Title:
Conditions for Safe Deceleration of Strings of Vehicles

Author:
Lygeros, John
Lynch, Nancy

Publication Date:
01-01-2000

Series:
Research Reports

Publication Info:
Research Reports, California Partners for Advanced Transit and Highways (PATH), Institute of Transportation Studies (UCB), UC Berkeley

Permalink:
http://escholarship.org/uc/item/306410hv

Keywords:
Acceleraton (Mechanics)--Mathematical models, Automobiles--Automatic control--Mathematical models, Automobile driving--Braking--Automation, Express highways--Safety measures

Abstract:
A simple model for a string of vehicles is constructed. The model explicitly accounts for the possibility of repeated collisions between the vehicles in the string. Based on the model a notion of safety is formulated for the string. Necessary and sufficient conditions are presented that specify when a string of vehicles is safe while performing a simple emergency deceleration maneuver where all vehicles start decelerating at a fixed rate after some delay. The conditions are interpreted in terms of their implications for the safety of platoons of vehicles.

This paper uses Postscript Type 3 fonts. Although reading it on the screen is difficult it will print out just fine.

CALIFORNIA PATH PROGRAM
INSTITUTE OF TRANSPORTATION STUDIES
UNIVERSITY OF CALIFORNIA, BERKELEY

Conditions for Safe Deceleration of Strings of Vehicles

John Lygeros
Nancy Lynch

California PATH Research Report
UCB-ITS-PRR-2000-2

This work was performed as part of the California PATH Program of the University of California, in cooperation with the State of California Business, Transportation, and Housing Agency, Department of Transportation; and the United States Department of Transportation, Federal Highway Administration.

Report for MOU 319

January 2000

ISSN 1055-1425

CALIFORNIA PARTNERS FOR ADVANCED TRANSIT AND HIGHWAYS

Conditions for Safe Deceleration of Strings of Vehicles*

John Lygeros† and Nancy Lynch‡

†Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, CA 94720-1770
lygeros@eecs.berkeley.edu

‡Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139
lynch@lcs.mit.edu

Abstract

A simple model for a string of vehicles is constructed. The model explicitly accounts for the possibility of repeated collisions between the vehicles in the string. Based on the model a notion of safety is formulated for the string. Necessary and sufficient conditions are presented that specify when a string of vehicles is safe while performing a simple emergency deceleration maneuver where all vehicles start decelerating at a fixed rate after some delay. The conditions are interpreted in terms of their implications for the safety of platoons of vehicles.

1 Introduction

Hybrid systems have attracted the attention of both computer theorists and control engineers. Our work ultimately aims at a rapprochement of these two perspectives. Here we use a combination of techniques from the two areas to address a specific problem in transportation. This is the problem of the safety of a collection of vehicles traveling one behind the other in a single lane; we refer to such a collection as a *string of vehicles*. The problem is hybrid as it involves both continuous vehicle motion and (possibly) collisions, which in our setting are treated as discrete velocity changes. We try to establish conditions under which a string of vehicles will be safe while executing a particular maneuver.

We start by developing a detailed model for the system in the *Hybrid Input/Output Automaton* modeling framework (Section 3). Modest extensions of the original framework of [1] are needed to capture all the phenomena of interest for this problem. Then, in Section 4 we introduce the emergency deceleration maneuver, whose safety analysis is the primary focus of this paper. We give some necessary and some sufficient conditions under which the safety of the maneuver can be guaranteed. Finally, in Section 7, we discuss the implications of our results in the context of platooning of vehicles.

*Research supported by the PATH program, Institute of Transportation Studies, University of California, Berkeley, under MOU-238, MOU-310, MOU-312 and MOU-319.

We believe our work is potentially of both theoretical and practical importance. On the theoretical side we hope that the results presented here will be extended to a general methodology for dealing with hybrid systems, one where continuous and discrete techniques are combined in a coherent framework. The practical implications of our work are more immediate. Our results indicate that the design of specialized emergency maneuvers may be crucial to the success of an automated highway system that allows for the formation of platoons.

2 Modeling Formalism and Definitions

The vehicles will be modeled in the Hybrid Input-Output Automaton (HIOA) framework of [1]. In this section we give a brief overview of this modeling formalism. We also specify some special classes of automata that will be used in subsequent sections.

2.1 Notation

Let $dom(f)$ and $range(f)$ denote respectively the domain and range of the function f. Functions are denoted by $f : dom(f) \to range(f)$. If f is a function and X a set, then we write $f \lceil X$ for the *restriction* of f to X, i.e. the function g with $dom(g) = dom(f) \cap X$ satisfying $g(x) = f(x)$, for all $x \in dom(g)$. We say that two functions f and g are compatible if $f \lceil dom(g) = g \lceil dom(f)$. If f and g are compatible functions, then we write $f \cup g$ for the function h with $dom(h) = dom(f) \cup dom(g)$ such that $h(x) = f(x)$, if $x \in dom(f)$, and $h(x) = g(x)$, otherwise, for all $x \in dom(h)$. If f is a function whose range consists of a set of functions and X is a set, then we write $f \downarrow X$ for the restriction of the functions in $range(f)$ to the set X, i.e. the function g with $dom(g) = dom(f)$ defined by $g(x) \triangleq f(x) \lceil X$, for all $x \in dom(g)$.

We fix the *time axis*, T, to be the set of real numbers, $\mathbb{R}$[1]. Let $T^{\geq 0} = \{t \in T \mid t \geq 0\}$. For $T' \subseteq T$ and $t \in T$, we define $T' + t \triangleq \{t' + t \mid t' \in T'\}$. For a function f with domain T', we define $f + t$ to be the function with domain $T' + t$ satisfying $(f + t)(t') = f(t' - t)$, for all $t' \in T' + t$. An *interval*, T_I, is a non-empty convex subset of T. As usual, intervals are denoted by $[t_1, t_2] = \{t \in T \mid t_1 \leq t \leq t_2\}$, $[t_1, t_2) = \{t \in T \mid t_1 \leq t < t_2\}$ etc. An interval is right-open (left-open), if it does not have a maximal (minimal) element, and right-closed (left-closed), otherwise. We write $\max(T_I)$ and $\min(T_I)$ for the maximal and the minimal elements, respectively, of the interval T_I (if they exist), and $\sup(T_I)$ and $\inf(T_I)$ for the supremum and infimum, respectively, of the interval T_I in $T \cup \{-\infty, \infty\}$.

We assume a universal set $\mathcal{V}$ of typed *variables*. The type of a variable, denoted by $type(v)$, indicates the set over which the variable takes values. Let $Z \subseteq \mathcal{V}$. A *valuation* of Z is a function that associates to each variable v of Z a value in $type(v)$. We write $\mathbf{Z}$ for the set of valuations of Z. Often, valuations will be referred to as *states*.

A *trajectory* over a set of variables Z is a function $w : T_I \to \mathbf{Z}$, where T_I is a left-closed interval of T with $\min(T_I) = 0$. Let $traj(Z)$ denote the collection of all trajectories over Z. For $w \in traj(Z)$, we define the *limit time* of w by $ltime(w) \triangleq \sup(dom(w))$. A trajectory w is finite if $ltime(w) \neq \infty$. We define the *first state* of a trajectory w, by $fstate(w) \triangleq w(0)$. If the domain of a trajectory w is right-closed, then we define the *last state* of w by $lstate(w) \triangleq w(ltime(w))$. If T_I is a left-closed interval with $\min(T_I) \in dom(w)$, then we define the *curtailment* of w to T_I by $w \dagger T_I \triangleq (w \lceil T_I) - \min(T_I)$. A trajectory with domain $[0, 0]$ is called a *point* trajectory. If s is a state, then we define $\wp(s)$ to be the point trajectory that maps 0 to s. If w is a finite trajectory with domain T_I, w' is a trajectory with domain T_I', and T_I right-closed implies $lstate(w) = fstate(w')$,

[1]For the HIOA definitions, T can in fact be any subgroup of $(\mathbb{R}, +)$.

we define the *concatenation* of w and w' to be the trajectory $w \frown w' \triangleq w \cup (w' + ltime(w))$. The concatenation operator can be extended to an infinite sequence of finite trajectories $w_0 w_1 w_2 \cdots$.

2.2 Hybrid I/O Automata and Composition

A *hybrid I/O automaton* (HIOA), $A = (U, X, Y, \Sigma^{in}, \Sigma^{int}, \Sigma^{out}, \Theta, \mathcal{D}, \mathcal{W})$, is a collection of:

- Three disjoint sets U, X, and Y of variables, called *input*, *internal*, and *output variables*, respectively. We write $V \triangleq U \cup X \cup Y$ and let s, u, and w denote elements of $\mathbf{V}$, $\mathbf{U}$, and $traj(V)$, respectively.
- Three disjoint sets Σ^{in}, Σ^{int}, and Σ^{out} of actions, called *input*, *internal*, and *output actions*, respectively. We assume that Σ^{in} contains a special element e, the environment action, which represents the occurrence of a discrete transition outside the system that is unobservable, except (possibly) through its effect on the input variables. We write $\Sigma \triangleq \Sigma^{in} \cup \Sigma^{int} \cup \Sigma^{out}$ and let a range over Σ.
- A non-empty set $\Theta \subseteq \mathbf{V}$ of *initial states* satisfying:

  Init (initial states closed under change of input variables)
  $s \in \Theta \Rightarrow \exists s' \in \Theta \ : \ s' \lceil U = u \ \wedge \ s' \lceil Y = s \lceil Y$

- A set $\mathcal{D} \subseteq \mathbf{V} \times \Sigma \times \mathbf{V}$ of *discrete transitions* satisfying:

  D1 (input action enabling)
  $a \in \Sigma^{in} \Rightarrow \exists s' \in \mathbf{V} \ : \ s \xrightarrow{a} s'$

  D2 (environment actions that do not change inputs do not affect the state)
  $s \xrightarrow{e} s' \ \wedge \ s \lceil U = s' \lceil U \Rightarrow s = s'$

  D3 (discrete transitions do not depend on input variable changes)
  $s \xrightarrow{a} s' \Rightarrow \exists s'' \in \mathbf{V} \ : \ s \xrightarrow{a} s'' \ \wedge \ s'' \lceil U = u \ \wedge \ s'' \lceil Y = s' \lceil Y$

  $s \xrightarrow{a} s'$ is a shorthand for $(s, a, s') \in \mathcal{D}$.

- A set $\mathcal{W}$ of *trajectories* over V satisfying:

  T1 (existence of point trajectories)
  $\wp(s) \in \mathcal{W}$

  T2 (closure under subintervals)
  $w \in \mathcal{W} \ \wedge \ (T_I \text{ left-closed subinterval of } dom(w)) \Rightarrow w \dagger T_I \in \mathcal{W}$

  T3 (completeness)
  $(\forall t \in T^{\geq 0} : w \dagger [0, t] \in \mathcal{W}) \Rightarrow w \in \mathcal{W}$

The intuition behind Axioms **Init** and **D1-3** is that a HIOA is responsible for performing locally controlled actions and for modifying the values of its local variables, whereas the environment of a HIOA is responsible for performing input actions and modifying the values of the input variables. Axiom **Init** says that a system may not constrain the initial values of its input variables. Axiom **D1** says that a HIOA should accept all input actions in all states. Axiom **D2** postulates that an environment action that does not affect the input variables can *not* be "detected" by the automaton and, therefore, leaves the state unchanged. Axiom **D3** states that there is no functional dependence between the input and the output variables of a HIOA during a transition; that is, a HIOA can *not*

react instantaneously to an input variable change. This is done to avoid cyclic constraints during the interaction of two systems. Under these conditions one can show that the composition of two HIOA is still input enabled and that the environment can never block the output actions of a system.

Axioms **T1-3** state some natural conditions on the set of transitions: existence of point trajectories, closure under subintervals, and the fact that a full trajectory is in $\mathcal{W}$ if and only if all its prefixes are in $\mathcal{W}$.

Given a collection of hybrid automata the above definitions and axioms allow one to form new automata by appropriate operations. To ensure that the resulting automaton again satisfies the axioms we need to impose a compatibility requirement. Two HIOA, $A_i = (U_i, X_i, Y_i, \Sigma_i^{in}, \Sigma_i^{int}, \Sigma_i^{out}, \Theta_i, \mathcal{D}_i, \mathcal{W}_i)$, $i \in \{1,2\}$, are *compatible* if, for $i, j \in \{1,2\}, i \neq j$,

$$X_i \cap V_j = Y_i \cap Y_j = \Sigma_i^{int} \cap \Sigma_j = \Sigma_i^{out} \cap \Sigma_j^{out} = \emptyset.$$

Let $s \xrightarrow{a}_{A_i} s'$ be a shorthand for $(s, a, s') \in \mathcal{D}_i$. The *composition*, $A_1 \times A_2$, of two compatible HIOA A_1 and A_2 is the tuple $A = (U, X, Y, \Sigma^{in}, \Sigma^{int}, \Sigma^{out}, \Theta, \mathcal{D}, \mathcal{W})$ given by:

- $U = (U_1 \cup U_2) \setminus (Y_1 \cup Y_2)$, $X = X_1 \cup X_2$, $Y = Y_1 \cup Y_2$
- $\Sigma^{in} = (\Sigma_1^{in} \cup \Sigma_2^{in}) \setminus (\Sigma_1^{out} \cup \Sigma_2^{out})$, $\Sigma^{int} = \Sigma_1^{int} \cup \Sigma_2^{int}$, $\Sigma^{out} = \Sigma_1^{out} \cup \Sigma_2^{out}$
- $\Theta = \{s \in \mathbf{V} \mid s\lceil V_1 \in \Theta_1 \ \wedge \ s\lceil V_2 \in \Theta_2\}$
- For $i \in \{1,2\}$, define the projection function $\pi_{A_i} : \Sigma \to \Sigma_i$ by $\pi_{A_i}(a) \triangleq a$, if $a \in \Sigma_i$, and $\pi_{A_i}(a) \triangleq e$, otherwise. Then $\mathcal{D}$ is the subset of $\mathbf{V} \times \Sigma \times \mathbf{V}$ given by:
$$(s, a, s') \in \mathcal{D} \quad \Leftrightarrow \quad s\lceil V_1 \xrightarrow{\pi_{A_1}(a)}_{A_1} s'\lceil V_1 \quad \wedge \quad s\lceil V_2 \xrightarrow{\pi_{A_2}(a)}_{A_2} s'\lceil V_2$$
- $\mathcal{W}$ is the set of trajectories over V given by:
$$w \in \mathcal{W} \quad \Leftrightarrow \quad w \downarrow V_1 \in W_1 \ \wedge \ w \downarrow V_2 \in W_2$$

The projection notation π_{A_i}, for $i \in \{1,2\}$, can be extended to states, trajectories and discrete actions. It can be shown that [1]:

Proposition 1 *If A_1 and A_2 are compatible HIOA, then their composition $A_1 \times A_2$ is a HIOA.*

2.3 Executions, Reachable States & System Properties

A *hybrid execution fragment*, α, of a HIOA A is a finite or infinite alternating sequence $\alpha = w_0 a_1 w_1 a_2 w_2 \cdots$, where:

- Each w_i is a trajectory in $\mathcal{W}$ and each a_i is an action in Σ.
- If α is a finite sequence then it ends with a trajectory.
- If w_i is not the last trajectory in α then its domain is a right-closed interval and it is the case that $lstate(w_i) \xrightarrow{a_{i+1}} fstate(w_{i+1})$.

Similar to trajectories, if $\alpha = w_0 a_1 w_1 a_2 w_2 \cdots$ is a hybrid execution fragment, then we define the *limit time* of α by $ltime(\alpha) \triangleq \sum_i ltime(w_i)$ and the *first state* of α by $fstate(\alpha) \triangleq fstate(w_0)$. A hybrid execution fragment, α, is called an *execution* if $fstate(\alpha) \in \Theta$ and is called *finite* if α is a finite sequence and the domain of its final trajectory is a right-closed interval. If

$\alpha = w_0 a_1 w_1 \cdots a_n w_n$ is a finite hybrid execution fragment then we define the *last state* of α by $lstate(\alpha) \triangleq lstate(w_n)$. A finite hybrid execution fragment $\alpha = w_0 a_1 w_1 a_2 w_2 \cdots a_n w_n$ and a hybrid execution fragment $\alpha' = w_0' a_1' w_1' a_2' w_2' \cdots$ of A can be *concatenated* if $w_n \frown w_0'$ is defined and belongs to $\mathcal{W}$. In this case, the *concatenation* $\alpha \frown \alpha'$ is the hybrid execution fragment defined by:

$$\alpha \frown \alpha' \quad \triangleq \quad w_0 a_1 w_1 a_2 w_2 \cdots a_n (w_n \frown w_0') a_1' w_1' a_2' w_2' \cdots$$

A state s' of an automaton A is *reachable from a state s of A* if there exists a finite execution fragment α of A with $fstate(\alpha) = s$ and $lstate(\alpha) = s'$. A state s' is *reachable by A* if it is reachable from some $s \in \Theta$.

Consider an HIOA, A, with variables V. A *derived variable* of A is a function, f, with $dom(f) = \mathbf{V}$. Derived variables will be useful in analyzing the executions of A. A *property*, P, of A is a boolean derived variable of A. If $P(s)$ is true for a state $s \in \mathbf{V}$ we write $s \models P$ and say that "s satisfies property P". For a subset $S \subseteq \mathbf{V}$ we write $S \models P$ if $s \models P$ for all $s \in S$. Let $\mathcal{P}_A$ denote the set of all properties of A.

Definition 1 *A property P of A is* **invariant** *if for all states s reachable by A, $s \models P$. P is* **stable** *if s reachable by A and $s \models P$ imply that for all s' reachable from s, $s' \models P$.*

Lemma 1 *Consider an automaton A and assume that for all reachable states s, $s \models P$ implies that $s' \models P$ for all s' such that:*

- *$\exists w \in \mathcal{W}$ with $dom(w)$ right closed, $fstate(w) = s$ and $lstate(w) = s'$, or,*
- *$\exists a \in \Sigma$ with $s \xrightarrow{a} s'$.*

Then P is a stable property of A. If further $\Theta \models P$, then P is an invariant property of A.

Proof: Consider an arbitrary reachable state, s, of A such that $s \models P$. By definition, for all s_n reachable from s there exists a finite hybrid execution fragment $\alpha = w_0 a_1 w_1 a_2 w_2 \cdots a_n w_n$ with $fstate(\alpha) = s$ and $lstate(\alpha) = s_n$. We show $s_n \models P$ by induction on the length of α.

