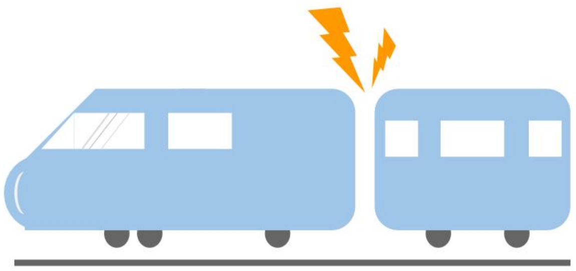
Lots & Lots of Trains: Efficient Transit with Distributed Hybrid Systems

Joshua Kalapos, Karen Bowman
Carnegie Mellon University



Abstract

Subway systems can be found in nearly every modern city. Since the late 19th century, metros have provided a way for everyone to quickly get where they need to be. Effective subway lines can reducing surface traffic congestion, as well as reduce emissions by taking cars off the road. As such, it is important to understand how to design a safe and efficient subway system. The system described in this paper describes the transit of trains through multiple modifiable railway designs. We attempt to model smaller sub-tracks that can be combined together to form a full railway. Additionally, for each of our sub-models we describe it's efficiency implications for different safety scenarios. By justifying their safety and relative efficiency, we hope to have created a system that is scalable in both track size and in number of trains.

1 Introduction

Fast and timely subway systems are essential to the existence of the bustling economic and social activity of many metropolitan areas. The world's busiest subway system in Beijing registers around 10 million riders per day, while all city subways in the top 10 register at least 4 million daily riders. The New York subway boasts 6,418 vehicles on its lines across 424 unique stations [Sch19]. Scheduling the movement of each train is incredibly sensitive to efficiency, as even small delays can get amplified down the line and cause major blockages of traffic.

Balanced with efficiency, safety is of course a concern as well, as, though they are not everyday occurrences, collisions and de-railings of subway cars do occur from time to time. As a result, efforts to increase subway safety have been put in place across different cities, including traffic signals, speed limits, and buffers. These methods promote safety to a varying degree, but have potentially increased the impact of traffic delays [Gor18]. For example, increasing buffer size between trains reduces the chances of collision, but decreases the number of trains capable of passing through a station in a given time period. Specific implementation of safety protocols also has consequences, where in the New York subway speed limit is enforced by timer signals. These signals detect passing trains and trigger emergency brakes on the train if their timer finds the train to be traveling too quickly. Not only are these signals often faulty, but drivers often conservatively slow down to avoid the lost vacation days and forced early retirement that come with failing a signal. Despite original intentions, the timer signals caused significant slowdown [Pea18].

Subway systems are complex, with their arbitrary number of railroad cars, separate lines, and faulty technical and human factors at play. However, even simple models can give us insights into the consequences to efficiency and safety that any subway policy may have. Given some realistic set of assumptions about a subway line, we plan on providing a model that will be able to deliver an evaluation on both the subway system's safety and efficiency.

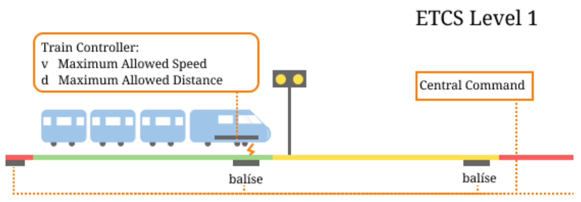
2 Related Work

In the case where there is a large number of trains in our system, it could be useful to consider applying quantified differential dynamic logic (QdL). QdL is logic for specifying and verifying distributed hybrid systems. With so many trains in action, we have to consider communication in addition to computation and control. Just computation and control form our familiar hybrid systems, but communicated is used in more complex applications. For example, a distributed car control application discussed in "Quantified Differential Dynamic Logic for Distributed Hybrid Systems" [Pla10] has many traits that would be similar to our use case for trains. Like with cars, we have to consider many trains moving continuously along the tracks. Trains may leave at the end of the line, or join the system at the beginning of the track. Though implemented in a different version of KeyMaeraX, in the paper they prove collision freedom in distributed car control which could be useful for reference when trying to prove the safety of our own mechanisms.

