Chapter 6

Example 2: Collision Avoidance on a Single Track

This chapter is similar to Chapter 5; instead of an overspeed protector, here we present a collision protector for the VEHICLES automaton. We define the protector CL-PROT that guarantees that none of the vehicles collide, provided that they are all abiding by the speed limit. The CL-PROT protector is defined as the composition of n separate copies of another protector called CL-PROT-SOLO$_i$, one copy for each vehicle $i \in I$. Each of the CL-PROT-SOLO$_i$ protectors, for $i \in I$, is an implementation of a particular instantiation of the abstract protector automaton of Section 3.2 and guarantees that the vehicle i does not collide into any of the vehicles it trails.

6.1 Protection System CL-PROT-SOLO$_i$

The CL-PROT-SOLO$_i$ automata are vehicle-wise collision protectors and individually guarantee that the vehicle i does not collide into any of the vehicles it trails, provided that all vehicles are abiding by the speed limit and that all other vehicles $i' \in I, i' \neq i$, do not collide into any of the vehicles they respectively trail. Each of the CL-PROT-SOLO$_i$ protectors, for $i \in I$, is an implementation of the abstract protector of Section 3.2 specialized to particular definitions of the parameters PP, S, R, G, j, and d.

The physical plant automaton, PP, is defined to be the VEHICLES automaton of Figure 4.1. The port j and the sampling period d are defined to be the port and sampling period with which the protector CL-PROT-SOLO$_i$ communicates with the VEHICLES automaton and are assumed arbitrary. The set of "good" states G is defined to be the set of states in which the

vehicle i has not collided into any of the other vehicles, *i.e.*, $G = VALID - P_{collided(i)}$. In this chapter, we use the notation G_i to refer to this definition of the set G. The set R is defined to be the set $R = P_{not\text{-}overspeed} \bigcap \left(\bigcap_{i' \in I, i' \neq i} G_{i'} \right)$. This definition restricts the states of the VEHICLES automaton to states in which all of the vehicles are abiding by the speed limit and in which each of the remaining vehicles has never collided into any other vehicle. The set of states S is defined to be the set *safe* defined in Section 3.2.1; that is, the set of states of the PP automaton for which a single input action of PP on port j can guarantee that, provided no new input actions on port j are allowed, all subsequently R-reachable states will be in G. Once again, the definition of the set *safe* is specialized to the above definitions of the automaton PP, the sets R and G, and the port j. In this chapter, we use the notation R_i and S_i to refer to the above definitions of the sets R and S.

The CL-PROT-SOLO$_i$ protector automaton is an implementation of the abstract protector automaton Abs(VEHICLES, S_i, R_i, G_i, j, d). More precisely, as is the case for the abstract protector Abs_j, we define the CL-PROT-SOLO$_i$ automaton to be the composition of a sensor and a discrete controller automaton. These automata are implementations of their abstract equivalents of Figures 3.2 and 3.3, specialized however, to the above definitions of the parameters PP, S, R, G, j, and d. The sensor automaton is precisely the specialization of the sensor automaton of Figure 3.2 to the above definitions of the parameters PP, *etc.* The discrete controller automaton is defined in Figure 6.1.

The braking strategy of the CL-PROT-SOLO$_i$ protector is as follows. The protector instructs the vehicle i to brake if it has a d time unit claim overlap with any of the vehicles it trails; that is, the protector instructs the vehicle i to brake if there exists a vehicle i', for $i' \in I, i' \neq i$, such that the sections of the track claimed by the vehicles i and i' in time d overlap and $x_i < x_{i'}$. The rationale behind this braking strategy is that a collision between two vehicles in the VEHICLES automaton can only be prevented by instructing the trailing vehicle to brake.

It is important to note that the abstract protector automaton Abs(VEHICLES, S_i, R_i, G_i, j, d) complies with the assumptions made about the abstract protector in Section 3.2.1. In particular, since the vehicle position variables, the vehicle velocity variables, and the *collided* variables are output variables of the VEHICLES automaton, the set *safe* is Y_{VEHICLES}-determinable and actions that guarantee safety can be determined from the output variables Y_{VEHICLES} of the VEHICLES automaton (Axioms 3.2.4 and 3.2.5, respectively). Moreover, the sets R_i and G_i are Y_{VEHICLES}-determinable (Axioms 3.2.6 and 3.2.7, respectively) and the set of start states S_i is a subset of the set *safe* (Axiom 3.2.8), since S_i is defined to be the set *safe*.

In Section 3.1 it was shown that the abstract protector Abs_j guarantees that the physical plant PP remains within G starting from S given R. Similarly, the CL-PROT-SOLO$_i$ automaton guarantees that the VEHICLES automaton remains within G_i starting from S_i given R_i. This is shown in the following section.

Figure 6.1 Discrete controller automaton for the protector CL-PROT-SOLO$_i$.

Actions:	Input:	e, the environment action (stuttering)
		$\texttt{snapshot}(y)_j$, for each valuation y of Y_{VEHICLES}
	Output:	$\texttt{brake}(i)_j$
		$\texttt{unbrake}(i)_j$
Variables:	Internal:	$send_j \in \{\texttt{brake}, \texttt{unbrake}, null\}$, initially $null$

Discrete Transitions:

$\texttt{snapshot}(y)_j$
Eff: if $\exists\, i' \in I, i' \neq i$ such that
$y \notin$ *disjoint-claimed-tracks*$(i, i', d) \wedge (y.x_i < y.x_{i'})$
then
$send_j := \texttt{brake}$
else
$send_j := \texttt{unbrake}$

$\texttt{brake}(i)_j$
Pre: $send_j = \texttt{brake}$
Eff: $send_j := null$

$\texttt{unbrake}(i)_j$
Pre: $send_j = \texttt{unbrake}$
Eff: $send_j := null$

Trajectories:
$w.send_j \equiv null$

6.2 Correctness of CL-PROT-SOLO$_i$

The main result to be shown is that CL-PROT-SOLO$_i \leq$ *Abs*(VEHICLES, S_i, R_i, G_i, j, d). Since both CL-PROT-SOLO$_i$ and *Abs*(VEHICLES, S_i, R_i, G_i, j, d) involve the composition of the same sensor automaton with distinct discrete controller automata, Theorem 2.7.4 applies. Therefore, it suffices to show that the discrete controller automaton of CL-PROT-SOLO$_i$ of Figure 6.1 implements the discrete controller automaton *DC*(VEHICLES, S_i, R_i, G_i, j, d) of Figure 3.3. According to Theorem 2.6.1, this follows by showing that there exists a simulation relation between the states of the discrete controller automaton of CL-PROT-SOLO$_i$ and the discrete controller automaton *DC*(VEHICLES, S_i, R_i, G_i, j, d). We first give some useful set definitions, then prove some lemmas, and finally show the existence of such a simulation relation. The correctness proof follows the steps of the correctness proof of Section 5.2.

In this section, we use the notation *future*$_i$, *safe*$_i$, *very-safe*$_i$, and *delay-safe*$_i$ to denote the specialization of the function *future*, the sets *safe* and *very-safe*, and the function *delay-safe*, which are defined in Section 3.2.1, to the automaton VEHICLES, the sets R_i and G_i, and

Table 6.1 Sets used in the correctness proof of CL-PROT-SOLO$_i$.