$s \models P$, therefore, by the lemma assumptions $s_0 \triangleq lstate(w_0) \models P$. For $k \in \{0, 1, \ldots, n\}$, let $s_k = lstate(w_k)$ and for $k \in \{1, \ldots, n\}$, let $s_k' = fstate(w_k)$. All s_k are reachable by A, as they are reachable from s by the finite hybrid execution fragment $\alpha_k = w_0 a_1 \cdots w_k$. Likewise, all s_k' are reachable by A as they are reachable from s by the finite hybrid execution fragment $\alpha_k' = w_0 a_1 \cdots a_k \wp(s_k')$. Assume $s_k \models P$. Then, by the lemma assumptions $s_{k+1}' \models P$, as $s_k \xrightarrow{a_{k+1}} s_{k+1}'$. Likewise, by the lemma assumptions $s_{k+1} \models P$, as w_{k+1} is right closed, $fstate(w_{k+1}) = s_{k+1}'$ and $lstate(w_{k+1}) = s_{k+1}$. The claim follows by induction. By the same argument, if in addition $\Theta \models P$, then P is an invariant property of A. ∎

Note that the proof of the lemma does not require axioms **Init** and **D3**. Therefore the conclusion of the lemma holds even if these axioms are violated. The system A, however, will no longer be an HIOA.

3 Vehicle String Model

Consider a string of N vehicles (Figure 1) moving one behind the other in a single lane, with vehicle 0 coming first. We will be interested in investigating the safety of this string. For this purpose we try to develop a simple yet general model for its dynamics. Our primary consideration is that the modeling framework should impose as few intrinsic limitations as possible while keeping the predicted evolution realistic.

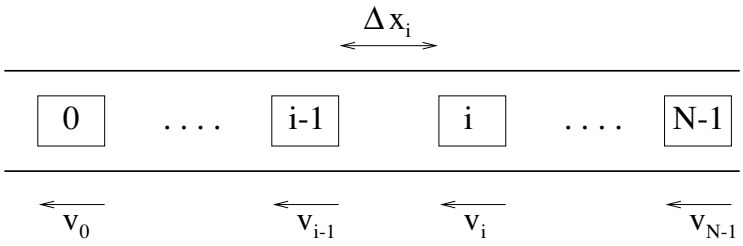

Figure 1: A string of vehicles

3.1 Notation

The overall model will be the composition of a number of HCS (Figure 2). The *plant* will be a hybrid automaton containing the dynamics of all the vehicles in the string. Its evolution will be captured by $2N$ real valued internal variables (x), N real valued input variables (u) and $3N$ real valued output variables (y^p). The plant automaton does not have input or output actions but has internal actions reflecting collisions, vehicles touching at zero relative velocity, etc. Each vehicle, i, is equipped with *sensors*. The sensor automaton S_i reads the values of the plant output variables as inputs and produces m_i real valued output variables (y_i^s). The sensors may have internal variables and actions and will in general contain delay buffers. Finally, each vehicle is equipped with a *controller*. The controller automaton, C_i, reads the corresponding sensor output variables, y_i^s, as inputs and uses them to generate the input variable u_i of the plant. The controller automaton may also have internal variables and actions and will in general contain delay buffers.

We start by developing a model for the plant. The plant is modeled by a HCS $P = (U_P, X_P, Y_P, \Sigma_P^{in}, \Sigma_P^{int}, \Sigma_P^{out}, \Theta_P, \mathcal{D}_P, \mathcal{W}_P)$. P has no input and no output actions, hence $\Sigma_P^{in} = \Sigma_P^{out} = \emptyset$. Here we are only interested in answering questions of "safety", encoded in terms of possible collisions among the vehicles of the string. The answers to these questions will depend on the relative spacing and the velocities of the vehicles, but not their absolute position on the road. Let Δx_i denote the spacing between vehicle i and $i-1$, v_i the speed of vehicle i, acc_i its acceleration and u_i its commanded acceleration[2] and define:

$$x_i = \begin{bmatrix} \Delta x_i \\ v_i \end{bmatrix} \in \mathbb{R}^2, \quad x = \begin{bmatrix} x_0 \\ \vdots \\ x_{N-1} \end{bmatrix} \in \mathbb{R}^{2N}, \quad acc = \begin{bmatrix} acc_0 \\ \vdots \\ acc_{N-1} \end{bmatrix} \in \mathbb{R}^N, \quad u = \begin{bmatrix} u_0 \\ \vdots \\ u_{N-1} \end{bmatrix} \in \mathbb{R}^N$$

Also let $Touching = \{Touching_0, \ldots Touching_N\}$ be a collection of boolean variables and define $X_P = \{x, acc, Touching\}$ and $U_P = \{u\}$. Finally, let:

$$y_i^p = \begin{bmatrix} y_{i1}^p \\ y_{i2}^p \\ y_{i3}^p \end{bmatrix} \in \mathbb{R}^3, \quad y^p = \begin{bmatrix} y_0^p \\ \vdots \\ y_{N-1}^p \end{bmatrix} \in \mathbb{R}^{3N}$$

and define $Y_P = \{y^p\}$.

It remains to specify the set of internal actions Σ_P^{int}, the corresponding transitions, $\mathcal{D}_P$, the set of initial conditions, Θ_P, and the set of trajectories, $\mathcal{W}_P$. The first two will be specified in Section 3.2 while the last two in Section 3.3. Section 3.4 contains some discussion suggesting that

[2] As discussed in Section 3.3, the commanded and actual acceleration may differ when vehicles are touching and pushing each other.

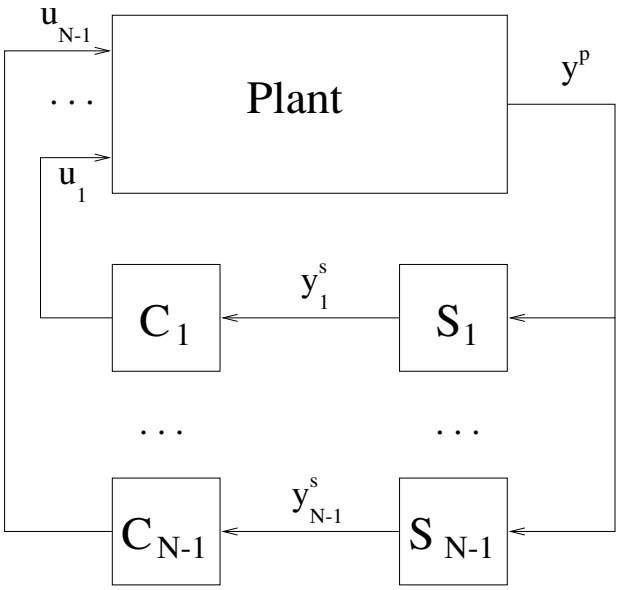

Figure 2: System modules

the resulting model is consistent with physical intuition. The pseudo-code for the plant model is given in Appendix A.

The role of the sensors and controllers is discussed in Section 3.5. Finally, Section 3.6 introduces the notion of safety we consider for this model.

3.2 Plant: Discrete Dynamics

The continuous system evolution can be interrupted by three classes of internal actions: collisions, vehicles touching at zero relative velocity (and subsequently "pushing" against one another) and vehicles moving apart (after having touched). Let $\mathit{Collision} = \{\mathit{Collision}_1, \ldots, \mathit{Collision}_{N-1}\}$, $\mathit{Touch} = \{\mathit{Touch}_1, \ldots, \mathit{Touch}_{N-1}\}$ and $\mathit{Separate} = \{\mathit{Separate}_1, \ldots, \mathit{Separate}_{N-1}\}$ denote the three classes of actions and define $\Sigma_P^{int} = \{\mathit{Collision}, \mathit{Touch}, \mathit{Separate}\}$. All actions are forced, i.e. we assume that the continuous evolution stops as soon as the precondition of an action becomes true, to allow the action to take place.

3.2.1 Collisions

Consider first the case of collisions. Let $\mathit{Collision}_i$ be an internal action that takes place whenever vehicle i collides with vehicle $i-1$. The precondition for $\mathit{Collision}_i$ is:

$$(\Delta x_i = 0) \wedge (v_i > v_{i-1}) \tag{1}$$

To determine the effect of the action we use a simple collision model. After the collision $\Delta x_j' = \Delta x_j$ for all j and $v_j' = v_j$ for all $j \notin \{i, i-1\}$. To determine v_i and v_{i-1} we solve a pair of equations:

$$M_i v_i' + M_{i-1} v_{i-1}' = M_i v_i + M_{i-1} v_{i-1} \tag{2}$$

$$v_{i-1}' - v_i' = (v_i - v_{i-1})\alpha_i \tag{3}$$

where M_i is the mass of vehicle i while α_i is the coefficient of restitution, a measure of the energy lost in the collision. Equation (2) is the *conservation of momentum equation* while Equation (3) is

referred to as the *restitution equation*. This collision model for a pair of vehicles is fairly accurate [2]. It has the advantage that a solution for x' always exists and can be found analytically. By appropriate choice of α (possibly as a function of the speeds) this collision model can capture a wide range of collision scenarios. To maintain a certain level of generality in the subsequent discussion we will typically assume that the coefficient of restitution is a function of the relative velocity $v_{i-1} - v_i$ at impact and will denote it by $\alpha_i(\cdot)$. To ensure that the model is realistic we impose the following assumption:

Assumption 1 *For all i, $M_i > 0$ and $\alpha_i(v) \in [0, 1]$ for all $v > 0$.*

Multiple instantaneous collisions are also possible in this model. These are situations where there exist N_1 and N_2 with $0 \leq N_1 < N_2 < N$ such that $\Delta x_{N_1} \neq 0$, $\Delta x_{N_2+1} \neq 0$ (if any) and for all i with $N_1 < i \leq N_2$, $\Delta x_i = 0$ and $v_i > v_{i-1}$. The value, x', of the state after the collision again satisfies $\Delta x_i' = \Delta x_i$ for all i and $v_i' = v_i$ for all $i < N_1$ or $i > N_2$. To determine the values of v_i for $N_1 \leq i \leq N_2$ we propose to resolve the multiple collision as a sequence of pairwise collisions, according to equations (2) and (3). The pairwise resolutions will keep taking place as long as there exists a j with $N_1 < j \leq N_2$ such that $v_j > v_{j-1}$. When this condition is violated we will say that the multiple collision has been resolved. The motivation behind this convention is that multiple instantaneous collisions are more of a mathematical necessity than a realistic concern. In "most" practical situations collisions will take place close to one another in time but not instantaneously. We would like the resolution convention to be "consistent" in this case. Our multiple collision arrangement reduces to a pairwise collision if $N_1 = N_2 - 1$.

3.2.2 Vehicles Touching

Now consider what happens when vehicles touch at zero relative velocity. This situation may arise because the continuous dynamics bring the vehicles together at zero speed, or after a collisions with $\alpha = 0$. Let $Touch_i$ be an internal action that takes place whenever vehicle i touches vehicle $i-1$ with zero relative velocity. The precondition for $Touch_i$ is:

$$(Touching_i = \text{False}) \wedge (\Delta x_i = 0) \wedge (v_i = v_{i-1}) \wedge (acc_i \geq acc_{i-1}) \tag{4}$$

The effect of $Touch_i$ is simply to declare the two vehicles as touching. In the usual notation:

$$Touching_i' = \text{True}$$

The value of $Touching_i$ will be used in Section 3.3 to determine the acceleration, acc_i of vehicle i.

3.2.3 Vehicles Separating

Finally, consider what happens when vehicles that are touching start moving away from one another. Let $Separate_i$ be an internal action that takes place whenever vehicle i is already touching vehicle $i-1$ and starts to move away. The precondition for $Separate_i$ is:

$$(Touching_i = \text{True}) \wedge [(acc_i < acc_{i-1}) \vee (v_i < v_{i-1})] \tag{5}$$

The effect of $Separate_i$ is simply to declare the two vehicles as no longer touching. In the usual notation:

$$Touching_i' = \text{False}$$

Note that, vehicles are declared as no longer touching as soon as they start moving apart, either because of a difference in deceleration or because of a difference in velocity (in case of a collision).

3.3 Plant: Continuous Dynamics

3.3.1 Initial Condition and Input Constraints

First we introduce some assumptions that will help ensure the system evolution remains realistic. We impose the following constraint on the initial conditions:

Assumption 2 *For all* $i = 0, \ldots, N-1$, $\Delta x_i(0) \geq 0$, $v_i(0) \geq 0$. $Touching_i(0) = \text{False}$. $Touching_N(0) = \text{False}$.

Physical limitations constrain the valuations of the input variables to lie in a rectangular compact set, i.e. $u_i(t) \in [a_i^{min}, a_i^{max}]$ for all i and for all t. The values of a_i^{min} and a_i^{max} are determined by the vehicle characteristics (engine, brakes, tires, etc.). To ensure that the model is realistic we impose the following assumption:

Assumption 3 *For all* i, $a_i^{min} < 0 < a_i^{max}$.

If needed at a later stage, the requirement on a_i^{min} and a_i^{max} can be relaxed to allow for "brakes on" ($a_i^{max} < 0$) and "brakes off" (possibly $a_i^{min} > 0$) failures.

3.3.2 Dynamical Equations

The set of trajectories $\mathcal{W}_P$ will be generated by a pair of functions (f, h). Assume there are no vehicles ahead of the string and set $\Delta x_0 \equiv \infty$. Then, for $i = 1, \ldots, N-1$ the laws of motion imply that:

$$\begin{aligned} \dot{\Delta x}_i(t) &= v_{i-1}(t) - v_i(t) \\ \dot{v}_i(t) &= acc_i(t) \end{aligned}$$

or, in standard vector notation:

$$\dot{x}(t) = \begin{bmatrix} 0 \\ 0 \\ v_0(t) - v_1(t) \\ 0 \\ v_1(t) - v_2(t) \\ \vdots \\ v_{N-2}(t) - v_{N-1}(t) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ acc_0(t) \\ 0 \\ acc_1(t) \\ 0 \\ \vdots \\ 0 \\ acc_{N-1}(t) \end{bmatrix} \triangleq f(x(t), acc(t)) \tag{6}$$

The value of the actual acceleration, acc_i, of vehicle i depends on the acceleration commanded by the controller of that vehicle, u_i, and on whether the vehicle is touching vehicle $i-1$ or vehicle $i+1$. In the case when the vehicles are not touching we simply set the actual acceleration equal to the commanded acceleration, i.e.:

$$(Touching_i = \text{False}) \wedge (Touching_{i+1} = \text{False}) \Longrightarrow acc_i = u_i \tag{7}$$

As long as the vehicles are not touching, f is a linear map in x and u and therefore is globally Lipschitz.

The case where vehicles are touching is more complicated. The reason is that when vehicles are pushing against one another, there are forces exerted from one vehicle to the other. Therefore, the actual acceleration of a vehicle depends not only on the acceleration commanded by its own

controller, but also on the accelerations commanded by the controllers of the neighboring vehicles that are pushing against it. We first motivate the proposed solution informally for two touching vehicles. We assume that when a vehicle, say i, is by itself (i.e. $(Touching_i = \text{False}) \wedge (Touching_{i+1} = \text{False})$) its acceleration is the result of a force $F_i = M_i u_i$ exerted by the road to the vehicle through the tires. In the case where two vehicles, say i and $i-1$ are touching, i.e. $(Touching_i = \text{True}) \wedge (Touching_{i+1} = \text{False}) \wedge (Touching_{i-1} = \text{False})$, we assume that the road still exerts forces F_i and F_{i-1} to these two vehicles. However, if $u_i \geq u_{i-1}$, a force, F, is also exerted from one vehicle to the other. In this case, the vehicles remain touching and accelerate at the same rate, therefore:

$$\left.\begin{array}{l} M_i acc_i = F_i - F \\ M_{i-1} acc_{i-1} = F_{i+1} + F \\ acc_i = acc_{i-1} \end{array}\right\} \Longrightarrow acc_i = acc_{i-1} = \frac{M_i u_i + M_{i-1} u_{i-1}}{M_i + M_{i-1}}$$

The vehicles separate as soon as $u_i < u_{i-1}$.

3.3.3 Multiple Touching Vehicles

We try to extend this two vehicle construction to an arbitrary number of touching vehicles. We first introduce some abstract definitions and then show how they apply to the vehicle problem. Consider a nonempty finite subset of the natural numbers $S \subset \mathbb{N}$ and let $\min(S)$ and $\max(S)$ denote its minimum and maximum element respectively. S is a *segment* if it consists of consecutive numbers. A *subsegment* of a segment S is any subset of S that is also a segment. For segments S_1 and S_2 with $\max(S_1) = \min(S_2) - 1$ we define their *concatenation* simply by $S_1 \cup S_2$. Whenever defined, the concatenation of two segments is also a segment; we denote this segment by $S_1 S_2$.

A *weighted average function on* S is any function $a : 2^S \to \mathbb{R}$ such that for all L, R subsegments of S:

$$\min\{a(L), a(R)\} \leq a(LR) \leq \max\{a(L), a(R)\} \tag{8}$$

whenever the concatenation LR is defined. Given a weighted average function on a segment, all subsegments naturally inherit a weighted average. A segment S with a weighted average function a is *unsplitable* if:

$$S = LR \Rightarrow a(L) \leq a(R)$$

Proposition 2 *If A and B are two unsplitable subsegments of S and $A \cap B \neq \emptyset$, then $A \cup B$ is an unsplitable subsegment of S.*

Proposition 3 *If A and B are two unsplitable subsegments of S, AB is defined and $a(A) \leq a(B)$, then AB is an unsplitable subsegment of S.*

A *partition of* S is a finite collection $S_1, \ldots, S_n$ where $S = \cup_{k=1}^{n} S_k$ and for all k, S_k is a segment and $S_k \cap S_l = \emptyset$ for $l \neq k$. Without loss of generality assume that $\min(S) = \min(S_1)$ and for all $1 < k \leq n$, $\min(S_k) = \max(S_{k-1}) + 1$ and write $S = S_1 S_2 \ldots S_n$. A partition of $S_1 \ldots S_n$ of S is called a *maximal partition* if:

1. for all $k = 1, \ldots, n$, S_k is unsplitable,
2. either $n = 1$ or for all $k = 2, \ldots, n$, $a(S_{k-1}) > a(S_k)$.