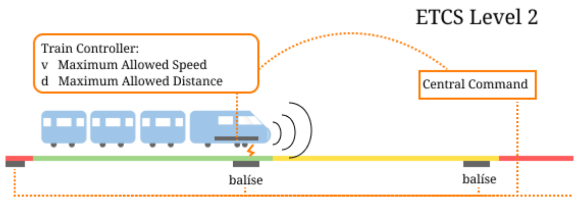
CoasterX [Boh+18] is similar to our goals in that its' goal is to verify a physical system as safe based on aspect of that system that are specified by the user. CoasterX is built on top of KeyMaeraX's core to create a unique user interface that is able to verify this unique input by re-using verification across different components. CoasterX reduced verification of the entire track to instead verification of components, which we will likely also do if we do a full track verification.

What CoasterX does not do, it provide any measure of safety vs efficiency trade-off. For a roller-coaster, the concept of efficiency is limited. However, for subways the concept of efficiency is in fact important. We want a tool that can show the direct trade-offs between efficiency and safety in the system.

Outside of the realm of formal proving, there are many groups that have tried their hand at



(a) ETCS Level 1 Control: Discrete train communication



(b) ETCS Level 2 Control: Continuous train communication

Figure 1: European Train Control System

modeling train safety for real train infrastructure, to help engineers plan real systems.

A great example of this is OpenTrack, a Swiss simulation software that models trains of different types in different weather conditions. It also determines if the ”headway’ between trains will be preserved given a specific infrastructure system and intended station timetable. However, OpenTrack is a synchronous time-step validator; this means that it checks for successive time points if the trains are in acceptable positions given their parameters. While this does model the train system as a whole reasonably accurately, it makes no claims of formal verification of safety. This is an area where our model could provide new value, if further developed. Finally, though it can simulate the impact of a random set of delays within given parameters, it cannot prove any definite relationships surrounding possible delays in a given track system. [NH04]

3 Background

3.1 Realistic Train Systems

There are countless train systems worldwide, but we chose to focus on European trains, specifically trains that use the European Train Control System (ETCS), due to its’ prevalence and quality of documentation.

A brief explanation of ETCS is as follows, also illustrated by figure 1.(a) and 1.(b).

ETCS Control: Figure 1.(a) and 1.(b)

ETCS is a system in which a Central Command tracks trains in a system, and informs them of control decisions. A control decision, often called a ”movement authority” in the industry, is a set of two parameters:

Maximum Speed: A speed the train must stay under until given a new Maximum Speed. Maximum Distance: The distance the train must brake by until given a new Maximum Distance.

These are updated regularly, giving the train ”permission” to continue movement.

There are many levels of ETCS, but the most common versions are Level-1 and Level-2, as pictured above. Level-1 is more common but less efficient; most systems aim to transition to Level-2.

ETCS Level-1

ETCS Level-1 uses balises, which are electronic beacons embedded in the track at fixed intervals, to communicate to trains and inform the Central Command of each train position.

This is the only reliable point of communicated from trains to Central Command in Level-1. In this way, train information and control is highly discrete.

ETCS Level-2

ETCS Level-2 also uses balises to inform Central Command of each train position, but also has a continuous radio signal to Central Command. This signal can receive commands, and gives Central Command information about current position in a continuous manner.

Central Command in Level-2 can make and send control decisions based off of continuous train positions.

3.2 Realistic Efficiency Measures

As mentioned earlier, there are three important aspects of efficiency for a system of trains. An engineer designing train infrastructure may weigh some measures over others depending on their needs. From these factors, a track is designed, a control system is chosen, and a timetable is created for the different stations. They are intuitively defined as follows:

1. **Throughput:** The throughput of a system from one station to another is defined as the number of trains that can pass through a given route in a set period time.
2. **Time of Travel:** The time of travel is defined as the time it takes for a train to travel from one station to another.
3. **Total Delay:** The sum of all delays for all trains passing through a set of stations throughout a given period of time.

A set of formally defined values for these different measures, as well as justification for a final efficiency measure will follow in Section 3.4.

3.3 Simplifying Assumptions on Train Motion

Our main assumption on the motion of the train is that it is one dimensional.