$W_i \subseteq VALID$, for $i \in I$, defined by

$$W_i = \{p \in R_i \cap G_i \mid \nexists\, i' \in I, i' \neq i : p.O_i \cap p.O_{i'} \neq \emptyset \ \wedge\ p.x_i < p.x_{i'}\}$$

$V_i \subseteq VALID$, for $i \in I$, defined by

$$V_i = W_i \cap P_{B_{ij}}$$

$T_i(t) \subseteq VALID$, for $i \in I$, and $t \in \mathbb{R}^{\geq 0}$, defined by

$$T_i(t) = \{p \in R_i \cap G_i \mid \nexists\, i' \in I, i' \neq i : p.C_i(t) \cap p.C_{i'}(t) \neq \emptyset \ \wedge\ p.x_i < p.x_{i'}\}$$

the port j of the CL-PROT-SOLO$_i$ protector. Moreover, since the environment action of the VEHICLES automaton is stuttering, its consideration is omitted in all inductive proofs involving the PP automaton.

We proceed by defining several sets that are used in the correctness proof of the protector CL-PROT-SOLO$_i$. For reference, their formal definitions appear in Table 6.1.

Let W_i be the subset of $R_i \cap G_i$ comprised of the states in which the section of the track owned by the vehicle i does not overlap the section of track owned by any of the vehicles it trails; that is, for every state p in W_i, $p \in R_i \cap G_i$ and there does not exist $i' \in I, i' \neq i$ such that $p.O_i \cap p.O_{i'} \neq \emptyset$ and $p.x_i < p.x_{i'}$.

Let V_i be the subset of W_i comprised of the states in which the protector j is requesting the vehicle i to brake; that is, $V_i = W_i \cap P_{B_{ij}}$.

Let $T_i(t)$, where $t \in \mathbb{R}^{\geq 0}$, be the subset of $R_i \cap G_i$ comprised of the states in which the section of the track claimed in time t by the vehicle i does not overlap the section of the track claimed in time t by any of the vehicles it trails; that is, for every state p in $T_i(t)$, $p \in R_i \cap G_i$ and there does not exist $i' \in I, i' \neq i$ such that $p.C_i(t) \cap p.C_{i'}(t) \neq \emptyset$ and $p.x_i < p.x_{i'}$.

Lemma 6.2.1 *For all $t, t' \in \mathbb{R}^{\geq 0}$, $t \leq t'$, the following hold:*

1. $T_i(t) \subseteq W_i \subseteq G_i$.
2. $V_i \subseteq W_i \subseteq G_i$.
3. $T_i(t') \subseteq T_i(t)$.
4. $T_i(0) = W_i$.

Proof: Follow directly from the definitions of the sets G_i, W_i, and $T_i(\tau)$, where $\tau \in \mathbb{R}^{\geq 0}$, and Lemma 4.4.2. ■

In the following three lemmas, we show that any state R_i-reachable from a state in V_i through an execution fragment that involves no input actions on port j, is in V_i. In the first lemma, we show that if the final state of such an execution fragment is in G_i and the section of track owned by the vehicle i has not grown since the beginning of the execution fragment, then the final state of the execution fragment is in V_i. In the second lemma, we show that the final state of any such execution fragment is in G_i and the section of track owned by the vehicle i does not grow throughout the execution fragment. Finally, the third lemma combines these two results and states formally the desired property.

Lemma 6.2.2 *Let $p \in V_i$ and $p' \in future_i(p, \mathbb{R}^{\geq 0})$. If $p' \in G_i$ and $p'.O_i \subseteq p.O_i$ then $p' \in V_i$.*

Proof: We need to show that $p' \in R_i \cap G_i \cap P_{B_{ij}}$ and that there does not exist $i' \in I, i' \neq i$ such that $p'.O_i \cap p'.O_{i'} \neq \emptyset$ and $p'.x_i < p'.x_{i'}$. We consider these conditions by cases:

1. $p' \in R_i$.

   This is the case because the function $future_i(p, \mathbb{R}^{\geq 0})$ only considers R_i-reachable states.

2. $p' \in G_i$.

   This is true by assumption.

3. $p' \in P_{B_{ij}}$.

   Since $p \in P_{B_{ij}}$, it is the case that $p.brake\text{-}req(i,j) = \texttt{True}$. Moreover, the $brake\text{-}req(i,j)$ variable can only be set to `False` by an $\texttt{unbrake}(i)_j$ action — an action not allowed by the function $future_i(p, \mathbb{R}^{\geq 0})$. Therefore, it follows that $p' \in P_{B_{ij}}$, as needed.

4. $\nexists\, i' \in I, i' \neq i,$ such that $p'.O_i \cap p'.O_{i'} \neq \emptyset$ and $p'.x_i < p'.x_{i'}$.

   Because $p \in V_i$ we have that for all $i' \in I, i' \neq i$ such that $p.x_i < p.x_{i'}$ it is the case that $p.O_i \cap p.O_{i'} = \emptyset$; that is, for all $i' \in I, i' \neq i$ such that $p.x_i < p.x_{i'}$ it is the case that $\max(p.O_i) < \min(p.O_{i'})$. However, by assumption it is the case that $p'.O_i \subseteq p.O_i$. Therefore, since the vehicle velocities are restricted to be non-negative, it follows that for all $i' \in I, i' \neq i$ such that $p'.x_i < p'.x_{i'}$ it is the case that $\max(p'.O_i) < \min(p'.O_{i'})$. This is sufficient to guarantee that there does not exist $i' \in I, i' \neq i$ such that $p'.O_i \cap p'.O_{i'} \neq \emptyset$ and $p'.x_i < p'.x_{i'}$.

■

Lemma 6.2.3 *For all $p \in V_i$, if $p' \in future_i(p, \mathbb{R}^{\geq 0})$, then $p' \in G_i$ and $p'.O_i \subseteq p.O_i$.*

Proof: Let α be an execution fragment of the VEHICLES automaton of n steps and trajectories, where $n \in \mathbb{N}$, that: starts in a state in V_i, is only comprised of states in R_i, and involves no input actions on port j. Letting p_{init} and p_{final} be the initial and final states of α, respectively, we must show that $p_{final} \in G_i$ and $p_{final}.O_i \subseteq p_{init}.O_i$. The proof is by induction on the length n of the execution fragment α.

For the base case, consider the execution fragment α of length $n = 0$; that is, α is an execution fragment that consists of a single point trajectory and, therefore, $p_{final} = p_{init}$. From Lemma 6.2.1, part 2, and the fact that $p_{init} \in V_i$, it follows that $p_{final} \in G_i$. Moreover, the fact that $p_{final}.O_i \subseteq p_{init}.O_i$ is trivially true.

The inductive step involves showing that if α is an execution fragment of length $n = k + 1$, for some $k \in \mathbb{N}$, then $p_{final} \in G_i$ and $p_{final}.O_i \subseteq p_{init}.O_i$. Let α' be the part of the execution fragment α comprised of the first k steps and trajectories. The induction hypothesis involves the assertion that if p'_{init} and p'_{final} are the initial and final states of α', respectively, then it is the case that $p'_{final} \in G_i$ and $p'_{final}.O_i \subseteq p'_{init}.O_i$. Moreover, from Lemma 6.2.2 it follows that $p'_{final} \in V_i$. Since the final state of α is reached from the final state of α' by a single step or trajectory, the inductive step involves the consideration of all possible steps and trajectories leading from p'_{final} to p_{final}.