Proposition 4 *If $S_1 \ldots S_n$ is a maximal partition of S, $1 \leq l \leq k \leq n$ and $\hat{S}_{lk} = \cup_{m=l}^{k} S_m$ then $a(\hat{S}_{lk}) \geq a(S_k)$.*

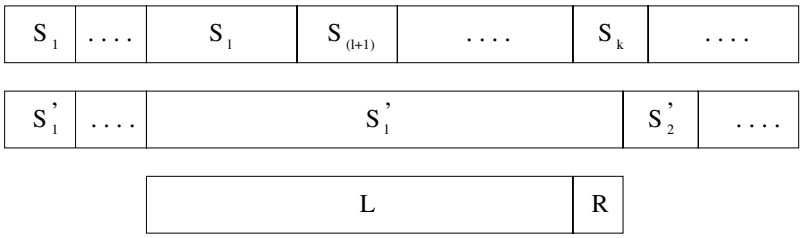

Figure 3: Maximal Partition

Theorem 1 *For every segment, S, and every weighted average function, a, on S there exists a unique maximal partition.*

Proof: For existence, let $\mathcal{S}$ denote the set of all unsplitable subsegments of S. Let $\{S_1, S_2, \ldots, S_n\}$ denote a collection of distinct maximal elements of $\mathcal{S}$ (i.e. for all $k = 1, \ldots, n$, $S_k \neq S_l$ for $l \neq k$ and $S_k \subseteq S' \in \mathcal{S}$ implies that $S_k = S'$) that covers S. Such a collection exists, as for all $i \in S$, $\{i\}$ is vacuously an unsplitable segment; therefore, each $i \in S$ belongs to a maximal subset of $\mathcal{S}$. We claim that $\{S_1, S_2, \ldots, S_n\}$ is a maximal partition of S. First note that $S_k \cap S_l = \emptyset$ for all $k \neq l$. Otherwise, $S_k \cup S_l \in \mathcal{S}$, as S_k and S_l are unsplitable and therefore, by Proposition 2, $S_k \cup S_l$ is also unsplitable. As S_k and S_l are both maximal this implies that $S_k = S_l = S_k \cup S_l$ which contradicts the assumption that S_k and S_l are distinct. Further, $S_k \in \mathcal{S}$, therefore by definition S_k is unsplitable, for all $k = 1, \ldots, n$. Finally, without loss of generality, assume $S = S_1 S_2 \ldots S_n$ and show $a(S_{k-1}) > a(S_k)$. If $n = 1$ the claim follows. If $n > 1$ and $a(S_{k-1}) \leq a(S_k)$, $S_{k-1}S_k \in \mathcal{S}$, as S_{k-1} and S_k are both unsplitable and therefore, by Proposition 3, $S_{k-1}S_k$ is also unsplitable. This contradicts the maximality of S_k and S_{k-1}.

To show uniqueness, assume, for the sake of contradiction that two different maximal partitions, $S_1 \ldots S_n$ and $S'_1 \ldots S'_m$, exist. Consider the first segment for which the two partition differ $S_l \neq S'_l$. Without loss of generality assume that $S_l \subset S'_l$. Define k as the segment for which $S_{k+1} \cap S'_l = \emptyset$ and $S_k \cap S'_l \neq \emptyset$. It is easy to see that the number k is well defined. Moreover, $k > l$ as $S_l \subset S'_l$ and $S_l \neq S'_l$ imply that $S_{l+1} \cap S'_l \neq \emptyset$ (refer to Figure 3). Define:

$$L = \bigcup_{m=l}^{k-1} S_m \quad R = S_k \cap S'_l$$

As $S_1 \ldots S_n$ is assumed to be maximal, $a(L) \geq a(S_{k-1})$ by proposition 4. Further, S_k unsplitable implies that $a(R) \leq a(S_k)$, by definition of weighted average. Overall, the maximality of $S_1 \ldots S_n$ implies that the partition $S'_l = LR$ satisfies $a(L) \geq a(S_{k-1}) > a(S_k) \geq a(R)$, which contradicts the maximality of $S'_1 \ldots S'_m$. ■

An algorithm for calculating the unique maximal partition of a segment is given in Appendix B. Returning to our vehicle example, assume there exist i, j satisfying $0 < i < j < N$ such that vehicles i to j are touching each other, i.e.:

$$(\mathit{Touching}_i = \text{False}) \wedge (\mathit{Touching}_{j+1} = \text{False}) \wedge \left(\bigwedge_{k=i+1}^{j} \mathit{Touching}_k = \text{True} \right)$$

Define the segment $S = \{i, \ldots, j\}$ and for every subset $S' \subseteq S$ consider the function:

$$a(S') = \frac{\sum_{k \in S'} M_k u_k}{\sum_{k \in S'} M_k} \tag{9}$$

Proposition 5 *a is a weighted average function on S.*

To determine the acceleration of the vehicles in this collection at a given instant, let $S_1 \dots S_n$ be the maximal partition of S at that instant and for all $k = 1, \dots, n$ set:

$$acc_l = a(S_k) \text{ for all } l \in S_k \tag{10}$$

The weighted average a is a linear function of the commanded acceleration u. Therefore, as long as the partition does not change, the vector field f generating the vehicle dynamics will be linear in both x and u, and hence globally Lipschitz. If the partition changes, some of the *Separate* actions will take place, splitting S into smaller segments.

3.3.4 Output Map

It remains to specify the outputs. We assume that in principle all the internal variables can be made available to the controllers. Limitations imposed by current sensing and communication technology should be incorporated in the sensor automata. We therefore set:

$$y_i^p(t) = \begin{bmatrix} x_i(t) \\ acc_i(t) \end{bmatrix} \Longrightarrow y^p(t) = h(x(t), acc(t))$$

As before, h is a linear map as long as $Touching_i$ remain constant and therefore it is globally Lipschitz.

3.4 Plant: Consistency & Limitations

The pairwise collisions that will be used to resolve a given multiple collision can be ordered in a number of different ways. One would hope the outcome of the resolution will depend only on the arrangement (velocities, masses and restitution) and not on the order of resolution.

Proposition 6 *If $\alpha_i \equiv 1$ and $M_i = M_j$ for all $N_1 \leq i, j \leq N_2$ then all possible orders of pairwise resolution lead to $v'_{N_1} = v_{N_2}$, $v'_{N_1+1} = v_{N_2-1}$, ..., $v'_{N_2} = v_{N_1}$ (i.e. the order of the velocities is reversed).*

Unfortunately this statement is not true in general:

Proposition 7 *If $\alpha_i < 1$ or $M_i \neq M_j$ for some $i, j \in [N_1, N_2]$, the state after the collision is resolved, x' , may depend on the order in which the collisions are resolved.*

This ambiguity is rather disturbing. To ensure that any theorems we prove remain valid we will have to show that they hold for *any possible ordering* in the resolution of multiple collisions. In other words, we allow our model to exhibit nondeterminism with respect to multiple collision resolution and prove that all claims hold for any nondeterministic choice.

To ensure that the proposed plant model agrees with physical intuition we show the following lemma:

Lemma 2 *Under Assumptions 1, 2 and 3, the plant automaton is such that:*

1. *For every segment S of touching vehicles* $\min_{i \in S}(u_i) \leq a(S) \leq \max_{i \in S}(u_i)$.
2. *Immediately after $Collision_i$, $v_i \leq v_{i-1}$.*

3. *Let E_i be the total energy of vehicles i and $i-1$ before Collision$_i$ occurs:*

$$E_i = \frac{1}{2}M_i v_i^2 + \frac{1}{2}M_{i-1}v_{i-1}^2 \tag{11}$$

The energy, E_i', after Collision$_i$ satisfies $E_i' \leq E_i$.

4. $(\text{Touching}_0 = \textit{False}) \wedge (\text{Touching}_N = \textit{False})$ *is an invariant property of the plant.*

5. $\wedge_{i=1}^{N-1}\left[(\text{Touching}_i = \textit{True}) \Rightarrow (\Delta x_i = 0)\right]$ *is an invariant property of the plant.*

6. $\wedge_{i=1}^{N-1}\left[(\Delta x_i > 0) \Rightarrow (\text{Touching}_i = \textit{False})\right]$ *is an invariant property of the plant.*

7. $\wedge_{i=0}^{N-1}\left[\Delta x_i \geq 0\right]$ *is an invariant property of the plant.*

Proof: Part 1 follows from $\frac{\sum_{k\in S} M_k \min_{i\in S}(u_i)}{\sum_{k\in S} M_k} \leq \frac{\sum_{k\in S} M_k u_k}{\sum_{k\in S} M_k} \leq \frac{\sum_{i\in S} M_k \max_{i\in S}(u_i)}{\sum_{i\in S} M_k}$.

Part 2 follows from equations (1) and (3) as $\alpha_i \geq 0$ by Assumption 1.

For Part 3 we explicitly solve the pairwise collision equations (2) and (3). Without loss of generality set $i = 2$ and let $\alpha = \alpha_2$ and $M = M_2/M_1$. Some algebra leads to:

$$v_1' = \frac{(1-\alpha M)v_1 + M(1+\alpha)v_2}{1+M}, \quad v_2' = \frac{(1+\alpha)v_1 + (M-\alpha)v_2}{1+M} \tag{12}$$

Substituting into the formula for the energy and after some manipulation one gets:

$$\begin{aligned} & E_i' = \frac{M_1}{2}\left(\frac{(1+\alpha^2 M)v_1^2 + M(M+\alpha^2)v_2^2 + 2M(1-\alpha^2)v_1 v_2}{M+1}\right) \\ \Rightarrow \quad & E_i - E_i' = \frac{M_1}{2}\left(\frac{M(1-\alpha^2)(v_1^2 + v_2^2 - 2v_1 v_2)}{M+1}\right) \\ \Rightarrow \quad & E_i - E_i' = \frac{M_1 M_2 (1-\alpha^2)(v_1 - v_2)^2}{2(M_1+M_2)} \end{aligned} \tag{13}$$

By assumption 1, $M_i > 0$ and $\alpha_i \in [0,1]$, therefore the right hand side of equation (13) is always non-negative.

Part 4 is trivial, as *Touching*$_0$ and *Touching*$_N$ are set to False by Assumption 2 and are unaffected by both trajectories and actions.

For Part 5, note that *Touching*$_i$ is initially False for all i by Assumption 2. Therefore the property is initially true. Consider an arbitrary element of the conjunction, say (*Touching*$_i$ = True) $\Rightarrow (\Delta x_i = 0)$. Consider first the discrete transitions. Assume (*Touching*$_i$ = True) $\Rightarrow (\Delta x_i = 0)$ is true at the pre-state of *Collision*$_j$ for some $j \in \{1, \ldots, N-1\}$. Then (*Touching*$_i$ = True) $\Rightarrow (\Delta x_i = 0)$ is also true at the post-state, as both *Touching*$_i$ and $\Delta x_i 0$ are unaffected by the action.

Assume (*Touching*$_i$ = True) $\Rightarrow (\Delta x_i = 0)$ is true at the pre-state of *Touch*$_j$ for some $j \in \{1, \ldots, N-1\}$. If $j \neq i$ (*Touching*$_i$ = True) $\Rightarrow (\Delta x_i = 0)$ is also true at the post-state, as *Touching*$_i$ and $\Delta x_i = 0$ are unaffected by the action. If $i = j$, (*Touching*$_i$ = False) $\wedge (\Delta x_i = 0)$ must be true at the pre-state. Therefore, (*Touching*$_i$ = True) $\wedge (\Delta x_i = 0)$ will be true at the post-state, as Δx_i is unaffected by the action.

Assume (*Touching*$_i$ = True) $\Rightarrow (\Delta x_i = 0)$ is true at the pre-state of *Separate*$_j$ for some $j \in \{1, \ldots, N-1\}$. If $j \neq i$ (*Touching*$_i$ = True) $\Rightarrow (\Delta x_i = 0)$ is also true at the post-state, as *Touching*$_i$ and $\Delta x_i = 0$ are unaffected by the action. If $i = j$, *Touching*$_i$ = False at the post-state, therefore (*Touching*$_i$ = True) $\Rightarrow (\Delta x_i = 0)$ will again be true.

Now consider the continuous evolution. Assume that $(\mathit{Touching}_i = \text{True}) \Rightarrow (\Delta x_i = 0)$ is true at some state, s, and consider all trajectories that start at s. Distinguish two cases. If $\mathit{Touching}_i$ is false at s, then it will also be false at the final state of the trajectory, as, by definition of $\mathcal{W}_P$, the value of $\mathit{Touching}_i$ remains constant along trajectories. Therefore, $(\mathit{Touching}_i = \text{True}) \Rightarrow (\Delta x_i = 0)$ will be true at the final state.

If $\mathit{Touching}_i$ is true at s, then $(\Delta x_i = 0)$ must also be true. If at this point $(acc_i < acc_{i-1}) \vee (v_i < v_{i-1})$ is true the precondition of action $\mathit{Separate}_i$ is satisfied. If at this point $(v_i > v_{i-1})$ is true, the precondition of action $\mathit{Collision}_i$ is satisfied. In either case the trajectory terminates (by definition of $\mathcal{W}_P$) while $(\mathit{Touching}_i = \text{True}) \Rightarrow (\Delta x_i = 0)$ is still true. If $(acc_i \geq acc_{i-1}) \wedge (v_i = v_{i-1})$ is true the system proceeds along the continuous trajectory[3]. acc_i and acc_{i-1} are determined by the maximal partition of a collection of touching vehicles (which may include more than vehicles i and $i-1$). By construction of the maximal partition, $acc_i \leq acc_{i-1}$ ($acc_i < acc_{i-1}$ if i is the first vehicle of an element of the partition and $acc_i = acc_{i-1}$ otherwise). Overall, continuous evolution proceeds as long as $(acc_i \geq acc_{i-1}) \wedge (v_i = v_{i-1}) \wedge (acc_i \leq acc_{i-1})$, i.e. as long as $(acc_i = acc_{i-1}) \wedge (v_i = v_{i-1})$. In this case, $\dot{\Delta} x_i = v_{i-1} - v_i = 0$ and $\ddot{\Delta} x_i = acc_{i-1} - acc_i = 0$ and therefore $\Delta x_i = 0$ at the last state of the trajectory, as $\Delta x_i = 0$ at s. Overall, $(\mathit{Touching}_i = \text{True}) \Rightarrow (\Delta x_i = 0)$ is preserved by continuous evolution. Part 5 follows by Lemma 1.

For Part 6, note again that $\mathit{Touching}_i$ is initially False for all i by Assumption 2. Therefore the property is initially true. Consider an arbitrary element of the conjunction, say $(\Delta x_i > 0) \Rightarrow (\mathit{Touching}_i = \text{False})$. Consider first the discrete transitions. Assume $(\Delta x_i > 0) \Rightarrow (\mathit{Touching}_i = \text{False})$ is true at the pre-state of $\mathit{Collision}_j$ for some $j \in \{1, \ldots, N-1\}$. Then, the property will also be true at the post-state, as both $\mathit{Touching}_i$ and Δx_i are unaffected by the action.

Assume $(\Delta x_i > 0) \Rightarrow (\mathit{Touching}_i = \text{False})$ is true at the pre-state of Touch_j for some $j \in \{1, \ldots, N-1\}$. If $j \neq i$, the property will also be true at the post-state, as $\mathit{Touching}_i$ and Δx_i are unaffected by the action. If $i = j$, $(\mathit{Touching}_i = \text{False}) \wedge (\Delta x_i = 0)$ must be true at the pre-state. Therefore, $(\mathit{Touching}_i = \text{True}) \wedge (\Delta x_i = 0)$ will be true at the post-state, as Δx_i is unaffected by the action. Hence, $(\Delta x_i > 0) \Rightarrow (\mathit{Touching}_i = \text{False})$ is again true at the post-state.

Assume $(\Delta x_i > 0) \Rightarrow (\mathit{Touching}_i = \text{False})$ is true at the pre-state of $\mathit{Separate}_j$ for some $j \in \{1, \ldots, N-1\}$. If $j \neq i$, the property will also be true at the post-state, as $\mathit{Touching}_i$ and $\Delta x_i = 0$ are unaffected by the action. If $i = j$, $\mathit{Touching}_i = \text{True}$ at the pre-state, therefore $\Delta x_i = 0$ at the pre-state, by Part 5. Therefore, $(\Delta x_i > 0) \Rightarrow (\mathit{Touching}_i = \text{False})$ will again be true at the post-state, as Δx_i is unaffected by the action.

The proof that $(\Delta x_i > 0) \Rightarrow (\mathit{Touching}_i = \text{False})$ is preserved by continuous evolution is identical to the same proof for Part 5. Part 6 follows by Lemma 1.

Finally, for Part 7, note that the property is true at the initial state, by Assumption 2. The property is preserved by discrete transitions, as they all leave the Δx_i unaffected. The proof for the continuous evolution follows by the argument given for Part 5. ■

Part 3 shows that the proposed collision model can simulate a wide range of energy loss situations, from perfectly elastic (no energy loss, $\alpha_i = 1$) to plastic (vehicles do not bounce at all, $\alpha_i = 0$). Note that no claim is made about the vehicles not moving backwards. From equation (12), v_2' may in fact be negative, if, for example, $v_1 = 0$, $M < 1$ and $\alpha = 1$ (i.e., a light vehicle hits a stopped heavy vehicle elastically). Therefore, collisions may force vehicles to go backwards.

The main limitation of our model is that is does not account for the lateral motion of the vehicles. We assume that all vehicles effectively move along a straight line. This assumption may be unrealistic, especially in the presence of collisions when large forces and moments can be exerted

[3] Touch_i can not take place as $\mathit{Touching}_i$ is true.

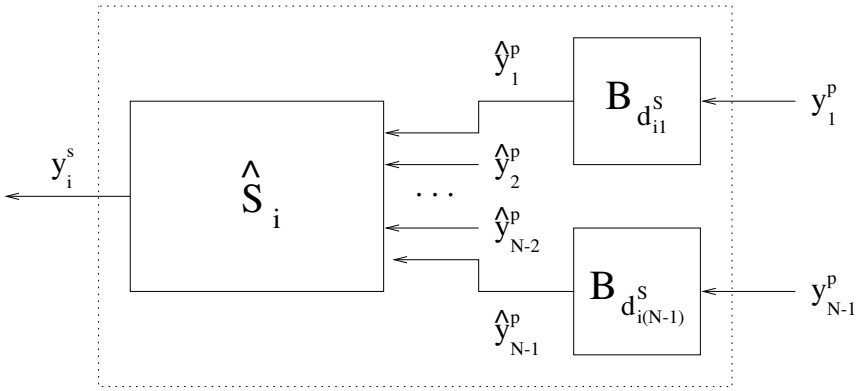

Figure 4: Sensor module S_i

from one vehicle to another. The situation will be even worse when the vehicles move along a curved road.

