}, a^i_{t_k}, s^i_{t_{k+1}})$
  addChild$(n.children, n_{child})$
end for
$a^i_{t_k} \leftarrow$RandomSelection(A^i)
$\mathcal{R} \leftarrow \mathcal{R} +$ ComputeReward$(n, a^i_{t_k}, a^{\mathcal{C}\setminus i}_{t_k})$
$n_{leaf} \leftarrow$SelectNode$(n.children, a^i_{t_k})$

D. ASM Simulation policy

When we reach an unvisited leaf node in the tree performing selection procedures, we perform simulation steps until the planning horizon or until a terminal node is reached. A terminal node is a node where collision is detected between any two CAVs or a CAV and the misbehaving vehicle. In each simulation step, joint actions are randomly sampled and joint states are derived from previous joint state and currently sampled joint action. Value of cumulative reward is updated in each simulation step.

Algorithm 4 ASM-Simulation Policy

Input: $n, \mathcal{R}$
Output: $\mathcal{R}$

repeat
  $a_{t_k} \leftarrow$ RandomSelection(A)
  $s_{t_{k+1}} \leftarrow$ ComputeState$(n.s_{t_k}, a_{t_k})$
  $\mathcal{R} \leftarrow \mathcal{R} +$ ComputeReward(n, a_{t_k})
  $n_{child} \leftarrow$ ExpandSingleChildNode(n, a_{t_k})
  $n \leftarrow n_{child}$
until n is terminal node **or** horizon is reached

E. ASM Backpropagation policy

The cumulative reward received at the end of the simulation procedure is backpropagated from the leaf node to the root node. Node values and visit counts of nodes along the path from the leaf node to the root node are updated. Additionally, all the joint state values s for all nodes along the reverse path are set back to $null$ in their state parameter vectors except at the root node.

Algorithm 5 ASM-Backpropagation Policy

Input: $n, \mathcal{R}$

while $n \neq n_\mu$ **do**
  $n.cnt \leftarrow n.cnt + 1$
  $n.\nu \leftarrow n.\nu + \mathcal{R}$
  $n.s_{t_k} \leftarrow$ null
  $n \leftarrow n.n_{parent}$
end while

After maximum number of iterations, each expanded node n in the tree contains $n.\nu$ and $n.cnt$ values. Estimated value of state-action value function at node n for an individual CAV $Q^*(n_{parent}.s^i_{t_{k-1}}, n.a^i_{t_{k-1}})$ can be estimated using following equation:

$$Q^*(n_{parent}.s^i_{t_{k-1}}, n.a^i_{t_{k-1}}) \approx \frac{n.\nu}{n.cnt} \tag{1}$$

MCTS tree is stripped of extra variables. Only individual states, individual actions and estimated Q^* values are retained at each node of the tree. This tree is then shared with the coalition leader using V2V communication.

F. Beam search for action selection by coalition leader

The coalition leader performs the action selection for CAVs using the *beam-search* method. Beam search with the beam size = 1 is employed to ensure that the best possible sequence of actions for each coalition member satisfy the collision constraints, when possible. Starting from time step 0, and at each time step, the algorithm picks the best actions according to Q^* values for each coalition member and checks that the joint action set does not introduce any collisions. If a collision between two CAVs is identified, then the next best plan is picked for one of the two vehicles and the collision check is performed again. After finding the best possible sequence of actions for all coalition members, the plans are shared with each coalition member using V2V communication.

VI. Illustrative Scenarios

ASM-MCTS was implemented in MATISSE 3.0 [10], a microscopic multi-agent based traffic simulation system. MATISSE offers the features necessary to simulate scenarios for Intelligent Transportation Systems including CAVs (e.g., V2V, V2I, I2I, sensing devices, etc). In MATISSE, a CAV is modeled as an autonomous decision-making agent that has the capability to dynamically sense its surroundings through simulated sensors while communicating with other CAVs. Simulated sensors include long-range radar and 360-degree LIDAR.

In order to examine the behavior of our proposed algorithm in real world scenarios, we perform several simulations in MATISSE [1]. CAVS agents navigate in a coalition formation on a three lane highway. In the scenarios, we consider different types of misbehaviors and different positions of misbehaving vehicles with respect to the coalition. We use c to represent a MATISSE simulation cycle. A vehicle's decision-making process in case of detection/warning of a misbehaving vehicle includes 20,000 iterations of ASM-MCTS and a planning horizon of 4. A time step t as defined is Section III corresponds to one simulation cycle. We use Δt=10.