In the case of a step, we consider all possible discrete actions by cases:

1. the actions $\texttt{brake}(i)_j$ and $\texttt{unbrake}(i)_j$ are not enabled because α involves no input actions on port j.

2. the $\texttt{brick-wall}(i)$ action sets the velocity of the vehicle i to zero and does not affect the variables $\mathit{collided}(i, i')$, for $i' \in I, i' \neq i$.

   From the induction hypothesis, it is the case that $p'_{final} \in G_i$. Therefore, since the internal action $\texttt{brick-wall}(i)$ does not affect the variables $\mathit{collided}(i, i')$, for $i' \in I, i' \neq i$, it follows that $p_{final} \in G_i$.

   Moreover, since the vehicle velocities are restricted to be non-negative, it is the case that $p_{final}.\dot{x}_i \leq p'_{final}.\dot{x}_i$. From Lemma 4.4.3, part 1, it follows that $p_{final}.O_i \subseteq p'_{final}.O_i$. However, from the induction hypothesis it is the case that $p'_{final}.O_i \subseteq p'_{init}.O_i$. Therefore, since $p_{init} = p'_{init}$, it follows that $p_{final}.O_i \subseteq p_{init}.O_i$, as needed.

3. the actions $\texttt{brake}(i')_{j'}$, $\texttt{unbrake}(i')_{j'}$, for $i' \in I, j' \in J, j' \neq j$, and $\texttt{brick-wall}(i'')$, for $i'' \in I, i'' \neq i$, affect neither the velocity of the vehicle i, nor the variables $\mathit{collided}(i, i''')$, for $i''' \in I, i''' \neq i$.

   From the induction hypothesis, it is the case that $p'_{final} \in G_i$. Therefore, since the actions $\texttt{brake}(i')_{j'}$, $\texttt{unbrake}(i')_{j'}$, for $i' \in I, j' \in J, j' \neq j$, and $\texttt{brick-wall}(i'')$, for $i'' \in I, i'' \neq i$, do not affect the variables $\mathit{collided}(i, i''')$, for $i''' \in I, i''' \neq i$, it follows that $p_{final} \in G_i$.

Moreover, since the input actions $\texttt{brake}(i')_{j'}$, $\texttt{unbrake}(i')_{j'}$, for $i' \in I, j' \in J, j' \neq j$, and the internal actions $\texttt{brick-wall}(i'')$, for $i'' \in I, i'' \neq i$, do not affect the velocity of the vehicle i, it is the case that $p_{final}.\dot{x}_i = p'_{final}.\dot{x}_i$. From Lemma 4.4.3, part 1, it follows that $p_{final}.O_i \subseteq p'_{final}.O_i$. However, from the induction hypothesis it is the case that $p'_{final}.O_i \subseteq p'_{init}.O_i$. Therefore, since $p_{init} = p'_{init}$, it follows that $p_{final}.O_i \subseteq p_{init}.O_i$, as needed.

4. the actions $\texttt{colliding-pair}(i', i'')$, for $i', i'' \in I, i' \neq i''$, and $\texttt{collision-effects}(i''')$, for $i''' \in I$, are not enabled because α is only comprised of states in R_i and $p'_{final} \in V_i$.

In the case of a trajectory, since $p' \in V_i$ and $V_i \subseteq P_{B_{ij}}$, Lemma 4.4.4, part 1, implies that $p_{final}.O_i \subseteq p'_{final}.O_i$. However, from the induction hypothesis it is the case that $p'_{final}.O_i \subseteq p'_{init}.O_i$. Therefore, since $p_{init} = p'_{init}$, it follows that $p_{final}.O_i \subseteq p_{init}.O_i$. Moreover, since $p'_{final} \in G_i$ and the variables $collided(i, i')$, for all $i' \in I, i' \neq i$, remain constant throughout the trajectory, it follows that $p_{final} \in G_i$, as needed. ■

Lemma 6.2.4 $future_i(V_i, \mathbb{R}^{\geq 0}) \subseteq V_i$.

Proof: Follows directly from Lemmas 6.2.2 and 6.2.3. ■

In the following two lemmas, we use Lemma 6.2.4 to show that $V_i \subseteq very\text{-}safe_i$ and $V_i \subseteq delay\text{-}safe_i(t)$, for any $t \in \mathbb{R}^{\geq 0}$, respectively.

Lemma 6.2.5 $V_i \subseteq very\text{-}safe_i$.

Proof: From the definition of *very-safe* in Section 3.2.1, we must show that the condition $future_i(V_i, \mathbb{R}^{\geq 0}) \subseteq G_i$ is satisfied. This follows directly from Lemma 6.2.4 and Lemma 6.2.1, part 2. ■

Lemma 6.2.6 *For any* $t \in \mathbb{R}^{\geq 0}$, *it is the case that* $V_i \subseteq delay\text{-}safe_i(t)$.

Proof: Follows directly from Lemma 6.2.5 and Lemma 3.2.5, part 1. ■

In the next three lemmas and the subsequent corollary, we show that the sets W_i and $safe_i$ are equal. First, we show that any state that is R_i-reachable from a state p in W_i through an execution fragment that involves no input actions on port j and has a limit time equal to zero, is in the set W_i. Then, we show that $W_i \subseteq safe_i$ and $safe_i \subseteq W_i$. Finally, the subsequent corollary states that $W_i = safe_i$.

Lemma 6.2.7 $future_i(W_i, 0) \subseteq W_i$.

Proof: Let α be an execution fragment of the VEHICLES automaton of n steps, where $n \in \mathbb{N}$, that: starts in a state in W_i, is only comprised of states in R_i, involves no input actions on port j, and has a limit time equal to zero. Moreover, let p_{init} and p_{final} be the initial and final states of α, respectively. By induction on the length n of the execution fragment α, we show that $p_{final} \in W_i$.

For the base case, consider the execution fragment α of length $n = 0$; that is, α is an execution fragment that consists of no steps and, therefore, $p_{final} = p_{init}$. Since $p_{init} \in W_i$, it follows that $p_{final} \in W_i$.

The inductive step involves showing that if α is an execution fragment of length $n = k + 1$, for some $k \in \mathbb{N}$, then $p_{final} \in W_i$. Let α' be the part of the execution fragment α comprised of the first k steps. The induction hypothesis involves the assertion that if p'_{final} is the final state of α', then it is the case that $p'_{final} \in W_i$. Since the final state of α is reached from the final state of α' by a single step, the inductive step involves the consideration of all possible steps leading from p'_{final} to p_{final}.

To complete the induction, we consider all possible discrete actions by cases:

1. the actions $\texttt{brake}(i)_j$ and $\texttt{unbrake}(i)_j$ are not enabled because α involves no input actions on port j.