3.5 Sensors and Controllers

The sensors provide the controllers with information about the plant variables. The sensors of each vehicle can be modeled by an automaton $S_i = (U_{S_i}, X_{S_i}, Y_{S_i}, \Sigma^{in}_{S_i}, \Sigma^{int}_{S_i}, \Sigma^{out}_{S_i}, \Theta_{S_i}, \mathcal{D}_{S_i}, \mathcal{W}_{S_i})$. Here we only impose minimal limitations on the sensing arrangement. In particular we only require that $y^p \in U_{S_i}$ and define $Y_{S_i} = \{y^s_i\}$. For the trajectory set, $\mathcal{W}_{S_i}$, we only assume that each piece of information (plant output variable) can be made available to the controller with some delay. We assume that the delay depends only on the relative position of the vehicles in the string. This assumption can easily be relaxed, at the expense of complicating the notation. A typical sensor in this case is shown in Figure 4. d^S_{ij} is the *sensing delay*, i.e. the time it takes for information about vehicle j to reach vehicle i.

The heart of the sensing arrangement is now encoded by the HCS $\hat{S}_i$. The automaton can in general be very complicated: it may contain additional input variables (to model sensing noise for example), internal variables (to model data filtering or sensor fusion), internal actions (to model fault detection), etc. In subsequent sections we will only consider very simple sensors, whose output variable values are the same as the (delayed) values of some of their input variables. In this case $\hat{S}_i$ can be described by a projection map:

$$\begin{aligned} h_i : \mathbb{R}^{3N} \times \mathbb{R}^N &\longrightarrow \mathbb{R}^{m_i} \\ \hat{y}^p &\longrightarrow y^s_i \end{aligned}$$

The controller for vehicle i uses the readings of the corresponding sensors, y^s_i, to calculate at each time instant the value of the control u_i. The controller of each vehicle is modeled by an automaton $C_i = (U_{C_i}, X_{C_i}, Y_{C_i}, \Sigma^{in}_{C_i}, \Sigma^{int}_{C_i}, \Sigma^{out}_{C_i}, \Theta_{C_i}, \mathcal{D}_{C_i}, \mathcal{W}_{C_i})$. We again try to impose minimal limitations on the controller arrangement. We only require that $y^s_i \in U_{C_i}$ and that $u_i \in Y_{C_i}$. For the trajectory set, $\mathcal{W}_{C_i}$, we only assume that the controller can influence the plant after some delay. A typical controller is then shown in Figure 5. d^A_i is the *actuation delay*, i.e. the time it takes for the control calculated by controller i to be implemented by vehicle i.

The heart of the controller is encoded by the HCS $\hat{C}_i$. This automaton can again be very complicated in general. It may contain additional input variables (to model actuation uncertainty for example), internal variables (to model dynamic controllers), internal actions, etc. Moreover,

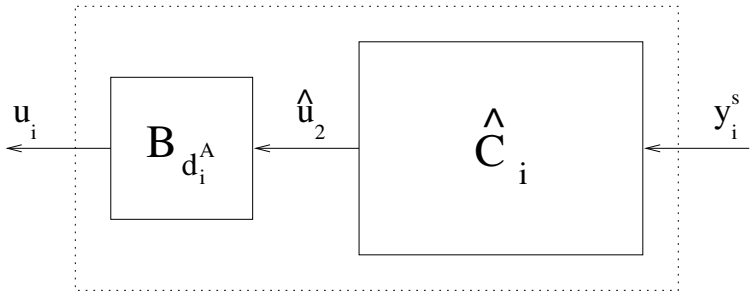

Figure 5: Controller module C_i

the controller and sensing automata may contain additional input/output variables or actions to coordinate with one another (to facilitate fault detection for example). Here we will only make use of very simple controller automata that can be encoded by a map:

$$\begin{array}{rcl} g_i : \mathbb{R}^{m_i} & \longrightarrow & [a_i^{min}, a_i^{max}] \\ y_i^s(t) & \longmapsto & \hat{u}_i(t) \end{array}$$

To ensure that the model is realistic we impose the following assumption:

Assumption 4 *For all i, j, $d_{ij}^S \geq 0$, $d_i^A \geq 0$ and y_i^s is independent of u_i in h_i.*

The bound on the delays is imposed to ensure that the sensor/controller composition is causal, i.e. it does not produce inputs for the plant that depend on future values of the plant state. The independence of y_i^s from u_i is to avoid the possibility of ill-posed compositions between the sensors, the controllers and the plant in the case where all the delays are zero. This last assumption is a minor technicality, u_i is anyway already available to the controller C_i that calculates it and therefore there is not need for the sensor S_i to include it in the y_i^s information. It is easy to see that under Assumption 4:

Lemma 3 *P, C_i and S_i for $i = 0, \ldots, N-1$ are compatible.*

3.6 System Parameters and Measures of Safety

The discussion so far has specified a class of models. The class is parameterized by a relatively small number of parameters. For each $i, j = 0, \ldots, N-1$ these parameters are:

- the actuation delay: $d_i^A \in \mathbb{R}_+$
- the sensing delays: $d_{ij}^S \in \mathbb{R}_+$,
- the mass: $M_i > 0$,
- the acceleration bounds, $a_i^{min}, a_i^{max} \in \mathbb{R}$,
- the restitution, $\alpha_i : \mathbb{R}_+ \to [0, 1]$.

Overall this gives $4N + N^2$ real parameters and the restitution functions. The class of models is further parameterized by:

- the initial conditions (including those for the delay buffers),

- the sensing structure $\hat{S}_i$, $i = 0, \ldots, N-1$,
- the control structure $\hat{C}_i$, $i = 0, \ldots, N-1$.

A *string instance* (or simply a *string*) is a HCS obtained by specifying all the above elements, i.e. assigning values to all parameters, fixing initial conditions, and giving HCS models for the sensors and controllers.

The executions of a string may involve collisions among the vehicles. The string is said to *generate* a sequence of collisions:

$$C = \{(i_k, \Delta v_k, T_k)\}_{k=1}^{K} \tag{14}$$

with $i_k \in \{1, \ldots, N-1\}$, $\Delta v_k > 0$, $T_k \geq T_{k-1} \geq 0$, if there exists an execution such that for all k, $(i_k, \Delta v_k, T_k) \in C$ if and only if $Collision_{i_k}$ occurs at time T_k in the execution with relative velocity Δv_k. For multiple collisions, all pairs of colliding vehicles appear individually. Note that, because of nondeterminism in the order of resolution for multiple collisions many different C's can be generated by the same string.

We are interested in defining the system performance in terms of the severity of the collisions experienced by the vehicles. Following [3], the relative velocity at impact is used as a measure of collision severity[4]. The performance measure can now be thought of as a function, *Safety*, mapping the system executions (in particular the collision sequence C) to a real number. One possible choice for this function is:

$$Safety : C \longmapsto \max_k \{\Delta v_k\} \tag{15}$$

If $K = 0$ define $Safety(C) = 0$[5]. We would like to keep the relative velocity of all collisions below a certain threshold, $v_A \geq 0$, i.e. guarantee that for all sequences C generated by the string $Safety(C) \leq v_A$. A commonly used threshold is $v_A = 3ms^{-1}$ [3].

The requirement for safety, stated above in terms of the system executions, can also be cast in the form of an invariant for the string.

Definition 2 *A string is* **safe** *if* $\bigwedge_{i=1}^{N-1} [(\Delta x_i = 0) \Rightarrow (v_i \leq v_{i-1} + v_A)]$ *is an invariant property. Otherwise the string is* **unsafe**.

It is easy to see that:

Proposition 8 *A string is safe if and only if* $\text{Safety}(C) \leq v_A$ *for all possible executions.*

4 Emergency Deceleration

4.1 Background

We introduce the *emergency deceleration maneuver*, the scenario we will attempt to analyze in the remaining of this paper. This is a situation where the first vehicle in the string applies maximum deceleration until it comes to a stop, thus endangering the remaining vehicles of the string. We would like to determine the conditions under which the remaining vehicles can maintain their safety despite this "malicious" behavior of the leader.

[4] If one would like to consider different performance measures, more information may need to be added to the collision sequence C.

[5] The proposed function reflects the severity of the worst collision. Other measures can be defined by appropriate choice of *Safety*. For example, $Safety(C) = K$ reflects the total number of collisions, $Safety(C) = \frac{1}{K}\sum_{k=1}^{K} \Delta v_k$ reflects the average relative velocity of collision, etc.

The safety of general strings of vehicles has been analyzed using a number of techniques. Most results in the literature start by partly characterizing the string instance by determining "automata" for the sensors and controllers and then trying to establish the range of initial conditions and parameters for which the string is safe. This type of analysis has led to conditions under which pairs of vehicles are guaranteed not to collide [4, 5] or at least experience safe collisions [6, 5, 7, 8]. In some cases the conditions have also been extended to longer or even infinite strings [9, 10]. Perhaps the most challenging problem in this area has been the design of controllers for platoons of vehicles. A *platoon* is a string of very tightly spaced vehicles. Typically intra-platoon spacings are of the order of 1-2 meters. The work of Swaroop [9] has shown that in order to maintain the stability of the string at such tight spacings each vehicle, i, needs to have access to information about its own internal variables x_i, the internal and input variables of the vehicle ahead y^p_{i-1} as well as the internal and input variables of the first vehicle in the string y^p_0. Under this sensor arrangement, controllers were designed in [9] to guarantee the safe operation of the platoon under a reasonably wide range of initial conditions and parameter values.

The safety of the controllers in [9] relies on the assumption that the behavior of the first vehicle is in some sense "reasonable". This means that the controller C_0 takes into account the limitations of the rest of the vehicles in the string when calculating u_0. For example, the controllers of [9] require that u_0 be bounded below by a function of a_i^{min} for all $i \geq 0$. This requirement is clearly violated in the case of the emergency deceleration maneuver. It is conjectured [11] that the platoon is going to be safe even in this case. The justification is that collisions are going to take place in rapid succession, because the vehicles are all close to one another. Therefore if the speeds of all vehicles are initially the same, the relative velocity at the time of collision is going to be small. Here we attempt to establish conditions under which this conjecture is true.

It is assumed that the emergency deceleration of vehicle 0 is caused by some abnormal condition, such as a mechanical malfunction (e.g. a brakes-on failure) or an obstacle (e.g. debris spilling over from an accident in an adjacent lane). The emergency deceleration maneuver is an example of an emergency maneuver; other examples include emergency lane change, emergency splitting of platoons, etc. The reader is referred to [12] for a more detailed discussion of emergency maneuvers and their initiation. Even though specialized controllers have been designed for some emergency maneuvers [13, 14, 15], none of the results available in the literature are sufficient to guarantee safety under such extreme conditions. We view our analysis of the emergency deceleration maneuver as a first step in this direction.

4.2 Default Deceleration Strategy

To construct strings that undergo emergency deceleration we need to fix the values of all initial conditions and parameters and to specify automata for all controllers and sensors. The following definitions that can be used to cut down on the number of situations that need to be considered:

Definition 3 *A string is* **initially at steady state** *if for all $i, j = 0, \ldots, N-1$, $v_i(0) = v$ for some $v \geq 0$, the internal variable of the actuation delay buffers satisfies $b^A_i(0) \equiv 0$ and the internal variable of the delay buffers $b^S_{ji}(0) \equiv y^p_i(0)$.*

The emergency deceleration maneuver calls for the first vehicle of the string to apply maximum deceleration until it comes to a stop. This behavior can be implemented in the string model if we let $d^S_{00} = d^A_0 = 0$ and define the sensor and controller of vehicle 0 by the maps:

$$y^s_0 \quad = \quad h_0(\hat{y}^p) = v_0 \tag{16}$$

$$u_0 \quad = \quad g_0(y_0^s) = \begin{cases} 0 & \text{if } (y_0^s = 0) \\ a_0^{min} & \text{if } (y_0^s > 0) \\ a_0^{max} & \text{if } (y_0^s < 0) \end{cases} \tag{17}$$

The lack of delays implies that there are no delay buffers to be initialized for vehicle 0.

The leading vehicle starts decelerating at time $t = 0$. Assume that it immediately notifies the following vehicles of its action. The following vehicles receive the notification after some communication delay, possibly dependent on their position in the string (modeled here by d_{i0}^S). How should they respond to this action of the leader? The simplest response would be for each vehicle to start decelerating as hard as possible as soon as it figures out there is an emergency until it comes to a stop. We refer to this strategy as the *default deceleration strategy*. The default deceleration strategy can be implemented in the string model if we let $d_{ii}^S = d_i^A = 0$ for all $i = 1, \ldots, N-1$ and define the sensor and controller of vehicle i by the functions:

$$y_i^s(t) \quad = \quad h_i(\tilde{y}^p(t)) = \begin{bmatrix} u_0(t - d_{i0}^S) \\ \Delta x_i(t) \\ v_i(t) \end{bmatrix} = \begin{bmatrix} y_{i_1}^s(t) \\ y_{i2}^s(t) \\ y_{i3}^s(t) \end{bmatrix} \tag{18}$$

$$u_i \quad = \quad g_i(y_i^s) = \begin{cases} 0 & \text{if } (y_{i_1}^s = 0) \vee (y_{i_3}^s = 0) \\ a_i^{min} & \text{if } (y_{i_1}^s \neq 0) \wedge (y_{i_3}^s > 0) \\ a_i^{max} & \text{if } (y_{i_1}^s \neq 0) \wedge (y_{i_3}^s < 0) \end{cases} \tag{19}$$

Under the default deceleration strategy there is only one delay associated with each $i = 1, \ldots, N-1$, namely d_{i0}^S. To simplify the notation we use d_i to denote this delay.

If the string is initially at steady state, equations (16)–(19) provide a partial specification. The string is still parameterized by $5N - 2$ real parameters ($N - 1$ for each of $\Delta x_i(0)$, d_i and a_i^{max}[6], N for each of a_i^{min} and M_i and 1 for v) and the $N - 1$ real valued restitution functions α_i. In subsequent sections we attempt to establish conditions on these parameters under which the string is safe with the default deceleration strategy.

We can reduce the number of parameters that need to be considered by making additional assumptions. A string initially at steady state satisfies the *uniform spacing* assumption if for all $i = 1, \ldots N-1$, $\Delta x_i(0) = F$ for some $F > 0$. The uniform spacing assumption reduces the number of parameters that need to be considered by $N - 2$. Note that the default deceleration strategy makes use of a_i^{max} only if a vehicle starts going backwards as a result of a collision. We say that the default deceleration strategy is *brakes only* if $a_i^{max} = -a_i^{min}$ for all $i = 0, \ldots, N-1$. The brakes only assumption can be interpreted as saying that even when going backwards a vehicle will use its brakes rather than its engine to stop (which in this case involves accelerating). The brakes only assumption cuts down the number of parameters by $N - 1$. To simplify the notation we will use a_i to denote a_i^{min} whenever the brakes only assumption is in effect.

The system description can be further simplified if we assume that a particular communication architecture is used to transmit the information about u_0 among the vehicles (we assume that x_i is sensed by each vehicle i for local use only). One possible choice is *hop-by-hop communication*, where the information is passed from one vehicle to the next. In this case the delay d_i increases linearly along the string, i.e. $d_i = id$ for some $d \geq 0$. Another possible architecture is *broadcast communication* where the information is transmitted by the leading vehicle and received simultaneously by vehicles $1, \ldots, N-1$. In this case the delay is $d_i = d$ for some $d \geq 0$ and $i = 1, \ldots, N-1$ ($d_0 = 0$). For either architecture the number of parameters is reduced by $N - 2$.

[6] In the next section it will be shown that for the emergency deceleration maneuver $v_0(0) \geq 0$ implies $v_0(t) \geq 0$ for all $t \geq 0$.

4.3 Limits of Safety and Problems of Interest

To motivate the problems that will be addressed in this paper we first derive some rough bounds on the level of safety that can be expected under the default deceleration strategy. It is easy to show the following:

Lemma 4 *Assume $d_i = 0$ and $a_i^{min} \geq a_j^{min}$ for all $0 \leq i \leq j \leq N-1$. Then any string (choice for the remaining parameters) initially at steady state is safe under the default deceleration strategy for any $v_A \geq 0$.*

Proof: We show that under the lemma assumptions no collisions are possible; then the string is trivially safe for all $v_A \geq 0$. We claim that the property:

$$P_{\text{trivial}} = [(v_j - v_i \leq 0) \text{ for all } 0 \leq i \leq j \leq N-1]$$

is an invariant property of the string under the lemma assumptions. The string is initially at steady state, therefore $v_i = v_j = v$ at $t = 0$ and the initial states satisfy P_{trivial}.

Assume P_{trivial} is satisfied at a given state. Then $(v_i - v_{i-1} \leq 0)$ for all $1 \leq i \leq N-1$ and the precondition for action $Collision_i$ can not be satisfied. Actions $Touch_i$ and $Separate_i$ may take place for some i, however both leave v_j unaffected for all j, therefore P_{trivial} will again be satisfied at the post-state.

For the continuous evolution, consider $i \leq j$. Along a trajectory:

$$\frac{d}{dt}(v_j - v_i) = acc_j - acc_i$$

Assume j is part of a segment of touching vehicles S_j and i is part of a segment of touching vehicles S_i. Then, by Part 1 of Lemma 2:

$$acc_j \leq \max_{k \in S_j} a_k^{min} = a_{\min(S_j)}^{min}$$
$$acc_i \leq \min_{k \in S_i} a_k^{min} = a_{\max(S_i)}^{min}$$

If $\min(S_j) \leq \max(S_i)$ then i and j are part of the same segment and $acc_j = acc_i$. If $\min(S_j) > \max(S_i)$ then $acc_j \leq acc_i$. In either case, $\frac{d}{dt}(v_j - v_i) \leq 0$. Therefore, if P_{trivial} is satisfied at the first state of a trajectory it will also be satisfied at the last state.