Since our main concern is control safety, we will ignore some of the effects of other aspects of physics on each train such as;

- Track Friction
- Wind Resistance
- Train Momentum

- Effect of Train Torque

We also make the following simplifying assumptions on the physical trains themselves for ease of modeling;

- **Train Length is Zero:** We see the length of the train as a complication unnecessary to our model, though this could be incorporated into the models without major modification to our proofs
- **Every Train is the Same:** Each train has the same acceleration and braking constants, though this could be incorporated into the models without major modification to our proofs

3.4 Simplifying Assumptions on Train Control

We assume that trains in our systems are traveling under *ETCS - Level 2 Protocol*, which means that they are equipped to receive radio signals at all times from a controller that knows the position of all other trains. This means they can be assigned a maximum speed and movement authority (a distance they are allowed to travel before braking) at any point in time.

We demonstrate a brief exploration of the limitations of ETCS - Level 1 in Section 5.5.5

This assumption is informed by the knowledge that Level-2 is the protocol used by higher impact train systems because it allows for more efficient train control decisions. The European Train Control System is also intended to replace the many variations of control in coming years throughout Europe [con19].

3.5 Simplifying Assumptions on Train Efficiency

The three main factors of Train Efficiency, formally defined, are as follows.

Train Throughput

Given a set of stations $\mathbf{S}$ that represent a route, and a set of trains in the system $\mathbf{T}$, the throughput of that route in a given time interval (t_i, t_f) is:

$$\frac{|t : t \in \mathbf{T}, \text{ and t stopped at s}, \forall s \in \mathbf{S} \text{ in time interval } (t_i, t_f)|}{t_f - t_i}$$

Time of Travel

Given some train t and the time it left some station a, t_a, and the time it arrived to some station b, t_b where such a route exists, the Time of Travel is:

$$t_b - ta$$

Train Delay

Given a set of stations $\mathbf{S}$ in a system, and a set of trains in the system $\mathbf{T}$, the total Train Delay is the sum of all delays for train $t \in \mathbf{T}$, whose actual arrival time at station s is $t_{s,actual}$ and expected arrival time is $t_{s,expected}$:

$$\sum_{t \in \mathbf{T}, s \in \mathbf{S}} t_{s,actual} - t_{s,expected} \text{ if } t_{s,expected} < t_{s,actual}$$

Throughput is an often studied measure of efficiency, which is best solved using network flow computations. Time of Travel is a simple computation. We have decided not to formally prove anything regarding throughput or time of travel, though we acknowledge they are important. Instead, we simplify their weight by remembering that reducing the speed of trains negatively impacts those values, and therefore choosing reasonable speeds.

We chose to focus on Train Delay in our studies on train efficiency. Specifically, we were interested in proving limits on overall train delays given a certain environment, and the relationship between a set of trains in the system.

The final limit we introduced for modeling and analysis is a maximum possible delay time.

4 Analysis of Safety & Delay

5 Approach

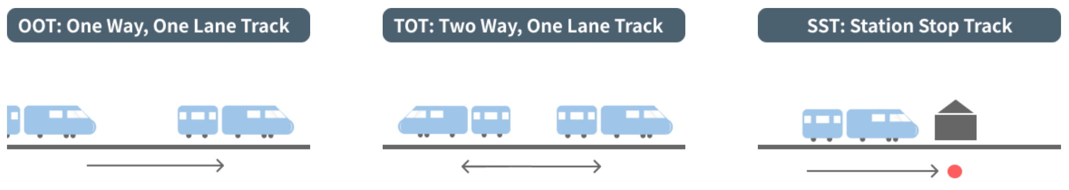
5.1 Track Composition Modeling

Within a rail system we've identified multiple types of segments that trains may travel on. For each different class of track, we attempt to prove its safety. The reasoning behind this is that these segments, now proven to be safe, can be inductively 'glued' together to make a large scale, diverse rail system. Here we'll describe the different track classes.

Our first set of rail types are simple segments. These will likely be the majority of a railway's design.

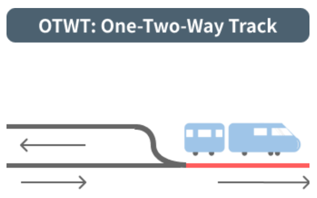
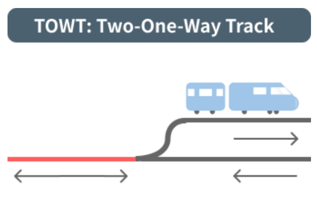
- one-lane, one-way track
- one-lane, two-way track
- track with station

A one-lane, one-way track is as it sounds. The track only serves a single direction. On the other hand, the two-way track can have trains traveling in either direction, so long as two



(a) (b) (c)

Figure 2: Simple Track Segments

(a) (b)

Figure 3: Transition Track Segments

trains aren't steaming towards each other. Lastly, a track with a station serves as a place that the train has to come to a stop at.