2. the actions $\texttt{brake}(i')_{j'}$, $\texttt{unbrake}(i')_{j'}$, for $i' \in I, j' \in J, j' \neq j$, affect neither the velocity of any of the vehicles, nor the variables $\mathit{collided}(i, i'')$, for $i'' \in I, i'' \neq i$.

   From the induction hypothesis, it is the case that $p'_{final} \in W_i$. Since the actions $\texttt{brake}(i')_{j'}$, $\texttt{unbrake}(i')_{j'}$, for $i' \in I, j' \in J, j' \neq j$, do not affect the variables $\mathit{collided}(i, i'')$, for $i'' \in I, i'' \neq i$, it follows that $p_{final} \in G_i$.

   Moreover, the actions $\texttt{brake}(i')_{j'}$ and $\texttt{unbrake}(i')_{j'}$, for $i' \in I, j' \in J, j' \neq j$, do not affect the velocity of any of the vehicles, *i.e.*, $p_{final}.\dot{x}_{i''} = p'_{final}.\dot{x}_{i''}$, for all $i'' \in I$. From Lemma 4.4.3, part 1, it follows that the section of the track owned by each of the vehicles does not grow, *i.e.*, $p_{final}.O_{i''} \subseteq p'_{final}.O_{i''}$, for all $i'' \in I$. Since $p'_{final} \in W_i$, the sections of track owned in state p'_{final} by the vehicle i does not overlap the sections of track owned by any of the vehicles it trails. From above however, $p_{final}.O_{i''} \subseteq p'_{final}.O_{i''}$, for all $i'' \in I$, and, therefore, the same applies for the state p_{final}.

   Finally, since all states in α are, by definition, restricted to the set R_i, it follows that $p_{final} \in W_i$.

3. the $\texttt{brick-wall}(i')$ actions, for $i' \in I$, set the velocity of the vehicle i' to zero and affect neither the variables $\mathit{collided}(i, i'')$, for $i'' \in I, i'' \neq i$, nor the velocity of any of the other vehicles, *i.e.*, $p_{final}.\dot{x}_{i'''} = p'_{final}.\dot{x}_{i'''}$, for all $i''' \in I, i''' \neq i'$.

   Without loss of generality, consider a particular $\texttt{brick-wall}(i')$ action, for some $i' \in I$.

   From the induction hypothesis, it is the case that $p'_{final} \in W_i$. Since the $\texttt{brick-wall}(i')$

action does not affect the variables $collided(i, i'')$, for $i'' \in I, i'' \neq i$, it follows that $p_{final} \in G_i$.

The `brick-wall`(i') action sets the velocity of the vehicle i' to zero. Therefore, since the vehicle velocities are restricted to be non-negative, it is the case that $p_{final}.\dot{x}_{i'} \leq p'_{final}.\dot{x}_{i'}$. From Lemma 4.4.3, part 1, it follows that $p_{final}.O_{i'} \subseteq p'_{final}.O_{i'}$. Moreover, since the `brick-wall`(i') action does not affect the velocity of any of the other vehicles, it is the case that $p_{final}.\dot{x}_{i''} = p'_{final}.\dot{x}_{i''}$, for all $i'' \in I, i'' \neq i'$. Again, from Lemma 4.4.3, part 1, it follows that the section of the track owned by any of the vehicles other than i' does not grow, *i.e.*, $p_{final}.O_{i''} \subseteq p'_{final}.O_{i''}$, for all $i'' \in I, i'' \neq i'$.

Since $p'_{final} \in W_i$, the sections of track owned in state p'_{final} by the vehicle i does not overlap the sections of track owned by any of the vehicles it trails. From above however, $p_{final}.O_{i''} \subseteq p'_{final}.O_{i''}$, for all $i'' \in I$, and, therefore, the same applies for the state p_{final}.

Finally, since all states in α are, by definition, restricted to the set R_i, it follows that $p_{final} \in W_i$.

4. the actions `colliding-pair`(i', i''), for $i', i'' \in I, i' \neq i''$, and `collision-effects`$(i''')$, for $i''' \in I$, are not enabled because α is only comprised of states in R_i and $p'_{final} \in W_i$.

■

Lemma 6.2.8 $W_i \subseteq safe_i$.

Proof: From the definition of *safe* in Section 3.2.1, we must show that any state $p \in W_i$ satisfies: (i) $future_i(p, 0) \subseteq G_i$, and (ii) there exists some action π such that for every $p', p'' \in R_i$ satisfying $p' \in future_i(p, 0)$ and $p' \xrightarrow{\pi} p''$, it is the case that $p'' \in very\text{-}safe_i$.

(i) The first condition follows from Lemma 6.2.7, Lemma 6.2.1, part 1, and the fact that $p \in W_i$.

(ii) For the second condition, consider the state p'' that follows from p' after a `brake`$(i)_j$ action is executed, *i.e.*, let $\pi =$ `brake`$(i)_j$. Since the `brake`$(i)_j$ action does not affect the velocity of the vehicle i, it is the case that $p''.O_i = p'.O_i$. However, from Lemma 6.2.7 and the fact that $p \in W_i$ it follows that $p' \in W_i$. Since (i) $p' \in W_i$, (ii) the execution fragment α is restricted to the set R_i, and (iii) the `brake`$(i)_j$ action affects neither the variables $collided(i, i')$, for $i' \in I, i' \neq i$, nor the velocity of any of the vehicles (and, therefore, nor the section of the track owned by any of the vehicles), it follows that $p'' \in W_i$. Moreover, since p'' follows from p' after a `brake`$(i)_j$ action, it is the case that $p'' \in P_{B_{ij}}$. From the above conditions, it follows that $p'' \in V_i$. Finally, Lemma 6.2.5 implies that $p'' \in very\text{-}safe_i$, as needed. ■

Lemma 6.2.9 *For any* $p \in R_i$, *if* $p \in safe_i$ *then* $p \in W_i$.

Proof: We show the contrapositive; that is, for any $p \in R_i$, if $p \notin W_i$ then $p \notin \mathit{safe}_i$. Since $W_i = \{p \in R_i \cap G_i \mid \nexists\, i' \in I, i' \neq i : p.O_i \cap p.O_{i'} \neq \emptyset \ \wedge \ p.x_i < p.x_{i'}\}$ and $p \in R_i$, we consider the condition $p \notin G_i$ and the condition that there exists $i' \in I, i' \neq i$, such that $p.O_i \cap p.O_{i'} \neq \emptyset$ and $p.x_i < p.x_{i'}$.

1. $p \notin G_i$.

   From Lemma 3.2.4, part 1, it is the case that $\mathit{safe}_i \subseteq G_i$. Since $p \notin G_i$, it follows that $p \notin \mathit{safe}_i$.

2. $\exists\, i' \in I, i' \neq i$, such that $p.O_i \cap p.O_{i'} \neq \emptyset$ and $p.x_i < p.x_{i'}$.

   Without loss of generality, let $i' \in I, i' \neq i$, be the vehicle that satisfies the conditions $p.O_i \cap p.O_{i'} \neq \emptyset$ and $p.x_i < p.x_{i'}$. Since $p \in \mathit{VALID}$, it is the case that the vehicles in state p have no positive length extent overlap and, therefore, there is only one vehicle i', for $i' \in I, i' \neq i$, satisfying the conditions $p.O_i \cap p.O_{i'} \neq \emptyset$ and $p.x_i < p.x_{i'}$.