Overall P_{trivial} is an invariant property for a string satisfying the lemma assumptions. Recall that P_{trivial} implies $v_i - v_{i-1} \leq 0$ and therefore the string is safe. ■

As there are no collisions in this case, the parameters a_i^{max}, M_i and the functions α_i do not enter the picture. Lemma 4 indicates that if there are no differences in deceleration capabilities and no delays the safety question is trivial. We can relax the assumptions of the lemma by allowing certain system parameters to lie in ranges. Assume that the brakes only assumption is in effect and consider the case where:

$$a_i^{min} \in [\underline{a}, \overline{a}] \quad (20)$$
$$d_i \in [\underline{d}_i, \overline{d}_i] \quad (21)$$
$$M_i \in [\underline{M}, \overline{M}] \quad (22)$$

The following provides a limit of what can be expected in this case:

Lemma 5 *Consider a string, initially at steady state, satisfying the uniform spacing assumption. Set* $F = 1m$, $v = 25ms^{-1}$ *and* $v_A = 3ms^{-1}$, $\alpha_i \equiv 1$ *and assume that the parameter values are bounded by* $\underline{a} = -9.32ms^{-2}, \overline{a} = -4.41ms^{-2}$ *and* $\underline{M} = \overline{M} = 1500Kg$. *Finally, assume that either* $\underline{d}_i = \overline{d}_i = d$ *for all* $i > 0$ *or* $\underline{d}_i = \overline{d}_i = id$ *for all* $i \geq 0$ *and let let* $d = 0.05s$. *Then there exists a string satisfying (20)–(22) which is unsafe under the default deceleration strategy.*

Proof: By numerical examples, see [16]. ∎

All the parameter values in Lemma 5 are realistic in terms of current technology. The conditions of the lemma seem very specific; however the same conclusion has been shown to hold for a wide range of cases. For example, the conclusion of the lemma trivially holds of any $\underline{d}_i, \underline{M}$ and $\underline{a}$ less than the quoted values and any $\overline{d}_i, \overline{M}$ and $\overline{a}$ greater than the quoted values. In the numerical experiments of [16] a number of alternatives were also considered: the range $[\underline{a}, \overline{a}]$ was reduced, d, v and F were varied and realistic, monotone decreasing functions were used for α_i. The conclusions were similar in all cases.

These limitations suggest a number of problems that can be addressed in this setting. We list a few below. All problems are parameterized by $\Delta x_i(0)$ and v. For simplicity we assume that in all cases except Problem 1 the brakes only assumption is in effect.

Problem 1: Establish conditions on $\underline{a}, \overline{a}, \underline{d}_i$ and $\overline{d}_i$ so that no collisions are possible under the default deceleration strategy in a string satisfying (20) and (21).

Problem 2: Establish conditions on $\underline{a}, \overline{a}, \underline{d}_i, \overline{d}_i, \underline{M}, \overline{M}$ and α_i so that, under the default deceleration strategy, any string satisfying (20)–(22) is safe.

Problem 3: Establish conditions on the same parameters so that, under the default deceleration strategy, there exists a string satisfying (20)–(22) which is unsafe.

Problem 4: Establish conditions on the same parameters such that there exists a deceleration strategy that for which any string satisfying (20)–(22) is safe.

Problem 5: Establish conditions on the same parameters so that, under any deceleration strategy, there exists a string satisfying (20)–(22) which is unsafe.

Problem 1 is relatively easy. It can be approached by considering only pairs of adjacent vehicles. The conditions can be inferred from calculations already available in the literature (as well as the calculations presented in this paper). M_i and α_i do not appear in the statement of Problem 1, as the objective is to avoid collisions altogether.

Problems 2 and 3 are more challenging and are the topic of this paper. The difficulty is that a collision between vehicle i and $i-1$ and the resulting change in velocity "couple" the dynamics of vehicle $i+1$ not only with those of i but also with those of $i-1$. Therefore the conditions of problems 2 and 3 will have to involve more than just adjacent vehicle pairs.

Problems 4 and 5 are substantially more difficult and will be the topic of future research. Problem 4 may be approached by solving a (very complicated) optimal control problem. Solving Problem 4 in this way will automatically provide a solution to Problem 5. Alternatively, Problem 5 can be addressed using techniques for proving impossibility results for distributed algorithms [17]. For both problems, important assumptions will have to be made about the information available to each vehicle; does vehicle i have access to the state of all other vehicles, does it have access to the bound on its deceleration, a_i^{min}, does it have access to the bounds for other vehicles, etc.

5 Safety of Strings of Length $N = 2$

We first develop necessary and sufficient conditions for a string of two vehicles to be safe under the default deceleration strategy. We refer to such a string as a *pair*. These conditions will form the basis of safety results for longer strings.

5.1 Basic Properties

Throughput this section we assume that $d_1 = 0$. Then, under the default deceleration strategy the commanded acceleration of vehicle $i = 1, 2$ can be written as a function of the vehicle state:

$$u_i = \begin{cases} a_i^{min} & \text{if } v_i > 0 \\ 0 & \text{if } v_i = 0 \\ a_i^{max} & \text{if } v_i < 0 \end{cases}$$

Proposition 9 $(v_0 \geq 0)$ *is an invariant property of the pair.*

Proof: By Assumption 2 $v_0(0) \geq 0$. If $(v_0 \geq 0)$ when $Collision_1$ occurs then, by equation (3), $v_0' \geq v_0 \geq 0$. Moreover, $Touch_1$ and $Separate_1$ do not affect v_0. Therefore $(v_0 \geq 0)$ is preserved by all the actions.

For the continuous evolution, assume $v_0 = 0$ at the first state of a trajectory. Recall that $\dot{v}_0 = acc_0$. If $Touching_1 =$ False, $acc_0 = u_0 = 0$ under the default deceleration strategy. If $Touching_1 =$ True, $acc_0 \geq \min\{u_0, u_1\}$. If $v_1 > 0$ the action $Collision_1$ takes place and the trajectory stops. If $v_1 < 0$, the action $mboxSeparate_1$ takes place and the trajectory stops. If $v_1 = 0$, $u_1 = 0$, therefore $acc_0 = 0$. ■

Proposition 10 $(v_1 \leq 0)$ *is a stable property of the pair.*

Proof: Assume $v_1 \leq 0$ when $Collision_1$ occurs. Then, by equation (3), $v_1' \leq v_1 \leq 0$. Moreover, $Touch_1$ and $Separate_1$ do not affect v_1. Therefore $(v_1 \leq 0)$ is preserved by all the actions.

For the continuous evolution, assume $v_1 = 0$ at the first state of a trajectory. If $Touching_1 =$ False, then $acc_1 = u_1 = 0$ under the default deceleration strategy. If $Touching_1 =$ True, $acc_1 \leq \max\{u_0, u_1\}$. $v_0 \geq 0$ by Proposition 9. If $v_0 > 0$, the action the action $mboxSeparate_1$ takes place and the trajectory stops. If $v_0 = 0$, $u_0 = 0$, therefore $acc_1 = 0$. ■

Proposition 11 *If* $(v_1 \leq 0)$ *the pair is safe (in particular* $Collision_1$ *cannot occur).*

Proof: If $(v_1 \leq 0)$ then $v_1 \leq 0 \leq v_0$ (by Proposition 9). Therefore, $v_1 \leq v_0 + v_A$ and the pair is safe. The precondition of $Collision_1$ will never be satisfied. ■

To derive more meaningful safety conditions consider the derived variables:

$$\begin{aligned} C_1, C_2, P_1, P_2 &: \mathbb{R}^3 \longrightarrow \mathbb{R} \\ C_1(\Delta x_1, v_1, v_0) &= (a_1 + a_0)v_0^2 - 2a_0 v_0 v_1 - 2a_0^2 \Delta x_1 \quad (23) \\ C_2(\Delta x_1, v_0, v_1) &= \frac{a_1}{a_0} v_0 - v_1 \quad (24) \\ P_1(\Delta x_1, v_0, v_1) &= (v_0 - v_1)^2 - 2(a_0 - a_1)\Delta x_1 - v_A^2 \quad (25) \\ P_2(\Delta x_1, v_0, v_1) &= v_1^2 - \frac{a_1}{a_0} v_0^2 + 2a_1 \Delta x_1 - v_A^2 \quad (26) \end{aligned}$$

Proposition 12 $(C_1(\Delta x_1, v_1, v_0) > 0) \Rightarrow v_0 > 0$

Proof: By Proposition 9, $v_0 \geq 0$. Moreover, $v_0 = 0$ implies that $C_1(\Delta x_1, v_1, v_0) = -2a_0^2 \Delta x_1 \leq 0$. ■

To simplify the notation we will explicitly mention the function arguments only when necessary. We also introduce a derived boolean variable C given by the expression:

$$C = [(C_1 \leq 0) \wedge (a_0 \leq a_1)] \vee [(C_2 \leq 0) \wedge (a_0 \geq a_1)] \vee [(v_0 = 0)] \tag{27}$$

P_1, P_2 and C will be used to construct invariants for the pair to encode safety conditions. A collision can take place either while both vehicles are moving or while while vehicle 1 is moving and vehicle 0 has stopped (by Proposition 11 a collision cannot take place once vehicle 1 stops). The property $(P_1 \leq 0)$ will encode conditions that guarantee safety if a collision takes place while both vehicles are still moving. $(P_2 \leq 0)$ will encode conditions that guarantee that either no collision takes place or a safe collision takes place after vehicle 0 has stopped. The predicate C will be used to distinguish the two cases.

5.2 Sufficient Conditions for Safety

Lemma 6 $(P_1 \leq 0) \vee (v_1 \leq 0)$ *is a stable property of the pair.*

Proof: Assume $(P_1 \leq 0) \vee (v_1 \leq 0)$ is true when $Collision_1$ occurs. By Proposition 11 $(v_1 \leq 0)$ can not be true in this case. Assume $(P_1 \leq 0)$ is true. Then, $P_1(\Delta x_1, v_0, v_1) = P_1(0, v_0, v_1) \leq 0$. Hence, by the restitution equation (3), $(v_0' - v_1')^2 = (v_0 - v_1)^2 \alpha_1^2 \leq (v_0 - v_1)^2 \leq v_A^2$, as $\alpha_1 \in [0, 1]$ by Assumption 1. Therefore, $P_1(\Delta x_1', v_0', v_1') = P_1(0, v_0', v_1') \leq 0$ and $(P_1 \leq 0) \vee (v_1 \leq 0)$ is again true after $Collision_1$. Moreover, $(P_1 \leq 0) \vee (v_1 \leq 0)$ is preserved by $Touch_1$ and $Separate_1$, as both these actions leave $\Delta x_1, v_0$ and v_1 unaffected.

Assume at some state, s, $(P_1 \leq 0) \vee (v_1 \leq 0)$ is true and consider all trajectories that start at s. If $(v_1 \leq 0)$ is true at s it will also be true at the last state of the trajectory by Proposition 10. If $(P_1 \leq 0) \wedge (v_1 > 0)$ is true at s, consider the variation of P_1 along a trajectory:

$$\begin{aligned} \frac{d}{dt} P_1 &= 2(v_0 - v_1)(acc_0 - acc_1) - 2(a_0 - a_1)(v_0 - v_1) \\ &= \begin{cases} 0 & \text{if } (v_0 > 0) \wedge (v_1 > 0) \wedge \neg Touching_1 \\ 2a_0 v_1 & \text{if } (v_0 = 0) \wedge (v_1 > 0) \wedge \neg Touching_1 \\ -2(a_0 - a_1)(v_0 - v_1) & \text{if } Touching_1 \end{cases} \end{aligned}$$

In the cases where $Touching_1 =$ False, $\dot{P}_1 \leq 0$, therefore $(P_1 \leq 0)$ will be true at least until $(v_1 \leq 0)$ becomes true. If $Touching_1 =$ True and $v_0 < v_1$ (resp. $v_0 > v_1$) action $Collision_1$ (resp. $Separate_1$) occurs and the trajectory stops. If $Touching_1 =$ True and $v_0 = v_1$, then $\dot{P}_1 = 0$. Overall, $(P_1 \leq 0) \vee (v_1 \leq 0)$ will be true at the last state of the trajectory. ■

Lemma 7 *If* $(P_1 \leq 0) \vee (v_1 \leq 0)$ *is true then the pair is safe.*

Proof: If $(v_1 \leq 0)$ is true the pair is safe by Proposition 11. If $(P_1 \leq 0)$, at the time when $\Delta x_1 = 0$, $P_1(\Delta x_1, v_0, v_1) = P_1(0, v_0, v_1) \leq 0$, therefore $(v_0 - v_1)^2 \leq v_A^2$. Hence, $v_1 \leq v_0 + v_A$ and the pair is safe. ■

Lemma 7 provides a sufficient condition for a pair of vehicles to be safe. We now seek situations that violate the condition of the lemma and yet are safe. Consider:

$$I = [P_1 \leq 0] \vee [C \wedge (P_2 \leq 0)] \tag{28}$$

Lemma 8 *$I \vee (v_1 \leq 0)$ is a stable property of the pair.*

Proof: If $(P_1 \leq 0) \vee [C \wedge (P_2 \leq 0)] \vee (v_1 \leq 0)$ is true at the pre-state of $Touch_1$ or $Separate_1$ it will also be true at the post-state as both actions leave $\Delta x_1, v_0$ and v_1 unaffected. Assume $(P_1 \leq 0) \vee [C \wedge (P_2 \leq 0)] \vee (v_1 \leq 0)$ is true when $Collision_1$ occurs. If $(P_1 \leq 0) \vee (v_1 \leq 0)$ is true, it will also be true after $Collision_1$ by Lemma 6. Assume $Collision_1$ occurs while $C \wedge (P_2 \leq 0)$ is true. We distinguish the following cases:

Case 1: $(v_0 = 0) \wedge (P_2 \leq 0)$ is true. Then, at $\Delta x_1 = 0$,

$$(v_0 = 0) \wedge (P_2 \leq 0) \Rightarrow v_1^2 - v_A^2 \leq 0 \Rightarrow v_1 = v_1 - v_0 \leq v_A$$

Case 2: $(C_1 \leq 0) \wedge (a_0 \leq a_1) \wedge (P_2 \leq 0)$ is true. Then, at $\Delta x_1 = 0$,

$$\begin{aligned} (C_1 \leq 0) \wedge (a_0 \leq a_1 < 0) &\Rightarrow ((a_0 + a_1)v_0^2 - 2a_0 v_0 v_1 \leq 0) \wedge (a_0 \leq a_1 < 0) \\ &\Rightarrow (\frac{a_0 + a_1}{2a_0} v_0 \geq v_1) \wedge (0 < \frac{a_0 + a_1}{2a_0 -} \leq 1) \end{aligned}$$

Therefore, $v_0 \geq v_1$ and this hence $(C_1 \leq 0) \wedge (a_0 \leq a_1) \wedge (P_2 \leq 0)$ cannot be true when $Collision_1$ occurs.

Case 3: $(C_2 \leq 0) \wedge (a_0 \geq a_1) \wedge (P_2 \leq 0)$ is true. Then, at $\Delta x_1 = 0$,

$$\begin{aligned} (C_2 \leq 0) \wedge (a_0 \geq a_1) \wedge (P_2 \leq 0) &\Rightarrow (\frac{a_1}{a_0} v_0 \leq v_1) \wedge (\frac{a_1}{a_0} \geq 1) \wedge (v_1^2 - \frac{a_1}{a_0} v_0^2 - v_A^2 \leq 0) \\ &\Rightarrow (\frac{a_1}{a_0} \geq 1) \wedge \left((v_0 - v_1)^2 - v_A^2 - v_0^2 + \frac{a_1}{a_0} v_0^2 \leq 0 \right) \\ &\Rightarrow (v_0 - v_1)^2 - v_A^2 \leq 0 \end{aligned}$$

In all cases $0 < v_1 - v_0 \leq v_A$. Therefore $(v_0 - v_1)^2 \leq v_A^2$ and hence $(v_0' - v_1')^2 \leq v_A^2$ (by equation (3) and Assumption 1). Therefore, if $Collision_1$ occurs while $C \wedge (P_2 \leq 0)$ is true, $(P_1 \leq 0)$ will be true after the collision. Overall, if $(P_1 \leq 0) \vee [C \wedge (P_2 \leq 0)] \vee (v_1 \leq 0)$ is true when $Collision_1$ occurs it will also be true afterwards.

Assume at some state, s, $(P_1 \leq 0) \vee [C \wedge (P_2 \leq 0)] \vee (v_1 \leq 0)$ is true and consider the trajectories that start at this state. If $(P_1 \leq 0) \vee (v_1 \leq 0)$ is true at s it will also be true at the last state of the trajectory, by Lemma 6. If $C \wedge (P_2 \leq 0) \wedge (v_1 > 0)$ is true at s, consider the derivatives of the functions C_1, C_2 and P_2 along the trajectory:

$$\begin{aligned} \frac{d}{dt} C_1 &= 2(a_0 + a_1) v_0 \, acc_0 - 2a_0 \, acc_0 v_1 - 2a_0 v_0 \, acc_1 - 2a_0^2 (v_0 - v_1) \\ &= \begin{cases} 0 & \text{if } (v_0 > 0) \wedge \neg Touching_1 \\ 2a_0^2 v_1 & \text{if } (v_0 = 0) \wedge \neg Touching_1 \\ 2(a_1 v_0 - a_0 v_1) acc_0 - 2a_0^2 (v_0 - v_1) & \text{if } Touching_1 \end{cases} \\ \frac{d}{dt} C_2 &= \frac{a_1}{a_0} acc_0 - acc_1 \\ &= \begin{cases} 0 & \text{if } (v_0 > 0) \wedge \neg Touching_1 \\ -a_1 & \text{if } (v_0 = 0) \wedge \neg Touching_1 \\ \left(\frac{a_1}{a_0} - 1 \right) acc_0 & \text{if } Touching_1 \end{cases} \end{aligned}$$

$$\begin{aligned}\frac{d}{dt}P_2 &= 2v_1 acc_1 - 2\frac{a_1}{a_0}v_0 acc_0 + 2a_1(v_0 - v_1) \\ &= \begin{cases} 0 & \text{if } \neg Touching_1 \\ 2\frac{a_0 v_1 - a_1 v_0}{a_0} acc_0 + 2a_1(v_0 - v_1) & \text{if } Touching_1 \end{cases}\end{aligned}$$

Consider first the variation of P_2. If $Touching_1 =$ False and as long as $v_1 > 0$, $\dot{P}_2 = 0$. Therefore, if $(P_2 \leq 0)$ is true at s, $(P_2 \leq 0) \vee (v_1 \leq 0)$ will be true at the last state of the trajectory. If $Touching_1 =$ True and $v_1 \neq v_0$ the trajectory stops (as the precondition of either $Collision_1$ or $Separate_1$ is satisfied). If $Touching_1 =$ True and $v_1 = v_0$ then $\dot{P}_2 = 2(a_0 - a_1)v_0 acc_0/a_0$. If $a_0 > a_1$ the trajectory stops and action $Separate_1$ occurs. Otherwise, $\dot{P}_2 \leq 0$, therefore $(P_2 \leq 0)$ will be true at the last state of the trajectory.