The next set of rail types are the transitions that will serve as connectors between the different types of segments.

- two one-ways merge to two-way
- two-way to two one-ways

These transition points are especially important as all kinds of unsafe collisions can occur. Perhaps to save on material use, the railway might decide to merge two one-way tracks into a single two-way track. Eventually, this transitions back to the two one-way tracks. Intuitively, it should be possible for only one direction to be in use on the two way track at a time.

We can also offer to model the splitting of tracks in our model and prove their safety.

- split (one-way rail becomes two, one-way rails)
- merge (two, one-way rails become one, one-way rail)

Splitting and merging tracks is necessary or else you'll end up just having a single rail track for the whole line. Creating new tracks by splitting one is one way to get more lines.

In the world of trains there are likely many additional ways to break down tracks piece by piece, but for our study these are the situations that we hope to handle.

5.2 Preconditions

There are some aspects that are common among all of our different types of track systems.

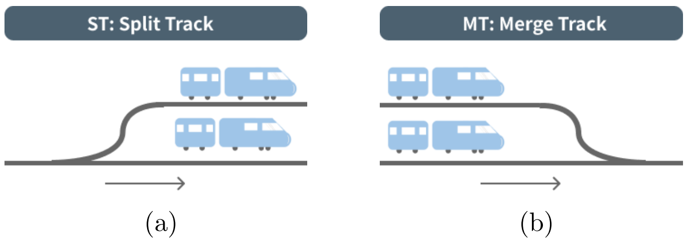


Figure 4: Merge Track Segments

First, that the trains have a maximum acceleration of $A > 0$ and maximum braking of $B > 0$. When braking, our brakes cannot turn the train around. So a train that starts in a direction will always either be stopped or end up still traveling in the same direction. In the real world of course it is possible to put a subway in reverse, but we choose not to model it because we have no reason to be moving backwards.

Additionally, train systems should start off safe before running, or else there is little the controller can do to prevent an accident. Though, what is considered a safe precondition somewhat varies between different types of track. Recall also that our controlled train will only have access to other train's positions.

It would be ideal to enforce mandatory behavior between trains, unexpected delays such as emergency stops, or faulty driving, can never guarantee us the best behavior.

5.3 Delay Modeling

We have out assumption that the maximum delay that can be created at any given station s for any given train t in the system is some $delayMax$. We are interested in proving some limit on delay time, given that bound, as well as our overall track preconditions.

Note that the ideal case for a delay for any given train t_i at a station of interest s is that it does not propagate to the train behind it, t_{i-1}. We therefore want to show given some preconditions, that:

1. Train t_{i-1} experiences no delay stopping at the station of interest s.
2. Train t_{i-1} and t_i regain those preconditions before train t_{i-1} reaches the station of interest s

Together, this can prove that the system can always "recover" from a given delay given those preconditions. We then can treat those preconditions as an invariant applied to a larger loop, to show no delay propagation can occur at that station s.

We are therefore interested in modeling two trains in the following scenrio, one follower $t_{follower}$ and one lead t_{lead}, such that given the lead starts at the station of interest s and experiences up to $delayMax$, that $t_{follower}$ has velocity $targetVel$, and some initial distance d exists between the lead t_{lead} and follower $t_{follower}$.

Our preconditions, which is also our delay invariant are the following:

Consider an event-trigger model; in this model we are able to simulate trains encountering multiple successive stations in a loop using discrete assignment to update station positions. Given our delay model, we can simulate the trains stopping at all stations.