   We must show that $p \notin \mathit{safe}_i$. However, $p \in \mathit{safe}_i$ implies that there exists some input action π on port j such that for every $p', p'' \in R_i$ satisfying $p' \in \mathit{future}_i(p, 0)$ and $p' \xrightarrow{\pi} p''$, it is the case that $p'' \in \mathit{very\text{-}safe}_i$. Therefore, it suffices to show that for any input action π on port j, there exist $p', p'' \in R_i$ satisfying $p' \in \mathit{future}_i(p, 0)$ and $p' \xrightarrow{\pi} p''$, such that $p'' \notin \mathit{very\text{-}safe}_i$. We consider each input action π on port j separately.

   (a) $\pi = \texttt{brake}(i)_j$.

   Consider the state $p' \in R_i$ that is reached from the state p through the execution of the action $\texttt{brick-wall}(i')$ and satisfies the condition $p'.\ddot{x}_{i'} = 0$; that is, $p' \in R_i$ such that $p'.\dot{x}_{i'} = 0$ and $p'.\ddot{x}_{i'} = 0$.

   Since the actions $\texttt{brick-wall}(i')$ and $\texttt{brake}(i)_j$ affect neither the position, nor the velocity of the vehicle i, it is the case that $p''.x_i = p'.x_i = p.x_i$ and $p''.\dot{x}_i = p'.\dot{x}_i = p.\dot{x}_i$. Therefore, since the section of track owned by the vehicle i depends only on the position and the velocity of the vehicle i, it is the case that $p''.O_i = p'.O_i = p.O_i$. Similarly, since the $\texttt{brick-wall}(i')$ action does not affect the position of the vehicle i' but sets its velocity to zero and the $\texttt{brake}(i)_j$ action affects neither the position, nor the velocity of the vehicle i', it follows that $p''.x_{i'} = p'.x_{i'} = p.x_{i'}$ and $p''.\dot{x}_{i'} = p'.\dot{x}_{i'} = 0$. Therefore, since $p.O_i \cap p.O_{i'} \neq \emptyset$, $p''.O_i = p.O_i$, and $p''.x_{i'} = p.x_{i'}$, it is the case that $p''.x_{i'} \in p''.O_i$.

   Now, consider the evolution of the VEHICLES automaton following the state p'' in which the vehicle i' remains stationary. Since $p''.x_{i'} \in p''.O_i$, it follows that at some state of such an evolution the action $\texttt{colliding-pair}(i, i')$ is enabled and, subsequently, executed. The state of the VEHICLES automaton following the execution of the action $\texttt{colliding-pair}(i, i')$ would, therefore, not be in G_i. It follows that $p'' \notin \mathit{very\text{-}safe}_i$ which implies that $p \notin \mathit{safe}_i$.

(b) $\pi = \texttt{unbrake}(i)_j$.

Consider the state $p' \in R_i$ that is reached from the state p through the execution of the actions $\texttt{brick-wall}(i')$ and $\texttt{unbrake}(i)_{j'}$, for all $j' \in J, j' \neq j$, and satisfies the condition $p'.\ddot{x}_{i'} = 0$; that is, $p' \in R_i$ such that $p'.\dot{x}_{i'} = 0$, $p'.\ddot{x}_{i'} = 0$, and $p'.\mathit{brake\text{-}req}(i, j') = \texttt{False}$, for all $j' \in J, j' \neq j$.

Since the actions $\texttt{brick-wall}(i')$ and $\texttt{unbrake}(i)_{j'}$, for all $j' \in J$, affect neither the position, nor the velocity of the vehicle i, it follows that $p''.x_i = p'.x_i = p.x_i$ and $p''.\dot{x}_i = p'.\dot{x}_i = p.\dot{x}_i$. Therefore, since the section of track owned by the vehicle i depends only on the position and the velocity of the vehicle i, it is the case that $p''.O_i = p'.O_i = p.O_i$. Similarly, since the action $\texttt{brick-wall}(i')$ does not affect the position of the vehicle i' but sets its velocity to zero and the actions $\texttt{unbrake}(i)_{j'}$, for all $j' \in J$, affect neither the position, nor the velocity of the vehicle i', it follows that $p''.x_{i'} = p'.x_{i'} = p.x_{i'}$ and $p''.\dot{x}_{i'} = p'.\dot{x}_{i'} = 0$. Therefore, since $p.O_i \cap p.O_{i'} \neq \emptyset$, $p''.O_i = p.O_i$, and $p''.x_{i'} = p.x_{i'}$, it is the case that $p''.x_{i'} \in p''.O_i$.

Now, consider the evolution of the VEHICLES automaton following the state p'' in which the vehicle i moves forward and the vehicle i' remains stationary. Since $p''.x_{i'} \in p''.O_i$, it follows that at some state of such an evolution the action $\texttt{colliding-pair}(i, i')$ is enabled and, subsequently, executed. The state of the VEHICLES automaton following the execution of the action $\texttt{colliding-pair}(i, i')$ would, therefore, not be in G_i. It follows that $p'' \notin \mathit{very\text{-}safe}_i$ which implies that $p \notin \mathit{safe}_i$.

Thus, for any input action π on port j, there exist $p', p'' \in R_i$ satisfying $p' \in \mathit{future}_i(p, 0)$ and $p' \xrightarrow{\pi} p''$, such that $p'' \notin \mathit{very\text{-}safe}_i$. It follows that $p \notin \mathit{safe}_i$, as needed.

■

Corollary 6.2.10 $W_i = \mathit{safe}_i$.

Proof: Follows directly from Lemmas 6.2.8 and 6.2.9. ■

In the next few lemmas, we show that any state p in the set $T_i(t)$, for any $t \in \mathbb{R}^{\geq 0}$, is in the set $\mathit{delay\text{-}safe}_i(t)$; that is, any state R_i-reachable from p within an amount of time t through an execution fragment that involves no input actions on port j, is in the set G_i and any state R_i-reachable from the state p in exactly an amount of time t through an execution fragment that involves no input actions on port j, is in the set safe_i.

Lemma 6.2.11 *Let $p \in T_i(\tau)$, where $\tau \in \mathbb{R}^{\geq 0}$, and $p' \in \mathit{future}_i(p, t)$, where $t \in [0, \tau]$. If $p' \in G_i$ and $p'.C_i(\tau - t) \subseteq p.C_i(\tau)$, then $p' \in T_i(\tau - t)$.*

Proof: We need to show that $p' \in R_i \cap G_i$ and that there does not exist $i' \in I, i' \neq i$ such that $p'.C_i(\tau - t) \cap p'.C_{i'}(\tau - t) \neq \emptyset$ and $p'.x_i < p'.x_{i'}$. We consider the conditions by cases:

1. $p' \in R_i$.

   This is the case because the function $future_i(p, t)$ only considers R_i-reachable states.

2. $p' \in G_i$.

   This is true by assumption.

3. $\nexists\, i' \in I, i' \neq i,$ such that $p'.C_i(\tau - t) \cap p'.C_{i'}(\tau - t) \neq \emptyset$ and $p'.x_i < p'.x_{i'}$.