Now consider the variation of C. Recall that $C \wedge (v_1 > 0)$ is assumed to be true at s. Distinguish two cases:

Case 1: $(C_1 \leq 0) \wedge (a_0 \leq a_1)$ is true at s. If $Touching_1 =$ False and as long as $v_1 > 0$ and $v_0 > 0$, $\dot{C}_1 = 0$. If $Touching_1 =$ True and $v_1 \neq v_0$ the trajectory stops (as the precondition of either $Collision_1$ or $Separate_1$ is satisfied). If $Touching_1 =$ True and $v_1 = v_0$ then $\dot{C}_1 = 2(a_1 - a_0)v_0 acc_0 \leq 0$ as $a_0 \leq a_1$. Overall, $[(C_1 \leq 0) \wedge (a_0 \leq a_1)] \vee (v_0 = 0) \vee (v_1 \leq 0)$ will be true at the final state of the trajectory.

Case 2: $(C_2 \leq 0) \wedge (a_0 \geq a_1)$ is true at s. If $Touching_1 =$ False and as long as $v_1 > 0$ and $v_0 > 0$, $\dot{C}_1 = 0$. If $Touching_1 =$ True and $v_1 \neq v_0$ the trajectory stops (as the precondition of either $Collision_1$ or $Separate_1$ is satisfied). If $Touching_1 =$ True and $v_1 = v_0$ then $\dot{C}_2 = (a_1 - a_0)acc_0/a_0 \leq 0$, as $a_0 \geq a_1$. Therefore, $[(C_2 \leq 0) \wedge (a_0 \geq a_1)] \vee (v_0 = 0) \vee (v_1 \leq 0)$ will be true at the final state of the trajectory.

Overall, if $(P_1 \leq 0) \vee [C \wedge (P_2 \leq 0)] \vee (v_1 \leq 0)$ is true at the first state of a trajectory, it will also be true at the last state. ■

Theorem 2 (Sufficient Condition for Pair Safety) *If I is initially true the pair is safe.*

Proof: I initially true and Lemma 8 imply $[P_1 \leq 0] \vee [C \wedge (P_2 \leq 0)] \vee (v_1 \leq 0)$ is an invariant property of the pair. If $(P_1 \leq 0) \vee (v_1 \leq 0)$ is true safety is guaranteed by Lemma 7. If $C \wedge (P_2 \leq 0)$ is true, the proof of Lemma 8 indicates that at $\Delta x_1 = 0$, $v_1 - v_0 \leq v_A$, which again implies safety. ■

5.3 Necessary Conditions for Safety

Conditions under which the string is unsafe can be obtained in a similar way. The proof of Theorem 2 indicates that if a collision is safe, all subsequent collisions will also be safe. Our conditions must therefore be such that the *first collision* is unsafe; more unsafe collisions may follow. Consider a derived boolean variable *Collided* which is initially false and becomes true when the actions $Collision_1$ occurs. Let:

$$C' = (C_1 \leq 0) \tag{29}$$

$$I' = [\neg C' \wedge (P_1 > 0)] \vee [(C' \vee (v_0 = 0)) \wedge (P_2 > 0)] \tag{30}$$

Lemma 9 $I' \vee (v_1 \leq 0) \vee$ Collided *is a stable property of the pair.*

Proof: Assume $I' \vee (v_1 \leq 0) \vee$ *Collided* is true at some state. Consider all trajectories that start at that state. If *Collided* is true at the start state of such a trajectory, it will trivially be true at the last state also. If $(v_1 \leq 0)$ is true at the start state it will also be true at the last state, by Proposition 10. Assume I' is true at the start state. We show I' remains true until $v_1 \leq 0$. We distinguish the following cases:
Case 1: $\neg C' \wedge (P_1 > 0)$ is true. $\neg C'$ implies that $v_i > 0$ by Proposition 12, hence from the proof of Lemma 6, $\dot{C}_1 = 0$ and $\dot{P}_1 = 0$. Therefore, if $\neg C' \wedge (P_1 > 0)$ is true at the start state of a trajectory it will be true at least until $v_1 \leq 0$.
Case 2: By a similar argument and using the calculations of Lemma 8 if $(C' \vee (v_i = 0)) \wedge (P_2 > 0)$ is true at the start state of a trajectory, it will continue to be true at least until $v_1 \leq 0$.

Overall, if $I' \vee (v_1 \leq 0) \vee$ *Collided* is true at the first state of a trajectory it will also be true at the last state.

Assume $I' \vee (v_1 \leq 0) \vee$ *Collided* is true when $Collision_1$ occurs. After the collision *Collided* and hence $I' \vee (v_1 \leq 0) \vee$ *Collided*, will be true. ■

Theorem 3 (Necessary Condition for Pair Safety) *If* $I' \wedge (v_1 > 0) \wedge \neg$Collided *is true initially then the pair is unsafe.*

Proof: By Lemma 9, if $I' \wedge (v_1 > 0) \wedge \neg$ *Collided* is true at the initial states, $I' \vee (v_1 \leq 0) \vee$ *Collided* is an invariant property of the pair. Therefore, I' will remain true at least until either $(v_1 \leq 0)$ or *Collided* become true. We show that *Collided* becomes true before $(v_1 \leq 0)$. First note that if $\Delta x_1 = 0$ while I' is true then $(v_0 - v_1)^2 > v_A^2$. To see this consider the following cases:
Case 1: $\neg C' \wedge (P_1 > 0)$ is true. Then, at $\Delta x_1 = 0$, $P_1 > 0 \Rightarrow (v_0 - v_1)^2 - v_A^2 > 0$.
Case 2: $(C' \vee (v_0 = 0)) \wedge (P_2 > 0)$ is true.
Case 2.1: $(v_0 = 0) \wedge (P_2 > 0)$ is true. Then, at $\Delta x_1 = 0$,

$$(v_0 = 0) \wedge (P_2 > 0) \Rightarrow v_1^2 - v_A^2 > 0 \Rightarrow v_1 = v_1 - v_0 > v_A$$

Case 2.2: $(C_1 \leq 0) \wedge (P_2 > 0)$ is true. Then, at $\Delta x_1 = 0$,

$$\begin{aligned}(C_1 \leq 0) \wedge (P_2 > 0) &\Rightarrow ((a_0 + a_1)v_0^2 - 2a_0 v_0 v_1 \leq 0) \wedge (v_1^2 - \frac{a_1}{a_0}v_0^2 - v_A^2 > 0) \\ &\Rightarrow a_0(v_0^2 + v_1^2 - 2v_0 v_1) - a_0 v_A^2 < 0 \\ &\Rightarrow (v_0 - v_1)^2 - v_A^2 > 0\end{aligned}$$

Note that sooner or later $v_1 = 0$ will be reached. Here v_1 plays the role of a progress variable. Assume, for the sake of contradiction, that v_1 becomes 0 while I' is true, before *Collided* becomes true. Consider again cases:
Case 1: $\neg C' \wedge (P_1 > 0)$ is true.

$$\begin{aligned}(C_1 > 0) \wedge (P_1 > 0) &\Rightarrow ((a_0 + a_1)v_0^2 - 2a_0 v_0 v_1 \geq 0) \wedge ((v_0 - v_1)^2 - 2(a_0 - a_1)\Delta x_1 - v_A^2 > 0) \\ &\Rightarrow v_1^2 - \frac{a_1}{a_0}v_0^2 + 2a_1 \Delta x_1 - v_A^2 > 0\end{aligned}$$

Note that $\Delta x_1 < 0$ is needed to satisfy the above expression at $v_1 = 0$. Therefore, as $v_1 \to 0$, Δx_1 must cross 0 from above. This implies that at $\Delta x_1 = 0$, $d\Delta x_1/dt = v_0 - v_1 \leq 0$. The relative velocity calculation given above guarantees that at $\Delta x_1 = 0$, $|v_1 - v_0| > v_A$, hence $v_0 - v_1$ is bounded away from 0. Therefore a collision does happen ($\Delta x_1 = 0$ and $v_1 > v_0$), with relative velocity greater than v_A.

Case 2: $(C' \vee (v_i = 0)) \wedge (P_2 > 0)$ is true. Then $(P_2 > 0) \Rightarrow v_1^2 - \frac{a_1}{a_0} v_0^2 + 2a_1 \Delta x_1 - v_A^2 > 0$. The claim follows by the same argument given above.

Overall, the above calculation indicates that if $I' \wedge (v_1 > 0) \wedge \neg \mathit{Collided}$ is true $\mathit{Collision}_1$ will occur. Moreover, at the time when $\Delta x_1 = 0$, $v_1 > v_A + v_0$. Therefore there exist reachable states where the property $[\Delta x_i = 0 \Rightarrow v_1 \leq v_0 + v_A]$ is violated and the pair is unsafe. ■

In subsequent proofs the following corollary will also be useful.

Corollary 1 *If* $(P_1 > 0) \wedge (P_2 > 0) \wedge (v_1 > 0)$ *initially then the pair is unsafe.*

Proof: As $C' \vee [\neg C' \vee (v_i = 0)]$ is true, $(P_1 > 0) \wedge (P_2 > 0) \Rightarrow I'$. The conclusion follows from Theorem 3. ■

6 Safety of Strings of Length $N > 2$

6.1 Sufficient Conditions

Next, we derive a very simple sufficient condition for a string of arbitrary length to be safe. Even though the condition is conservative, interesting conclusions about the safety of platoons of vehicles can be derived from it (see Section 7). Unless otherwise stated we assume that $\underline{d}_i = \overline{d}_i = 0$.

Definition 4 *A string undergoing emergency deceleration under the default deceleration strategy is* **near uniform mass** *if* $\alpha_i(v) \equiv \alpha$ *is constant and* $\alpha M_{k-1} \leq M_k \leq M_{k-1}/\alpha$.

A near uniform mass string is such that the masses of all its vehicles are close to one another. This allows us to put some bounds on the change of speed that a collision can induce. For example, it can be shown that the vehicles of a near uniform mass string will never go backwards.

Proposition 13 $\bigwedge_{i=0}^{N-1}(v_i \geq 0)$ *is an invariant property of a near uniform mass string.*

Proof: Let $Q = \bigwedge_{i=0}^{N-1}(v_i \geq 0)$. By Assumption 2, $v_i(0) \geq 0$ for all i, therefore the initial states satisfy property Q. Assume Q is satisfied at some state. Consider all trajectories that start at that state. Consider an arbitrary vehicle i and assume that it is part of a segment of touching vehicles S_i. If there exists $j \in S_i$ with $v_j \neq v_i$ the precondition of at least one $\mathit{Collision}_k$ or Separate_k action with $k \in S_i$ will be satified and the trajectory will stop. If $v_j = v_i = 0$, $acc_i \geq \min_{j \in S_i}(u_j) = 0$ under the default deceleration strategy. Therefore, if Q is true at the first state of a trajectory is will also be true at the last state.

Q is trivially preserved by Touch_j and $\mathit{Separate}_j$ for all $j > 0$, as these actions do not affect the v_i's. Assume Q is true when $\mathit{Collision}_j$ occurs for some $j > 0$. Let v_i' denote the velocity of vehicle i after $\mathit{Collision}_j$. If $i \notin \{j, j-1\}$, $v_i' = v_i$, therefore $(v_i \geq 0)$ implies $(v_i' \geq 0)$. If $i = j - 1$, $v_i' \geq v_i$ by the restitution equation (3), therefore $(v_i \geq 0)$ implies $(v_i' \geq 0)$. Finally, if $i = j$, solving the conservation of momentum and restitution equations (2) and (3) leads to:

$$v_i' = \frac{M_{i-1}(1+\alpha)v_{i-1} + (M_i - M_{i-1}\alpha)v_i}{M_i + M_{i-1}} \geq \frac{M_{i-1}(1+\alpha)v_{i-1}}{M_i + M_{i-1}}$$

Therefore, $(v_{i-1} \geq 0)$ implies $(v_i' \geq 0)$. Overall, for any $j > 0$, if Q is true when $\mathit{Collision}_j$ occurs, Q will also be true after the collision. ■

We now construct invariant properties that allow us to characterize the safety of such a string. Define Δx_{ij} for $0 \leq i < j \leq N-1$, $\hat{a}_{min}$ and $\hat{a}_{max}$ by:

$$\Delta x_{ij} = \sum_{k=i+1}^{j} \Delta x_k, \quad \hat{a}_{min} = \min_{0 \leq k < N} a_k, \quad \hat{a}_{max} = \max_{0 \leq k < N} a_k$$

For any pair of vehicles $i < j$, consider the function:

$$P(\Delta x_{ij}, v_i, v_j) = v_j - \frac{\hat{a}_{max}}{\hat{a}_{min}} v_i - v_A \tag{31}$$

Proposition 14 $(P(\Delta x_{ij}, v_i, v_j) = 0) \wedge (v_i > 0) \Rightarrow (v_j > 0)$ *for a near uniform mass string.*

Proof: $(P(\Delta x_{ij}, v_i, v_j) = 0)$ implies that $v_j = \frac{\hat{a}_{max}}{\hat{a}_{min}} v_i - v_A$. $\hat{a}_{min} \leq \hat{a}_{max} < 0$ by Assumption 3 and $v_A \geq 0$ by definition, therefore if $v_i > 0$ then $v_j > 0$. ■

Lemma 10 *The property* $\left(\bigwedge_{i=0}^{N-2} \bigwedge_{j=i+1}^{N-1} (P(\Delta x_{ij}, v_i, v_j) \leq 0)\right)$ *is stable for a near uniform mass string.*

Proof: Assume $\bigwedge_{i=0}^{N-2} \bigwedge_{j=i+1}^{N-1} (P(\Delta x_{ij}, v_i, v_j) \leq 0)$ is true at some state. Consider first all trajectories that start at that state. For one of the $P(\Delta x_{ij}, v_i, v_j)$ to become greater than zero along the trajectory, $(P(\Delta x_{ij}, v_i, v_j) = 0)$ must be true first. In this case:

$$\frac{d}{dt} P = \begin{cases} a_j - \frac{\hat{a}_{max}}{\hat{a}_{min}} a_i & \text{if } v_i > 0 \wedge v_j > 0 \\ a_j & \text{if } v_i = 0 \wedge v_j > 0 \\ 0 & \text{if } v_i = 0 \wedge v_j = 0 \end{cases}$$

Recall that, $v_i, v_j \geq 0$ by Proposition 13 and $(v_i > 0) \wedge (v_j = 0)$ can not be true if $P(\Delta x_{ij}, v_i, v_j) = 0$ by Proposition 14. As $\hat{a}_{max} \geq a_j$ and $\hat{a}_{min} \leq a_i$, $\dot{P} \leq 0$ in all cases. Therefore, if $(P(\Delta x_{ij}, v_i, v_j) \leq 0)$ at the first state of a trajectory it will also be true at the last state.

Assume $\bigwedge_{i=0}^{N-2} \bigwedge_{j=i+1}^{N-1} (P(\Delta x_{ij}, v_i, v_j) \leq 0)$ is true when $Collision_l$ occurs. Let v_k' denote the velocity of vehicle k after $Collision_l$. Consider each $P(\Delta x_{ij}, v_i, v_j)$ independently. If $l \notin \{i, i+1, j, j+1\}$ $v_i' = v_i$ and $v_j' = v_j$. Therefore $P(\Delta x_{ij}, v_i, v_j) \leq 0$ implies $P(\Delta x_{ij}, v_i', v_j') \leq 0$. We treat the remaining four cases one at a time:

Case 1: $l = i$. Assume $v_i' \geq v_{i-1}$. Then, $P(\Delta x_{ij}, v_{i-i}, v_j) \leq 0$ implies $P(\Delta x_{ij}, v_i', v_j) \leq 0$.

Case 2: $l = i + 1$. $v_i' \geq v_i$. Therefore, $P(\Delta x_{ij}, v_i, v_j) \leq 0$ implies $P(\Delta x_{ij}, v_i', v_j) \leq 0$.

Case 3: $l = j$. $v_j' \leq v_j$. Therefore $P(\Delta x_{ij}, v_i, v_j) \leq 0$ implies $P(\Delta x_{ij}, v_i, v_j') \leq 0$.

Case 4: $l = j + 1$. Assume $v_j' \leq v_{j+1}$. Then, $P(\Delta x_{ij}, v_i, v_{j+1}) \leq 0$ implies $P(\Delta x_{ij}, v_i, v_j') \leq 0$.

It remains to show that $v_j' \leq v_{j+1}$ in case of $Collision_{j+1}$ and $v_i' \geq v_{i-1}$ in case of $Collision_i$. Solving the conservation of momentum and restitution equations reveals that the first condition is satisfied if $M_j - \alpha M_{j+1} \geq 0$ while the second if $M_i - \alpha M_{i-1} \geq 0$. Both these conditions are met by a near uniform mass string. ■

Theorem 4 (Sufficient Condition for String Safety) *A near uniform mass string of N vehicles is safe if $P(\Delta x_{ij}(0), v_i(0), v_j(0)) \leq 0$ for all i, j with $0 \leq i < j \leq N - 1$.*

Proof: Lemma 10 and the theorem assumptions imply that $(P(\Delta x_{ij}, v_i, v_j) \leq 0)$ is an invariant property of the near uniform mass string. As $\hat{a}_{min} \leq \hat{a}_{max}$, this implies that $v_j - v_i \leq v_A$ and hence the string is safe. ■

The following corollaries follow directly from Theorem 4:

Corollary 2 *Consider a string of N vehicles for which $P(\Delta x_{ij}(0), v_i(0), v_j(0)) \leq 0$ for all i, j with $0 \leq i < j \leq N - 1$. If $\underline{d}_i = \overline{d}_i = d$ and $\underline{M} = \overline{M}$ the string is safe under the default deceleration strategy.*

Corollary 3 *A near uniform mass string, initially at steady state with velocity v ($v_i = v$ for all i), consisting of vehicles satisfying (20)–(22) is safe if* $\left(1 - \frac{\overline{a}}{\underline{a}}\right) v - v_A \leq 0$.