A. Misbehaving vehicle accelerating from behind a coalition

As depicted in figure 2, Scenario 1 considers a coalition of 6 vehicles, led by vehicle $V4$. Each of the vehicles in the coalition are navigating at the cruising speed of 2 meters/cycle. At cycle c_n, vehicle $V5$ detects that vehicle Vm is misbehaving, coming from behind at the speed of 6 meters/cycle. $V5$ immediately sends the estimated state parameters s^m for the misbehaving vehicle to the coalition members. At cycle c_{n+1}, after receiving the misbehaving alert, each coalition member executes Algorithm 1 $ASM{-}MCTS(n_\mu, A)$ to compute their individual MCTS tree. The root node n_μ of each vehicle tree contains the coalition current joint state, i.e., the current states of all coalition members. The joint action space A contains possible actions for each coalition member for each possible state up to the planning horizon. An ASM-MCTS iteration's expansion step, i.e., $ASM - ExpansionPolicy(n, A, \mathcal{R})$ in Algorithm 3, only considers the actions that a vehicle can perform. For instance, in Figure 2A, there are four possible actions for $V3$ a_{accel}, a_{decel}, $a_{maintain}$ and a_{clr}, and four child nodes are added to $V3$'s root node during ASM-MCTS execution performed by V3. Actions for coalition members other than $V3$ are randomly sampled in selection policy (step 5 of Algorithm 2) as well as expansion policy (step 1 of Algorithm 3). Whereas in simulation step (step 2 of Algorithm 4), actions for all coalition members including $V3$ are randomly sampled. After $ASM - MCTS$ in Algorithm 1 is executed, each vehicle shares its derived MCTS tree with the leader $V4$. At cycle c_{n+2}, $V4$ receives the MCTS trees from all vehicles in the coalition, and derives final action plans for each coalition member. Table II shows the final actions chosen by $V4$ for each coalition member in Scenario 1. At cycle c_{n+3}, $V5$ changes its lane to second lane to avoid collision with the misbehaving vehicle. $V4$ decelerates to allow $V6$ to change lane for $V6$'s next action. $V2$ accelerates to gain safe speed and avoid future collisions with the misbehaving vehicle. At cycle c_{n+13}, $V3$ needs to change its lane to second lane to avoid the collision with the misbehaving vehicle. $V6$ also changes its lane to second lane to avoid collision with $V3$. $V2$ continues performing acceleration action. At cycle c_{n+23}, $V5$ decelerates whereas $V1$ and $V2$ accelerate. At cycle v_{n+33}, $V2$ changes its lane to second lane to avoid collision with the misbehaving vehicle in the first lane. $V4$ and $V5$ change their lanes as well. These seemingly unnecessary actions by $V4$ and $V5$ are performed as they have higher Q^* values in the respective MCTS trees of $V4$ and $V5$.

Agent	Planned Action Sequences
V_1	$a^1_{maintain}$, $a^1_{maintain}$, a^1_{accel}, a^1_{accel}
V_2	a^2_{accel}, a^2_{accel}, a^2_{accel}, a^2_{clr}
V_3	$a^3_{maintain}$, a^3_{clr}, $a^3_{maintain}$, a^3_{accel}
V_4	a^4_{decel}, $a^4_{maintain}$, $a^4_{maintain}$, a^4_{cll}
V_5	a^5_{clr}, $a^5_{maintain}$, a^5_{decel}, a^5_{cll}
V_6	$a^6_{maintain}$, a^6_{clr}, $a^6_{maintain}$, a^6_{accel}

TABLE II
Scenario 1: Final actions chosen by coalition leader

B. Misbehaving vehicle breaks in front of a coalition

In this scenario, we consider a coalition of 5 vehicles. Misbehaving vehicle is detected to be breaking in front of the coalition by vehicle V3 at cycle c_n. Two simulation cycles are used by coalition members in collaborative decision making task. At cycle c_{n+3}, V3 immediately changes lane to the third lane to avoid the collision with the misbehaving vehicle. V4 also changes lane to the third lane at cycle c_{n+3} to avoid the future collision with the misbehaving vehicle. V1 and V2 perform two consecutive acceleration actions from cycle c_{n+13}