   Because $p \in T_i(\tau)$ we have that for all $i' \in I, i' \neq i$, such that $p.x_i < p.x_{i'}$, it is the case that $p.C_i(\tau) \cap p.C_{i'}(\tau) = \emptyset$; that is, for all $i' \in I, i' \neq i$, such that $p.x_i < p.x_{i'}$, it is the case that $\max(p.C_i(\tau)) < \min(p.C_{i'}(\tau))$. However, by assumption it is the case that $p'.C_i(\tau - t) \subseteq p.C_i(\tau)$. Therefore, since the vehicle velocities are restricted to be non-negative, it follows that for all $i' \in I, i' \neq i$, such that $p'.x_i < p'.x_{i'}$, it is the case that $\max(p'.C_i(\tau - t)) < \min(p'.C_{i'}(\tau - t))$. This is sufficient to guarantee that there does not exist $i' \in I, i' \neq i$, such that $p'.C_i(\tau - t) \cap p'.C_{i'}(\tau - t) \neq \emptyset$ and $p'.x_i < p'.x_{i'}$.

■

Lemma 6.2.12 *For all $p \in T_i(\tau)$, where $\tau \in \mathbb{R}^{\geq 0}$, and $p' \in future_i(p, t)$, where $t \in [0, \tau]$, it is the case that $p' \in G_i$ and $p'.C_i(\tau - t) \subseteq p.C_i(\tau)$.*

Proof: Let $\tau \in \mathbb{R}^{\geq 0}$ and α be an execution fragment of the VEHICLES automaton of n steps and trajectories, where $n \in \mathbb{N}$, that: starts in a state in $T_i(\tau)$, is only comprised of states in R_i, involves no input actions on port j, and has a limit time t that lies in the interval $[0, \tau]$. Letting p_{init} and p_{final} be the initial and final states of α, respectively, we must show that $p_{final} \in G_i$ and $p_{final}.C_i(\tau - t) \subseteq p_{init}.C_i(\tau)$. The proof is by induction on the length n of the execution fragment α.

For the base case, consider the execution fragment α of length $n = 0$; that is, α is an execution fragment that consists of a single point trajectory and, therefore, $p_{final} = p_{init}$ and $\alpha.ltime = 0$, *i.e.*, $t = 0$. From Lemma 6.2.1, part 1, and the fact that $p_{init} \in T_i(\tau)$, it follows that $p_{final} \subseteq G_i$. Moreover, since $t = 0$, the fact that $p_{final}.C_i(\tau - t) \subseteq p_{init}.C_i(\tau)$ is trivially true.

The inductive step involves showing that if α is an execution fragment of length $n = k + 1$, for some $k \in \mathbb{N}$, with $\alpha.ltime = t$, where $t \in [0, \tau]$, then $p_{final} \in G_i$ and $p_{final}.C_i(\tau - t) \subseteq p_{init}.C_i(\tau)$. Let α' be the part of the execution fragment α comprised of the first k steps and trajectories and let $\alpha'.ltime = t'$, where $t' \in [0, t]$. The induction hypothesis involves the assertion that if p'_{init} and p'_{final} are the initial and final states of α', respectively, then it is the case that $p'_{final} \in G_i$ and $p'_{final}.C_i(\tau - t') \subseteq p'_{init}.C_i(\tau)$. Moreover, from Lemma 6.2.11

it follows that $p'_{final} \in T_i(\tau - t')$. Since the final state of α is reached from the final state of α' by a single step or trajectory, the inductive step involves the consideration of all possible steps and trajectories leading from p'_{final} to p_{final}.

In the case of a step, keeping in mind that the limit times of α' and α are equal, *i.e.*, $t' = t$, we consider all possible actions by cases:

1. the actions $\texttt{brake}(i)_j$ and $\texttt{unbrake}(i)_j$ are not enabled because α involves no input actions on port j.

2. the $\texttt{brick-wall}(i)$ action sets the velocity of the vehicle i to zero and does not affect the variables $\mathit{collided}(i, i')$, for $i' \in I, i' \neq i$.

   From the induction hypothesis, it is the case that $p'_{final} \in G_i$. Therefore, since the internal action $\texttt{brick-wall}(i)$ does not affect the variables $\mathit{collided}(i, i')$, for $i' \in I, i' \neq i$, it follows that $p_{final} \in G_i$.

   Moreover, since the vehicle velocities are restricted to be non-negative, it is the case that $p_{final}.\dot{x}_i \leq p'_{final}.\dot{x}_i$. From Lemma 4.4.3, part 2, it follows that $p_{final}.C_i(\tau - t) \subseteq p'_{final}.C_i(\tau - t')$. However, from the induction hypothesis it is the case that $p'_{final}.C_i(\tau - t') \subseteq p'_{init}.C_i(\tau)$. Therefore, since $p_{init} = p'_{init}$, it follows that $p_{final}.C_i(\tau - t) \subseteq p_{init}.C_i(\tau)$, as needed.

3. the actions $\texttt{brake}(i')_{j'}$, $\texttt{unbrake}(i')_{j'}$, for $i' \in I, j' \in J, j' \neq j$, and $\texttt{brick-wall}(i'')$, for $i'' \in I, i'' \neq i$, affect neither the velocity of the vehicle i, nor the variables $\mathit{collided}(i, i''')$, for $i''' \in I, i''' \neq i$.

   From the induction hypothesis, it is the case that $p'_{final} \in G_i$. Therefore, since the actions $\texttt{brake}(i')_{j'}$, $\texttt{unbrake}(i')_{j'}$, for $i' \in I, j' \in J, j' \neq j$, and $\texttt{brick-wall}(i'')$, for $i'' \in I, i'' \neq i$, do not affect the variables $\mathit{collided}(i, i''')$, for $i''' \in I, i''' \neq i$, it follows that $p_{final} \in G_i$.

   Moreover, since the input actions $\texttt{brake}(i')_{j'}$, $\texttt{unbrake}(i')_{j'}$, for $i' \in I, j' \in J, j' \neq j$, and the internal actions $\texttt{brick-wall}(i'')$, for $i'' \in I, i'' \neq i$, do not affect the velocity of the vehicle i, it is the case that $p_{final}.\dot{x}_i = p'_{final}.\dot{x}_i$. From Lemma 4.4.3, part 2, it follows that $p_{final}.C_i(\tau - t) \subseteq p'_{final}.C_i(\tau - t')$. However, from the induction hypothesis it is the case that $p'_{final}.C_i(\tau - t') \subseteq p'_{init}.C_i(\tau)$. Therefore, since $p_{init} = p'_{init}$, it follows that $p_{final}.C_i(\tau - t) \subseteq p_{init}.C_i(\tau)$, as needed.

4. the actions $\texttt{colliding-pair}(i', i'')$, for $i', i'' \in I, i' \neq i''$, and $\texttt{collision-effects}(i''')$, for $i''' \in I$, are not enabled because α is only comprised of states in R_i and $p'_{final} \in T_i(\tau - t')$.

In the case of a trajectory, Lemma 4.4.4, part 2, implies that $p_{final}.C_i(\tau - t) \subseteq p'_{final}.C_i(\tau - t')$. However, from the induction hypothesis it is the case that $p'_{final}.C_i(\tau - t') \subseteq p'_{init}.C_i(\tau)$.