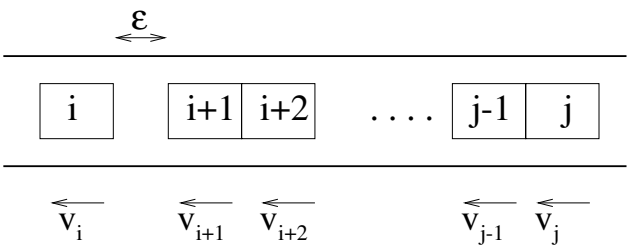

Figure 6: Final configuration for theorem proof

6.2 Necessary Conditions

Now consider a string of N vehicles. We seek necessary conditions such that any platoon formed by a collection of vehicles satisfying (20)-(22) is guaranteed to be safe. Start with the case:

$$\underline{d}_i = \overline{d}_i = 0 \text{ and } \alpha_i(v) \equiv 1 \tag{32}$$

i.e. no delay ($d_i = 0$ for all i) and elastic collisions. Assume that the string is initially moving at steady state with velocity v, i.e.:

$$x_i(0) = \begin{bmatrix} \Delta x_i(0) \\ v \end{bmatrix} \tag{33}$$

Theorem 5 (Necessary Condition for String Safety) *All strings of N vehicles satisfying (20)-(22) and (32)-(33) are guaranteed to be safe under the default deceleration strategy only if $(P_1(\Delta x_{ij}(0), v, v) \leq 0) \vee (P_2(\Delta x_{ij}(0), v, v) \leq 0)$ is true for all i, j with $0 \leq i < j \leq N-1$ and for all $a_i, a_j \in [\underline{a}, \overline{a}]$.*

The definitions of the derived variables P_1 and P_2 are given in equations (25) and (26). The proof is constructive: we show that, if the above condition is violated, one can construct a platoon that satisfies all the theorem conditions and yet, under the default deceleration strategy, exhibits collisions at relative velocities above v_A. The idea of the construction is to bring the vehicles from their initial arrangement to the final arrangement of Figure 6 without any collisions taking place. The construction will be such that after resolving the multiple collision between vehicles $i+1, \ldots, j$ the velocity of vehicle $i+1$ will be the same as the velocity of vehicle j before the collision. For ϵ small enough, the next collision will be between vehicles $i+1$ and i and the relative velocity will be ϵ close to the relative velocity with which vehicles j and i would have collided if vehicles $i+1, \ldots, j-1$ were not there. By Corollary 1 this velocity is greater than v_A. The multiple collision is used only to make the calculation simpler. During the proof it will become apparent that (using Proposition 6) the effect is the same if the collisions take place pairwise with some arbitrary order. Before presenting the proof of the theorem we introduce the following proposition (the proof is given in the appendix).

Proposition 15 *Assume there exist i, j with $0 \leq i < j \leq N-1$ and $a_i, a_j \in [\underline{a}, \overline{a}]$ such that $(P_1(\Delta x_{ij}(0), v, v) > 0) \wedge (P_2(\Delta x_{ij}(0), v, v) > 0)$ is true. Then for $\epsilon > 0$ sufficiently small there exist $a_k \in [\underline{a}, \overline{a}]$ for $i < k < j$ and a time $T > 0$ such that no collisions have occurred in $[0, T)$ and $\Delta x_{i+1}(T) = \epsilon$ and $\Delta x_k(T) = 0$ for all $i+1 < k \leq j$.*

Proof: (of Theorem 5) Assume for the sake of contradiction that there exist $0 \leq i < j \leq N-1$ and $a_i, a_j \in [\underline{a}, \overline{a}]$ such that $(P_1(\Delta x_{ij}(0), v, v) > 0) \wedge (P_2(\Delta x_{ij}(0), v, v) > 0)$. We construct a platoon satisfying (20)-(22) and (32)-(33) (i.e. pick $a_k \in [\underline{a}, \overline{a}], k \neq i, j$ and $M_k \in [\underline{M}, \overline{M}]$ for all k) that will exhibit collisions at relative velocities greater than v_A under the default deceleration strategy.

Without loss of generality, for all $k < i$ (if any) choose $a_k \geq a_i$ and for all $k > j$ (if any) choose $a_k \leq a_j$ that satisfy the theorem condition. This is always possible as $a_i, a_j \in [\underline{a}, \overline{a}]$. Also, for all k choose $M_k = M$ for any $M \in [\underline{M}, \overline{M}]$. The choice of a_k ensures that the vehicles ahead of i and behind j do not interfere with our calculation. The choice of M_k is valid and makes the calculations considerably easier.

If $j = i + 1$ the conclusion of the theorem follows by contradiction, using Corollary 1. If $j > i + 1$, choose a_k for $i < k < j$ according to Proposition 15. Then, at time T a multiple collision takes place between vehicles $i + 1, \ldots, j$. By Proposition 6, the velocity of vehicle $i + 1$ after the collision will be the same as the velocity of vehicle j before the collision. For ϵ small enough, the next collision will be δ seconds later, between vehicles $i + 1$ and i, where δ satisfies:

$$\begin{aligned} \frac{a_i - a_{i+1}}{2}\delta^2 + (v_i(T^-) - v_j(T^-))\delta + \epsilon = 0 \quad &\text{if} \quad C_1(\Delta x_{ij}(0), v, v) \geq 0 \\ -\frac{a_{i+1}}{2}\delta^2 - v_j(T^-)\delta + \epsilon = 0 \quad &\text{if} \quad C_1(\Delta x_{ij}(0), v, v) < 0 \end{aligned}$$

At the time of impact $T + \delta$:

$$v_i(T + \delta) - v_{i+1}(T + \delta) = \begin{cases} v_i(T^-) - v_j(T^-) + \delta(a_i - a_{i+1}) & \text{if } C_1(\Delta x_{ij}(0), v, v) \geq 0 \\ -v_j(T^-) - \delta a_{i+1} & \text{if } C_1(\Delta x_{ij}(0), v, v) < 0 \end{cases}$$

In either case the root of interest $\delta \to 0$ as $\epsilon \to 0$. As a consequence, as $\epsilon \to 0$ the relative velocity at impact between vehicles $i + 1$ and i tends to $v_i(T) - v_j(T)$ which in turn tends to the relative velocity of collision between vehicles j and i if vehicles $i + 1, \ldots, j - 1$ were not there. This quantity is greater than v_A by Corollary 1, therefore there exists an ϵ small enough for which the relative velocity of collision between vehicles i and $i + 1$ is greater than v_A. The conclusion of the theorem follows by contradiction. ■

Corollary 4 *The conditions of Theorem 5 are necessary as long as* $\cap_{j=0}^{N-1}[\underline{d_j}, \overline{d_j}] \neq \emptyset$.

Proof: For all i and some $d \in \cap_{j=0}^{N-1}[\underline{d_j}, \overline{d_j}]$, choose $d_i = d$. The construction of Theorem 5 trivially generalizes. ■

7 Implications for Platooning

7.1 Bounds on the System Parameters for Safe Platooning

We start with the sufficient condition of Section 6.1. Consider a near uniform mass string and let $\overline{a} - \underline{a} = \epsilon$. Then, according to Corollary 3 all strings whose vehicles satisfy (20)–(22) are guaranteed to be safe under the default deceleration strategy if:

$$\left(1 - \frac{\overline{a}}{\underline{a}}\right) v - v_A \leq 0 \Rightarrow \epsilon \leq -\frac{\underline{a} v_A}{v} \tag{34}$$

Substituting "typical" values of $\underline{a} = -9ms^{-2}$ and $v_A = 3ms^{-1}$ leads to $\epsilon \leq 1.08$ for $v = 25ms^{-1}$ and $\epsilon \leq 0.9$ for $v = 30ms^{-1}$.

To make use of the necessary conditions of Section 6.2, note that:

$$\frac{\partial P_1}{\partial a_i} = -2\Delta x_{ij} \leq 0, \quad \frac{\partial P_1}{\partial a_j} = 2\Delta x_{ij} \geq 0, \quad \frac{\partial P_2}{\partial a_i} = \frac{a_j}{a_i^2} v_i^2 \leq 0, \quad \frac{\partial P_2}{\partial a_j} = -\frac{v_i^2}{a_i} + 2\Delta x_{ij} \geq 0$$

N	ϵ (ms^{-2})		
	$v = 25ms^{-1}$, $F = 1m$	$v = 30ms^{-1}$, $F = 1m$	$v = 25ms^{-1}$, $F = 2m$
2	4.5	4.5	2.25
3	2.25	2.25	1.125
4	1.5	1.5	1.125
5	1.125	1.125	1.125
≥ 6	1.125	0.9	1.125

Table 1: Maximum allowable difference in deceleration capability

Therefore, the condition $(P_1(\Delta x_{ij}(0), v, v) \leq 0) \vee (P_2(\Delta x_{ij}(0), v, v) \leq 0)$ for all $a_i, a_j \in [\underline{a}, \overline{a}]$ is equivalent to $(P_1(\Delta x_{ij}(0), v, v) \leq 0) \vee (P_2(\Delta x_{ij}(0), v, v) \leq 0)$ for $a_i = \underline{a}$ and $a_j = \overline{a}$ or equivalently:

$$(-2(\underline{a} - \overline{a})\Delta x_{ij} - v_A^2 \leq 0) \vee \left((1 - \frac{\overline{a}}{\underline{a}})v^2 + 2\overline{a}\Delta x_{ij} - v_A^2 \leq 0\right)$$

To further simplify the calculation assume that at steady state the string (platoon) satisfies the uniform spacing assumption, i.e. $\Delta x_i = F$ for all i. Then the necessary condition for string safety requires that for all $i \leq j$:

$$\begin{aligned} & (2\epsilon(j-i)F - v_A^2 \leq 0) \vee \left(-\frac{\epsilon}{\underline{a}}v^2 + 2\overline{a}(j-i)F - v_A^2 \leq 0\right) \\ \Rightarrow \quad & \epsilon \leq \max\left\{\frac{v_A^2}{2(j-i)F}, \frac{2(j-i)\underline{a}^2F - \underline{a}v_A^2}{v^2 - 2(j-i)\underline{a}F}\right\} \end{aligned}$$

This condition should hold for all $i \leq j$, therefore, for a platoon of size N to be safe we need:

$$\epsilon \leq \min_{j-i=1,..,N-1} \max\left\{\frac{v_A^2}{2(j-i)F}, \frac{2(j-i)\underline{a}^2F - \underline{a}v_A^2}{v^2 - 2(j-i)\underline{a}F}\right\} \tag{35}$$

Table 1 shows the necessary condition for the variation in deceleration capability for $\underline{a} = -9ms^{-2}$ and $v_A = 3ms^{-1}$. The numbers indicate that the sufficient condition is conservative for small platoon sizes but approaches the necessary condition as the platoon size increases[7]. Based on the characteristics of vehicles on current highways the bound on ϵ is reasonable for $N = 2$ but rather restrictive for higher platoon sizes (even under perfect road conditions). Note also that the calculation saturates after the first few followers; a similar observation was made in [9] about the increase in deceleration effort required along a platoon for "string stability".

7.2 Ways to Improve Safety

The above calculations indicate that the safety of the platooning system under emergency braking can only be guaranteed under rather limited conditions, in particular for small platoons consisting of vehicles of similar deceleration capabilities. A number of alternatives can be considered in an attempt to improve on these restrictions as much as possible.

[7] Using Lemma 7 it is easy to verify that for $N = 2$ the values of Table 1 are both necessary and sufficient.

7.2.1 Modify the Parameters

Taking partials of equations (34) and (35) with respect to $\underline{a}$, v and v_A indicates that the conditions become easier to satisfy as $\underline{a}$ and v decrease and v_A increases. The value of $\underline{a}$ is lower bounded by the limitations of the tires and brakes; the value of $-9ms^{-2}$ is already rather optimistic for most vehicles and driving conditions. Likewise, reducing the value of v reduces the highway throughput; the value $v = 25ms^{-1}$ is already considered low. Finally, increasing v_A is unacceptable from the point of view of safety; $v_A = 3ms^{-1}$ is already considered high. Taking partials of (35) with respect to F indicates that the first term becomes easier to satisfy as F decreases while the second term becomes easier to satisfy as F increases. In the numerical examples the first term dominates for platoon sizes up to $N = 6$. It is therefore likely that a reduction in the following distance can lead to improvement in safety; however, the value $F = 1m$ is already rather low for the current technology.

Effect of d is small. $\underline{M}$ and $\overline{M}$ have no effect, the necessary condition needs to hold even for identical masses. Try to reduce α_i. Tradeoff: $\alpha(0) = 1$, therefore need some relatively bad collisions to absorb energy.

7.2.2 Non-spontaneous Platooning

Assume vehicles can estimate their own deceleration capability and decide to join a platoon only if a sufficient condition is satisfied. For example, order platoons by increasing deceleration capability. Safety in this case is guaranteed by Lemma 4. Safety will be sensitive to estimate of deceleration capability. Problems:

1. Estimate difficult to obtain on line, has to be inferred by ABS measurements and may be inaccurate until maximum deceleration is applied. Large safety margins likely to be needed.

2. Deceleration capability changes on line. In a slow time scale due to brake and tire ware and in a fast time scale due to variation in driving conditions (weather and terrain), brake heating etc. Platoon may cease to be safe at some point. Situation seems hopeless for inconsistent terrain, e.g. ice patches.

3. Average platoon size no longer arbitrary. Probability distribution for deceleration capability and queuing argument can be used to obtain expected platoon size. Relatively simple if platoon ordered in increasing deceleration capability can be very complicated if more elaborate sufficient condition is used.

7.2.3 Better Emergency Controllers

Try to solve problems 4 and 5.

References

[1] N. Lynch, R. Segala, F. Vaandrager, and H. Weinberg, "Hybrid I/O automata," in *Hybrid Systems III*, no. 1066 in LNCS, pp. 496–510, Springer Verlag, 1996.

[2] C.-Y. Chan, "Collision analysis of vehicle following operations by two-dimentional simulation model," Tech. Rep. UCB-ITS-PRR-97-4, Institute of Transportation Studies, University of California, Berkeley, 1997.

[3] A. Hitchcock, "Casualties in accidents occuring during split and merge maneuvers," tech. rep., PATH Technical Memo 93-9, Institute of Transportation Studies, University of California, Berkeley, 1993.

[4] J. Lygeros, D. N. Godbole, and S. Sastry, "A game theoretic approach to hybrid system design," in *Hybrid Systems III*, no. 1066 in LNCS, pp. 1–12, Springer Verlag, 1996.

[5] P. Li, L. Alvarez, R. Horowitz, P.-Y. Chen, and J. Carbaugh, "Safe velocity tracking controller for AHS platoon leader," in *IEEE Conference on Decision and Control*, pp. 2283–2288, 1996.

[6] J. Frankel, L. Alvarez, R. Horowitz, and P. Li, "Safety oriented maneuvers for IVHS," in *American Control Conference*, pp. 668–672, 1995.

[7] J. Lygeros, "To brake or not to brake? is there a question?," in *IEEE Conference on Decision and Control*, pp. 3723–3728, 1996.

[8] E. Dolginova and N. Lynch, "Safety verification for automated platoon maneuvers: a case study," in *Proceedings of HART97* (O. Maler, ed.), no. 1201 in LNCS, pp. 154–170, Berlin: Springer Verlag, 1997.

[9] D. Swaroop, *String Stability of Interconnected systems: an application to platooning in automated highway systems*. PhD thesis, Department of Mechanical Engineering, University of California, Berkeley, 1994.

[10] J. Lygeros, D. N. Godbole, and S. Sastry, "A verified hybrid controller for automated vehicles," Tech. Rep. UCB-ITS-PRR-97-9, Institute of Transportation Studies, University of California, Berkeley, 1997.

[11] S. Shladover, "Operation of automated guideway transit vehicles in dynamically reconfigured trains and platoons," Tech. Rep. UMTA-MA-0085-79-3, U.S.Deprtement of Transportation, 1979.

[12] J. Lygeros, D. N. Godbole, and M. E. Broucke, "Extended architecture for degraded modes of operation of IVHS," in *American Control Conference*, pp. 3592–3596, 1995.

[13] D. N. Godbole, J. Lygeros, E. Singh, A. Deshpande, and A. Lindsey, "Design and verification of coordination layer protocols for degraded modes of operation of AHS," in *IEEE Conference on Decision and Control*, pp. 427–432, 1995.

[14] Y. Yang and B. Tongue, "Intra-platoon collision behavior during emergency operations," *Vehicle System Dynamics*, vol. 23, no. 4, pp. 279–292, 1994.

[15] M. Tomizuka, S. Patwardhan, W. B. Zhang, and P. Devlin, "Theory and experiments of tire blow-out effects and hazard reduction control for automated vehicle lateral control system," in *American Control Conference*, pp. 1207–1209, 1994.

[16] D. N. Godbole and J. Lygeros, "Tools for safety and throughput analysis of automated highway systems," in *American Control Conference*, pp. 2031–2035, 1997.

[17] N. Lynch, *Distributed Algorithms*. Morgan Kaufmann, 1996.

A Hybrid Automata Pseudo-Code

A.1 Plant Automaton Code

Variables:

Input:

$u_i \in [a_i^{min}, a_i^{max}]$, for all $i \in \{0, \ldots, N-1\}$

Internal:

$\Delta x_i \in \mathbb{R}$, for all $i \in \{0, \ldots, N-1\}$, initially ≥ 0

$v_i \in \mathbb{R}$, for all $i \in \{0, \ldots, N-1\}$, initially ≥ 0

$Touching_i \in$ **Bool**, for all $i \in \{0, \ldots, N\}$, initially False

Output:

$y_i^p \in \mathbb{R}^3$, for all $i \in \{0, \ldots, N-1\}$

Derived:

$acc_i \in \mathbb{R}$, for all $i \in \{0, \ldots, N-1\}$

(see text)

Actions:

Input:

e, the environment action

Internal:

$Collision_i$, for $i \in \{1, \ldots, N-1\}$

$Touch_i$, for $i \in \{1, \ldots, N-1\}$

$Separate_i$, for $i \in \{1, \ldots, N-1\}$

Discrete Transitions:

e:

Effect: arbitrarily reset the input variables

$Collision_i$:

Precondition:

$(\Delta x_i = 0) \wedge (v_i > v_{i-1})$

Effect:

Reset v_i and v_{i-1} to v_i' and v_{i-1}' so that:

$M_i v_i' + M_{i-1} v_{i-1}' = M_i v_i + M_{i-1} v_{i-1}$

$v_{i-1}' - v_i' = (v_i - v_{i-1})\alpha_i$

$Touch_i$:

Precondition:

$(Touching_i = \text{False}) \wedge (\Delta x_i = 0) \wedge (v_i = v_{i-1}) \wedge (acc_i \geq acc_{i-1})$

Effect:

$Touching_i := \text{True}$

$Separate_i$:

Precondition:

$(Touching_i = \text{True}) \wedge [(acc_i < acc_{i-1}) \vee (v_i < v_{i-1})]$

Effect:

$Touching_i := \text{False}$

Trajectories:

Input variables follow arbitrary trajectories
For all $i \in \{0, \ldots, N-1\}$ and for all $t \geq 0$:

$\dot{\Delta x}_i(t) = v_{i-1}(t) - v_i(t)$

$\dot{v}_i(t) = acc_i(t)$

$Touching_i(t) = Touching_i(0)$

$y_i^p(t) = [\Delta x_i(t) \;\; v_i(t) \;\; acc_i(t)]^T$

Trajectories stop once the precondition of an action becomes true

B Construction of Maximal Partitions

Consider a segment S with a weighted average function a. Assume without loss of generality that $S = \{1, \ldots, n\}$. The following algorithm attempts to construct the maximal partition of S. The algorithm maintains a state, consisting of a candidate partition of S, $\mathcal{S} = \{S_1, \ldots, S_k\}$, where k may change along the algorithm execution.