[1]Demos available at www.utdallas.edu/~dhruv/CAVdemos

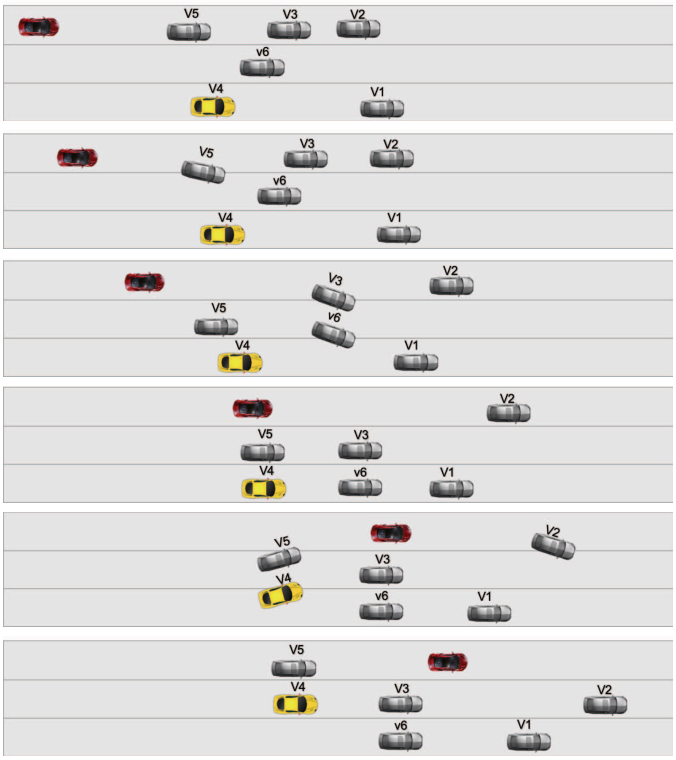

Fig. 1. Scenario 1: Misbehaving vehicle accelerating from behind a coalition

to c_{n+33} while other coalition members maintain their speeds. V2 changes its lane to the second lane at cycle c_{n+33}. At the end, all coalition members have avoided collisions with the misbehaving vehicle.

C. Misbehaving vehicle accelerates within a coalition

In this scenario, we consider a coalition of 7 vehicles. Misbehaving vehicle is part of this coalition. It suddenly starts accelerating in the middle of the coalition endangering other coalition members at cycle c_n. Figure **1** depicts this scenario. Two simulation cycles are used by coalition members in collaborative decision making task. At cycle c_{n+3}, vehicle V3 immediately needs to change its lane to third lane to avoid the collision with the misbehaving vehicle. V2 accelerates to allow enough space for lane change from cycle c_{n+3} to c_{n+13}. During this interval, V1 also decides to accelerate and V6 decides to change its lane to first lane. At cycle c_{n+13}, V5 and V6 decide to change their lanes to second lane. At cycle c_{n+23}, V1 decides to change its lane to second lane while maintaining safe speed such that it does not collide with the misbehaving vehicle. V5 also decides to change its lane to the third lane at c_{n+23}. For the last action, all coalition members just maintain their speeds and avoid collision with the misbehaving vehicle.

VII. Conclusion

In this paper, we presented an approach for collaborative decision-making of CAVs in the presence of a misbehaving vehicle. The decision-making is achieved in two steps. First, each CAV executes ASM-MCTS to derive mitigation action plans, then the coordinated plans are sent to the coalition leader to ensure that they are conflict-free. The proposed ASM-MCTS algorithm reduces the search tree by decreasing the branching factor. We implemented ASM-MCTS in MATISSE and illustrated the proposed approach through multiple simulation scenarios.

The presented work only considers one coalition. Our goal is to expand our approach to consider multiple coalitions and compare its performance to other solutions implemented in MATISSE.

References

[1] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and decision-making for autonomous vehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 187–210, 2018.

[2] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, "Mastering the game of go without human knowledge," *nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[3] T. Kessler and A. Knoll, "Cooperative multi-vehicle behavior coordination for autonomous driving," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 1953–1960.

[4] M. Duering and P. Pascheka, "Cooperative decentralized decision making for conflict resolution among autonomous agents," in *2014 IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA) Proceedings*. IEEE, 2014, pp. 154–161.

[5] D. Lenz, T. Kessler, and A. Knoll, "Tactical cooperative planning for autonomous highway driving using monte-carlo tree search," in *2016 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2016, pp. 447–453.

[6] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in games*, vol. 4, no. 1, pp. 1–43, 2012.

[7] K. Kurzer, C. Zhou, and J. M. Zöllner, "Decentralized cooperative planning for automated vehicles with hierarchical monte carlo tree search," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 529–536.

[8] H. Manoochehri and R. Wenkstern, "Dynamic coalition structure generation for autonomous connected vehicles," in *2017 IEEE International Conference on Agents (ICA)*. IEEE, 2017, pp. 21–26.

[9] C. Boutilier, "Sequential optimality and coordination in multiagent systems," in *IJCAI*, vol. 99, 1999, pp. 478–485.

[10] B. Torabi, M. Al-Zinati, and R. Z. Wenkstern, "Matisse 3.0: A large-scale multi-agent simulation system for intelligent transportation systems," in *International Conference on Practical Applications of Agents and Multi-Agent Systems*. Springer, 2018, pp. 357–360.