Therefore, since $p_{init} = p'_{init}$, it follows that $p_{final}.C_i(\tau - t) \subseteq p_{init}.C_i(\tau)$. Moreover, since $p'_{final} \in G_i$ and the variables $collided(i, i')$, for all $i' \in I, i' \neq i$, remain constant throughout the trajectory, it follows that $p_{final} \in G_i$, as needed. ■

Lemma 6.2.13 *For $\tau \in \mathbb{R}^{\geq 0}$ and $t \in [0, \tau]$, it is the case that $future_i(T_i(\tau), t) \subseteq T_i(\tau - t)$.*

Proof: Follows directly from Lemmas 6.2.11 and 6.2.12. ■

Lemma 6.2.14 *For all $t \in \mathbb{R}^{\geq 0}$, it is the case that $T_i(t) \subseteq delay\text{-}safe_i(t)$.*

Proof: We need to show that $future_i(T_i(t), [0, t]) \subseteq G_i$ and $future_i(T_i(t), t) \subseteq safe_i$. The first condition follows directly from Lemma 6.2.13 and Lemma 6.2.1, part 1. For the second condition, from Lemma 6.2.13 and Lemma 6.2.1, part 3, it is the case that $future_i(T_i(t), t) \subseteq W_i$. Therefore, Lemma 6.2.8, implies that $future_i(T_i(t), t) \subseteq safe_i$, as needed. ■

In the following lemma, we show that the CL-PROT-SOLO$_i$ protector implements the protector Abs(VEHICLES, S_i, R_i, G_i, j, d). Since the protector automata CL-PROT-SOLO$_i$ and Abs_j involve the composition of the same sensor automaton with distinct discrete controller automata, it suffices to show that the discrete controller automaton of the protector CL-PROT-SOLO$_i$ implements the DC(VEHICLES, S_i, R_i, G_i, j, d) automaton.

Lemma 6.2.15 CL-PROT-SOLO$_i \leq Abs$(VEHICLES, S_i, R_i, G_i, j, d).

Proof: Both the CL-PROT-SOLO$_i$ and the Abs_j protectors involve the composition of the same sensor automaton with distinct discrete controller automata. From Theorem 2.7.4, it suffices to show that the discrete controller automaton of CL-PROT-SOLO$_i$ implements DC_j. This is shown by a simulation from the discrete controller automaton of CL-PROT-SOLO$_i$ to DC_j.

As in the overspeed case, the mapping between the states of the discrete controller automaton of CL-PROT-SOLO$_i$ and DC_j is almost the identity. In the discrete controller automaton of CL-PROT-SOLO$_i$, the variable $send_j$ is equal to either one of the labels `brake` and `unbrake`, or the value *null*. In the abstract discrete controller automaton, these valuations simply map to either the actions `brake`$(i)_j$ and `unbrake`$(i)_j$, or the value *null*, respectively.

The start states for the discrete controller automaton of CL-PROT-SOLO$_i$ and DC_j are the states in which $send_j = null$. These are mapped to each other according to the mapping discussed above.

Furthermore, since the trajectories in both discrete controller automata are identical, we need only consider their discrete transitions. We analyze the actions of the implementation by cases, letting p denote any complete state of the VEHICLES automaton that corresponds to y, *i.e.*, $p \in VALID$ and $p\lceil Y_{\text{VEHICLES}} = y$.

1. The $\texttt{snapshot}(y)_j$ action of the implementation sets $send_j$ to `brake`, or `unbrake`. In order to show that the behavior of the implementation is allowed by the specification, we must show that the input action $\texttt{snapshot}(y)_j$ of the implementation sets the value of the $send_j$ variable in such a way that the subsequently enabled action π of the implementation (i) guarantees that for all $p', p'' \in R_i$ such that $p' \in future_i(p, 0)$ and $p' \xrightarrow{\pi} p''$, it is the case that $p'' \in delay\text{-}safe_i(d)$, if $p \in safe_i$, and (ii) is an arbitrary output action of the implementation, otherwise.

   First, consider the case in which $p \in safe_i$. Since Corollary 6.2.10 implies that $p \in W_i$, the discrete controller automaton of CL-PROT-SOLO$_i$ sets the variable $send_j$ according to whether the state p is in $T_i(d)$, or not.

   On one hand, if $p \notin T_i(d)$ then the discrete controller automaton of CL-PROT-SOLO$_i$ sets the variable $send_j$ to `brake` and the $\texttt{brake}(i)_j$ action is enabled. However, since $p \in W_i$, Lemma 6.2.7 implies that $p' \in W_i$. Moreover, since the $\texttt{brake}(i)_j$ action affects neither the velocity of any of the vehicles, nor any of the *collided* variables, it follows that $p'' \in R_i$, $p'' \in G_i$, and $p''.\dot{x}_i = p'.\dot{x}_i$. Therefore, Lemma 4.4.3, part 1, implies that $p''.O_i \subseteq p'.O_i$. From the above conditions and the non-negative constraint on the vehicle velocities, it follows that $p'' \in W_i$. Moreover, since the $\texttt{brake}(i)_j$ action sets the $brake\text{-}req(i, j)$ variable to `True`, it follows that $p'' \in V_i$. Finally, from Lemma 6.2.6 it follows that $p'' \in delay\text{-}safe_i(d)$, as needed.

   On the other hand, if $p \in T_i(d)$ then the discrete controller automaton of the protector CL-PROT-SOLO$_i$ sets the variable $send_j$ to `unbrake` and the $\texttt{unbrake}(i)_j$ action is enabled. However, since $p \in T_i(d)$, Lemma 6.2.13 implies that $p' \in T_i(d)$. Since the $\texttt{unbrake}(i)_j$ action affects neither the velocity of any of the vehicles, nor any of the *collided* variables, it follows that $p'' \in R_i$, $p'' \in G_i$, and $p''.\dot{x}_i = p'.\dot{x}_i$. Therefore, Lemma 4.4.3, part 2, implies that $p''.C_i(d) \subseteq p'.C_i(d)$. From the above conditions and the non-negative constraint on the vehicle velocities, it follows that $p'' \in T_i(d)$. Finally, from Lemma 6.2.14 it follows that $p'' \in delay\text{-}safe_i(d)$, as needed.

   Next, consider the case in which $p \notin safe_i$. In this case, the $\texttt{snapshot}(y)_j$ action of the discrete controller automaton of CL-PROT-SOLO$_i$ sets the variable $send_j$ to either `brake` or `unbrake` and, subsequently, enables either the action $\texttt{brake}(i)_j$, or the action $\texttt{unbrake}(i)_j$. However, when $p \notin safe_i$, the DC_j automaton sets the variable $send_j$ arbitrarily and, subsequently, enables an arbitrary output action. Therefore, the behavior of the discrete controller automaton of CL-PROT-SOLO$_i$ is allowed by that of the DC_j automaton.

   Therefore, the effects of the $\texttt{snapshot}(y)_j$ action of the implementation are allowed by its specification.