Initialization: Set $k = n$, $\mathcal{S} = \{\{1\}, \{2\}, \ldots, \{n\}\}$.

While $k > 1$ and $\exists i \in \{1, \ldots, k-1\}$ such that $a(S_{i-1}) \leq a(S_i)$

Do

$S_{i-1} = S_{i-1} S_i$

$S_j = S_{j+1}$ for $j \in \{i, \ldots, k-1\}$.

$k = k - 1$

oD

Proposition 16 *The following properties are invariant for the algorithm:*

1. *$\mathcal{S}$ is a partition of S.*
2. *$S_i \in \mathcal{S}$ are unsplitable.*

Proof: $\mathcal{S}$ is initially a partition (a trivial one). Each step of the algorithm can join two adjacent elements of $\mathcal{S}$ into a single element. The result is again a partition and part 1 follows.

$S_1, S_2, \ldots, S_n$ are initially unsplitable (vacuously). At each step of the algorithm S_i and S_{i-1} may be joined if and only if $a(S_{i-1}) \leq a(S_i)$. By Proposition 3, the resulting segment is also unsplitable. ■

Lemma 11 *The algorithm terminates in a finite number of steps. Upon termination $\mathcal{S}$ is a maximal partition of S.*

Proof: k is monotone decreasing, is initially equal to n and is bounded below by 1. Hence the algorithm is guaranteed to terminate.

The algorithm terminates if either $k = 1$ or $a(S_{i-1}) > a(S_i)$ for all $i \in \{1, \ldots, k-1\}$. Moreover, the S_i are unsplitable by Proposition 16. Therefore, $\mathcal{S}$ is a maximal partition. ■

C Additional Proofs

Proposition 10 *If A and B are two unsplittable subsegments of S and $A \cap B \neq \emptyset$, then $A \cup B$ is an unsplittable subsegment of S.*

Proof: If A and B are subsegments of S and $A \cap B \neq \emptyset$, $A \cup B$ is also a subsegment of S. Assume, without loss of generality that $\min(A) \leq \min(B)$. If $A \subseteq B$ ($A \supseteq B$) then $A \cup B = B$ ($A \cup B = A$) and hence unsplittable. Otherwise, as A and B are unsplittable:

$$a(A \setminus (A \cap B)) \leq a(A \cap B) \leq a(B \setminus (A \cap B))$$

By the properties of the weighted average function this implies that:

$$a(A \setminus (A \cap B)) \leq a(A) \leq a(A \cap B) \leq a(B) \leq a(B \setminus (A \cap B))$$

Assume that $A \cup B = LR$ for some segments L, R. Note that either $L \subseteq A$ or $R \subseteq B$. We would like to show that $a(L) \leq a(R)$. Assume first that $L \subseteq A$. As A is unsplittable $a(L) \leq a(A \setminus L)$, therefore, by definition of weighted average, $a(L) \leq a(A) \leq a(A \setminus L)$. But $R = (A \setminus L)(B \setminus (A \cap B))$, therefore:

$$a(R) \geq \min\{a(A \setminus L), a(B \setminus (A \cap B))\} \geq a(A)$$

It follows that $a(L) \leq a(R)$. The proof is similar if $R \subseteq B$. ■

Proposition 11 *If A and B are two unsplittable subsegments of S, AB is defined and $a(A) \leq a(B)$, then AB is an unsplittable subsegment of S.*

Proof: Clearly, AB is a subsegment of S. As before, let $AB = LR$ for some subsegments L and R and assume first that $L \subseteq A$. As A is unsplittable $a(L) \leq a(A \setminus L)$, therefore, by definition of weighted average, $a(L) \leq a(A) \leq a(A \setminus L)$. Moreover, $a(A) \leq a(B)$ implies that:

$$a(R) = a((A \setminus L)B) \geq \min\{a(A \setminus L), a(B)\} \geq a(A)$$

The conclusion follows. The proof is similar in the case $R \subseteq B$. ■

Proposition 12 *If $S_1 \ldots S_n$ is a maximal partition of S, $1 \leq l \leq k \leq n$ and $\hat{S}_{lk} = \cup_{m=l}^{k} S_m$ then $a(\hat{S}_{lk}) \geq a(S_k)$.*

Proof: For all l proceed by induction on k. If $k = l$, $\hat{S}_{lk} = S_l = S_k$ and the claim holds. Assume $a(\hat{S}_{lk}) \geq a(S_k)$ holds for some $k \geq l$ and show that it holds for $k + 1$. By the defining property of a weighted average function:

$$a(\hat{S}_{l(k+1)}) \geq \min\{a(\hat{S}_{lk}), a(S_{k+1})\}$$

But $a(\hat{S}_{lk}) \geq a(S_k) > a(S_{k+1})$ (by induction hypothesis and maximality). Therefore, $a(\hat{S}_{l(k+1)}) \geq a(S_{k+1})$. ■

Proposition 13 *a is a weighted average function on S.*

Proof: Let L and R be two subsegments of S such that LR is defined. Assume $a(L) \leq a(R)$. We would like to show that $a(L) \leq a(LR) \leq a(R)$. Indeed:

$$\begin{aligned}
a(L) \leq a(LR) &\Leftrightarrow \frac{\sum_{l\in L} M_l u_l}{\sum_{l\in L} M_l} \leq \frac{\sum_{l\in LR} M_l u_l}{\sum_{l\in LR} M_l} \\
&\Leftrightarrow \left(\sum_{l\in L} M_l u_l\right)\left(\sum_{l\in LR} M_l\right) \leq \left(\sum_{l\in LR} M_l u_l\right)\left(\sum_{l\in L} M_l\right) \\
&\Leftrightarrow \left(\sum_{l\in L} M_l u_l\right)\left(\sum_{l\in L} M_l + \sum_{l\in R} M_l\right) \leq \left(\sum_{l\in L} M_l u_l + \sum_{l\in R} M_l u_l\right)\left(\sum_{l\in L} M_l\right) \\
&\Leftrightarrow \left(\sum_{l\in L} M_l u_l\right)\left(\sum_{l\in R} M_l\right) \leq \left(\sum_{l\in R} M_l u_l\right)\left(\sum_{l\in L} M_l\right) \\
&\Leftrightarrow \frac{\sum_{l\in L} M_l u_l}{\sum_{l\in L} M_l} \leq \frac{\sum_{l\in R} M_l u_l}{\sum_{l\in R} M_l} \\
&\Leftrightarrow a(L) \leq a(R)
\end{aligned}$$

The proofs for $a(LR) \leq a(R)$ and in the case when $a(R) \leq a(L)$ are similar. ∎

Proposition 14 *If $\alpha_i \equiv 1$ and $M_i = M_j$ for all $N_1 \leq i, j \leq N_2$ then all possible orders of pairwise resolution lead to $v'_{N_1} = v_{N_2}$, $v'_{N_1+1} = v_{N_2-1}$, ..., $v'_{N_2} = v_{N_1}$ (i.e. the order of the velocities is reversed).*

Proof: Assume that the multiple collision is resolved by pairwise collisions and let $V(k) = \{v^{(k)}_{N_1}, \ldots, v^{(k)}_{N_2}\}$ denote the velocities of vehicles $N_1, \ldots, N_2$ after the k^{th} pairwise resolution has been performed. Clearly, $V(0) = \{v_{N_1}, \ldots, v_{N_2}\}$. We show that $V(k) = V(0)$ (up to reordering of the elements).

Proceed by induction. Assume that after k pairwise resolutions $V(k)$ is a permutation of $V(0)$. If further resolutions are needed, there exist (possibly many) $j \in (N_1, N_2]$ such that $v_j > v_{j-1}$. Pick any such j and resolve the conflict between j and $j-1$. For all $i \in [N_1, N_2]$ with $i \notin \{j, j-1\}$, $v_i^{(k+1)} = v_i^{(k)}$. Under the proposition assumptions $v_j^{(k+1)}$ and $v_{j-1}^{(k+1)}$ satisfy:

$$\begin{aligned}
M_j v_j^{(k+1)} + M_{j-1} v_{j-1}^{(k+1)} = M_j v_j^{(k)} + M_{j-1} v_{j-1}^{(k)} &\Rightarrow v_j^{(k+1)} + v_{j-1}^{(k+1)} = v_j^{(k)} + v_{j-1}^{(k)} \\
v_{j-1}^{(k+1)} - v_j^{(k+1)} = (v_j^{(k)} - v_{j-1}^{(k)})\alpha_j &\Rightarrow v_{j-1}^{(k+1)} - v_j^{(k+1)} = v_j^{(k)} - v_{j-1}^{(k)}
\end{aligned}$$

which implies that $v_{j-1}^{(k+1)} = v_j^{(k)}$ and $v_j^{(k+1)} = v_{j-1}^{(k)}$. Therefore $V(k+1)$ is also a permutation of $V(0)$. Note that the argument is independent of the ordering of the pairwise resolutions.

Resolutions will keep taking place until $v^{(k)}_{N_1} \geq v^{(k)}_{N_1+1} \geq \ldots \geq v^{(k)}_{N_2}$. As initially $v_{N_1} < v_{N_1+1} < \ldots < v_{N_2}$ and $V(k)$ is a permutation of $V(0)$, the claim of the proposition follows. ∎

Proposition 15 *If $\alpha_i < 1$ or $M_i \neq M_j$ for some $i, j \in [N_1, N_2]$, the state after the collision is resolved, x' , may depend on the order in which the collisions are resolved.*

Proof: By counter example. Consider a string of 3 vehicles undergoing simultaneous collisions. Assume the velocities at impact are $v_0 = 0$, $v_1 = 4$ and $v_2 = 8$. First let $M_0 = M_1 = M_2$ and

$\alpha_1 = \alpha_2 = 0.5$. The multiple collision can be resolved pairwise in two different orders:

$$\begin{array}{lllllll} v_0 = 0 & & v_0 = 3 & & v_0 = 3 & & v_0 = 5.4375 \\ v_1 = 4 & 1 \to 0 & v_1 = 1 & 2 \to 1 & v_1 = 6.25 & 1 \to 0 & v_1 = 3.8125 \\ v_2 = 8 & & v_2 = 8 & & v_2 = 2.75 & & v_2 = 2.75 \end{array}$$

$$\begin{array}{lllllll} v_0 = 0 & & v_0 = 0 & & v_0 = 5.25 & & v_0 = 5.25 \\ v_1 = 4 & 2 \to 1 & v_1 = 7 & 1 \to 0 & v_1 = 1.75 & 2 \to 1 & v_1 = 4.1875 \\ v_2 = 8 & & v_2 = 5 & & v_2 = 5 & & v_2 = 2.5625 \end{array}$$

Now let $\alpha_1 = \alpha_2 = 1$ and $M_0 = M_1/2 = M_2/3$. Collisions can again be resolved in two ways:

$$\begin{array}{lllllll} v_0 = 0 & & v_0 = 16/3 & & v_0 = 16/3 & & v_0 = 32/3 \\ v_1 = 4 & 1 \to 0 & v_1 = 4/3 & 2 \to 1 & v_1 = 28/3 & 1 \to 0 & v_1 = 20/3 \\ v_2 = 8 & & v_2 = 8 & & v_2 = 8/3 & & v_2 = 8/3 \end{array}$$

$$\begin{array}{lllllll} v_0 = 0 & & v_0 = 0 & & v_0 = 176/15 & & v_0 = 880/75 \\ v_1 = 4 & 2 \to 1 & v_1 = 44/5 & 1 \to 0 & v_1 = 44/15 & 2 \to 1 & v_1 = 388/75 \\ v_2 = 8 & & v_2 = 24/5 & & v_2 = 24/5 & & v_2 = 248/75 \end{array}$$

Total momentum is conserved in all cases. ■

Proposition 17 *Assume there exist i, j with $0 \le i < j \le N-1$ and $a_i, a_j \in [\underline{a}, \overline{a}]$ such that $(P_1(\Delta x_{ij}(0), v, v) > 0) \wedge (P_2(\Delta x_{ij}(0), v, v) > 0)$ is true. Then for $\epsilon > 0$ sufficiently small there exist $a_k \in [\underline{a}, \overline{a}]$ for $i < k < j$ and a time $T > 0$ such that no collisions have occurred in $[0, T)$ and $\Delta x_{i+1}(T) = \epsilon$ and $\Delta x_k(T) = 0$ for all $i + 1 < k \le j$.*

Proof: The easiest way to prove the claim is by a "geometric" argument (it can also be proved algebraically). First note that, as $v_i(0) = v_j(0) = v$ and $\Delta x_{ij} \ge 0$, $P_1(\Delta x_{ij}(0), v, v) > 0$ implies that $a_i < a_j$. Moreover, by Corollary 1, $(P_1(\Delta x_{ij}(0), v, v) > 0) \wedge (P_2(\Delta x_{ij}(0), v, v) > 0)$ implies that if i and j were the only vehicles in the string they would have collided. Therefore, for every $\epsilon < \Delta x_{ij}(0)$ there exists a $T > 0$ such that at $\Delta x_{ij}(T) = \epsilon$ (again if vehicles i and j were the only vehicles in the string).

Figure 7 shows the velocities of vehicles i and j as a function of time, under the default deceleration strategy. Case (a) corresponds to $C_1 \ge 0$ (refer to equation (23) where the vehicles collide while they are still moving) while case (b) corresponds to $C_1 < 0$ where the vehicles collide after vehicle i has stopped. The slopes of the two lines are equal to a_i and a_j respectively and the area of the shaded region is equal to $\Delta x_{ij}(0) - \epsilon$ while the area of the hashed region is equal to ϵ. More generally, for any pair of vehicles k and l with $i \le k < l \le j$ and $a_k \le a_l$ define $A_{kl}(t)$ to be the area of the shaded region in Figure 7, case (c). Note that, as long as there are no collisions [8]. $\Delta x_{kl}(t) = \Delta x_{kl}(0) - A_{kl}(t)$

Choosing a_k to satisfy the proposition involves choosing the slopes of the $v_k(t)$. The procedure is inductive. We start by choosing a_{i+1} such that $A_{i(i+1)}(T) = \Delta x_{i+1}(0) - \epsilon$. Assume that after $k - i$ steps $A_{ik}(T) = \Delta x_{ik}(0) - \epsilon$ and choose a_{k+1} to make $A_{k(k+1)}(T) = \Delta x_{k+1}(0)$. By construction $a_i \le a_{i+1} \le \ldots \le a_{j-1} \le a_j$ as $\Delta x_k(0) \ge 0$ and therefore $a_k \in [\underline{a}, \overline{a}]$ for all k. If i is still moving at time T the areas $A_{k(k+1)}(t)$ are monotonically increasing with time. Therefore, as $A_{k(k+1)}(T) = \Delta x_{k+1}(0)$, no collisions can occur in $[0, T)$. In the case where vehicle i is stopped at time T, some of the other vehicles may also have to stop touching each other. Still no collisions take place in $[0, T)$, as the relative velocity at which they touch is zero. ■

[8] Recall that by the definition of a collision vehicles are allowed to touch at zero relative velocity.

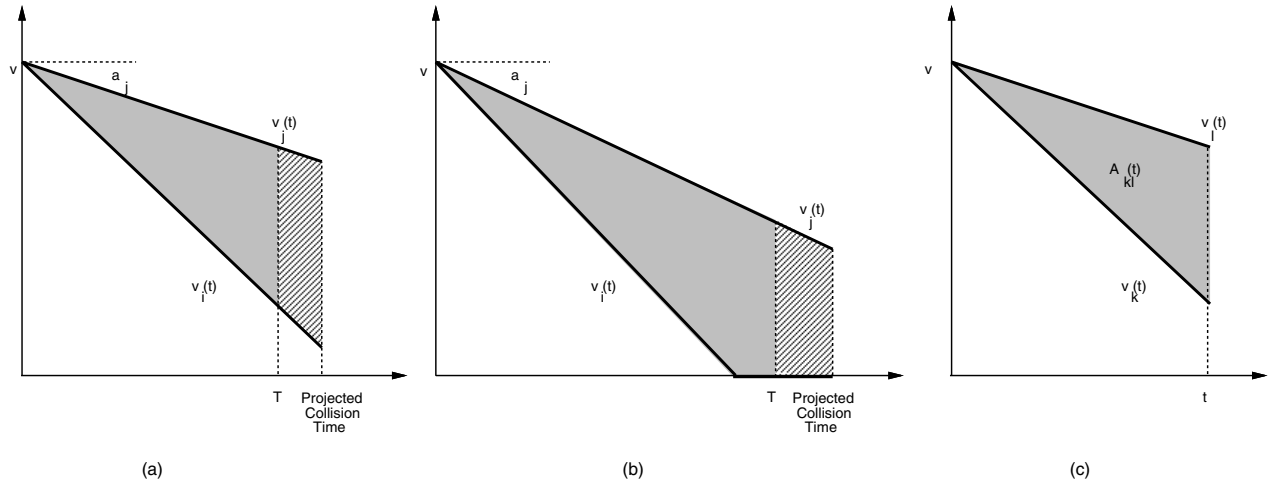

Figure 7: The definition of T and $A_{kl}(t)$