2. The $\texttt{brake}(i)_j$ and $\texttt{unbrake}(i)_j$ actions have identical effects in both discrete controller automata. When the $send_j$ variable matches the label `brake` or `unbrake` or the action

`brake`$(i)_j$ and `unbrake`$(i)_j$, respectively, the respective action is performed and the $send_j$ variable is set to *null* in both discrete controller automata.

3. The environment action in both discrete controller automata is stuttering. It follows that the mapping between the states of the discrete controller automaton of CL-PROT-SOLO$_i$ and the DC_j automaton prior to and succeeding the execution of the environment action remains the same.

■

Corollary 6.2.16 *The protector* CL-PROT-SOLO$_i$ *guarantees* G_i *in the* VEHICLES *automaton starting from* S_i *given* R_i.

Proof: Follows directly from Lemma 6.2.15 and Theorem 3.2.9. ■

6.3 Protection System CL-PROT

We now define the collision protector CL-PROT. While considering the CL-PROT automaton, we restrict the states of the VEHICLES automaton to the set $P_{not\text{-}overspeed}$ as defined in Section 4.2, *i.e.*, $R_{\text{CL-PROT}} = P_{not\text{-}overspeed}$. Let $G_{\text{CL-PROT}}$ and $S_{\text{CL-PROT}}$ be the intersection of G_i and S_i, for all $i \in I$, respectively, and CL-PROT be the composition of the protectors CL-PROT-SOLO$_i$, for all $i \in I$. The protector CL-PROT guarantees that the VEHICLES automaton remains within $G_{\text{CL-PROT}}$ starting from $S_{\text{CL-PROT}}$ given $R_{\text{CL-PROT}}$. For reference, the formal definitions of the CL-PROT automaton and the sets $G_{\text{CL-PROT}}$, $S_{\text{CL-PROT}}$, and $R_{\text{CL-PROT}}$ are shown in Table 6.2.

Lemma 6.3.1 *The protector* CL-PROT *guarantees* $G_{\text{CL-PROT}}$ *in the* VEHICLES *automaton starting from* $S_{\text{CL-PROT}}$ *given* $R_{\text{CL-PROT}}$.

In the following proof, we show that all the states of an execution of $PP \times$ CL-PROT starting from $S_{\text{CL-PROT}}$ given $R_{\text{CL-PROT}}$ are in $G_{\text{CL-PROT}}$. This is done by applying Theorem 3.1.8 and showing that the second condition of the theorem does not hold.

Proof: Let α be any execution of the system $PP \times$ CL-PROT starting from a state in $S_{\text{CL-PROT}}$ and in which all states are in $R_{\text{CL-PROT}}$.

From Theorem 3.1.8, one of the following holds:

1. Every state in α is in $G_{\text{CL-PROT}} = \bigcap_{i \in I} G_i$.
2. α can be written as $\alpha_1 \frown \alpha_2$, where

Table 6.2 Formal definitions of CL-PROT, $G_{\text{CL-PROT}}$, $S_{\text{CL-PROT}}$, and $R_{\text{CL-PROT}}$.

$$\text{CL-PROT} \equiv \prod_{i \in I} \text{CL-PROT-SOLO}_i$$

$$G_{\text{CL-PROT}} \equiv \bigcap_{i \in I} G_i$$

$$S_{\text{CL-PROT}} \equiv \bigcap_{i \in I} S_i$$

$$R_{\text{CL-PROT}} \equiv P_{\textit{not-overspeed}}$$

(a) All state occurrences in α_1 except possibly the last state occurrence are in the set $G_{\text{CL-PROT}} = \bigcap_{i \in I} G_i$.

(b) If the last state occurrence in α_1 is in $\overline{G_i}$, for some $i \in I$, then there exists $i' \in I, i' \neq i$, such that the last state occurrence in α_1 is in $\overline{G_{i'}}$.

(c) All state occurrences in α_2 except possibly the first state occurrence are in the set $\bigcap_{i \in N} \textit{past}(\overline{G_i}, \alpha)$, for some $N \subseteq I$, where $|N| \geq 2$.

We proceed by showing that it is not possible to decompose α as $\alpha_1 \frown \alpha_2$ while satisfying the three aforementioned conditions.

The violation of $\bigcap_{i \in I} G_i$ can only occur through the violation of at least one of the conditions G_i, where $i \in I$. Moreover, each of these conditions are violated only through the execution of a `colliding-pair` action. Without loss of generality, suppose that the first condition that is violated in α is the condition G_i, for some $i \in I$, and that such a violation has resulted through a `colliding-pair`(i, i') action, for some $i' \in I, i' \neq i$. Let p and p' be the states of the VEHICLES automaton prior to and succeeding this `colliding-pair`(i, i') action, *i.e.*, $p, p' \in R_{\text{CL-PROT}}$ such that $p \xrightarrow{\pi} p'$, where $\pi =$ `colliding-pair`(i, i'). Since the `colliding-pair`(i, i') action only sets the *collided*(i, i') variable to `True`, it follows that $p' \in \overline{G_i} \bigcap \left(\bigcap_{i'' \in I, i'' \neq i} G_{i''} \right)$. Now, we attempt to decompose α as $\alpha_1 \frown \alpha_2$:

1. Suppose we split α at any state preceding the state p. Then the state p is in α_2. Since p' is the first state in which one of the conditions $G_{i''}$, for $i'' \in I$, is violated, it is the case that $p \in \bigcap_{i'' \in I} G_{i''}$ and there does not exist $N \subseteq I$ such that $|N| \geq 2$ and $p \in \bigcap_{i \in N} \textit{past}(\overline{G_i}, \alpha)$. Therefore, the third condition is violated and this decomposition of α is not valid.

2. Suppose we split α at the state p. Then the state p' is in α_2. Since p' is the first state in which one of the conditions $G_{i''}$, for $i'' \in I$, is violated and since the state p' is in $\overline{G_i} \bigcap \left(\bigcap_{i'' \in I, i'' \neq i} G_{i''} \right)$, it follows that there does not exist $N \subseteq I$ such that $|N| \geq 2$ and $p' \in \bigcap_{i \in N} past(\overline{G_i}, \alpha)$. Therefore, the third condition is violated and this decomposition of α is not valid.

3. Suppose we split α at the state p'. Then p' is the last state of α_1 and the first state of α_2. However, $p' \in \overline{G_i} \bigcap \left(\bigcap_{i'' \in I, i'' \neq i} G_{i''} \right)$. Therefore, the second condition is violated and this decomposition of α is not valid.

4. Suppose we split α at any state succeeding p'. Then the state p' is in α_1. Since $p' \in \overline{G_i} \bigcap \left(\bigcap_{i'' \in I, i'' \neq i} G_{i''} \right)$, it is the case that $p' \notin \bigcap_{i'' \in I} G_{i''}$. Therefore, the first condition is violated and this decomposition of α is not valid.

Therefore, the execution α cannot be decomposed into any such α_1 and α_2. It follows that the first clause of Theorem 3.1.8 must hold; that is, every state in α is in $G_{\text{CL-PROT}}$. This implies that the protector CL-PROT guarantees $G_{\text{CL-PROT}}$ in the VEHICLES automaton starting from $S_{\text{CL-PROT}}$ given $R_{\text{CL-PROT}}$. ■