Delay Recovery Invariant

Given some train t_{lead}, $t_{follower}$, and pos_{lead}, $pos_{follower}$, vel_{lead}, $vel_{follower}$, we will require the following preconditions on

- $buffer$: distance between $pos_{follower}$, pos_{lead}
- $targetVel$: maximum speed of follower
- $maxVel$: maximum speed of lead

to ensure delay recovery at the station of interest.

$$\text{buffer} \geq \text{maxDelay} * \text{targetVel}$$

$$\frac{\text{buffer} + \text{targetVel}^2}{(2 * B)} \geq (1/2) * A * (\frac{B}{(\text{targetVel})^2})$$

$$\text{maxVel} \geq \frac{(\text{A} * \text{targetVel})}{B}$$

We also need preconditions ensuring the safety of the initial position of the follower;

$$pos_{follower} + buffer + \frac{targetVel^2}{(2 * B)} <= station$$

5.4 Two Train Physics: ODE Evolution

With trains and time triggered control, ODE evolution does not become very complex because trains can be imagined as traveling on one dimensional tracks. Their position is along a line and their velocity and acceleration are moving either forward or backward. With two trains, Train A and Train B, starting time $t = 0$, and a max evolution time T, we would have the ODE

ODE: Physics of Two Trains

$$\{\text{posF}' = \text{velF}, \text{velF}' = \text{accF}, \text{posL}' = \text{velL}, \text{velL}' = \text{accL}, \text{t}'{=}1 \ \& \ \text{t} \leq \text{T} \ \& \ \text{velA} \geq 0 \ \& \ \text{velB} \geq 0 \ \}$$

The actual motion being pretty basic, safety and efficiency comes from choosing the right acceleration for the trains before evolving the ODE. Trains that do not need to slow down on the track should not be forced to and trains of course should not accelerate into unsafe territory.

Preserving Delay Invariant

$$\text{Delay Invariant} \rightarrow [\textit{Delay HP}]((\text{Delay Invariant} \mid \text{posL} = \text{station}) \; \& \; \text{posF} \leq \text{posL})$$

5.5 Proofs

5.5.1 Simple Segments

The one lane, one-way track only has to deal with a single scenario for safety. A lead train and a follower train going in the same direction. Adding more trains to the system does not make it any more complex, as a follower of one train can be the leader of another train and so on. While the lead train is allowed to choose whatever acceleration they like (likely due to some unbeknownst train or station further up), it is the follower train's responsibility to not rear end the lead train and to maintain a safe buffer. The minimum safest buffer between two trains is the distance that the following train would take to come to a complete stop. If the lead train was not moving and the following train breaches this space, then a collision will likely occur. The exception being that the lead train starts moving again and returns the buffer to its original size. Lead train movements are never guaranteed though so it cannot be counted upon. The distance to brake comes from our kinematic equation.

Kinematic Equation

$$v_{final}^2 = v_{initial}^2 + 2aD$$

We want v_{final} to be a complete stop, and our acceleration a is $-B$. This would give us our braking distance.

Minimum Braking Distance

$$D = v_{initial}^2 / 2B$$

The follower train at its best only periodically knows the position of the lead train, so it should never put itself in a position in which it is too late to avoid collision. The train speeds up only if it knows that doing so will mean it can still brake in time. Also, the train can simply maintain its speed if that provides enough buffer as well. With the information that it has available, the follower train can do this check by testing

Acceleration Safety Test

$$\text{posF} + \frac{1}{2}AT^2 + \text{velF} + \frac{(velF+AT)^2}{2B} \leq \text{posL} - \text{buffer}$$

posF being the position of our following train, and T being the longest time delay that we can experience in our controller. Simply, we are calculating the distance traveled by accelerating for time T, and adding it to the distance it would take to brake from this new distance. Failing this condition means the controller should not choose to accelerate. Instead, it can brake, or do a similar test for maintained velocity.

For a one lane, two-way track, safety closes a lot of the variability. No pair of trains going in opposite directions on a one-lane track can be considered safe, so that possibility must be ruled out. When a train is on the two-way track, it is just taking the form of a one-way track for that moment, until the train leaves and another train starts. So the proof would be basically identical with a more explicit restriction that all trains are traveling in the same direction. In our one-way track we just considered the positive direction to be the only option.

When a train approaches one of its station stops, it should be considering its leader train or the station. Whichever one is closer to the train helps determine the train's control decisions. The location of both the lead train and the station are available to the train, so the train behaves the same as before except its safe braking distance is the minimum of the distance to the train or the station. It's not exactly unsafe for the train to miss the station, but it would be pretty inconvenient for the passengers.

5.5.2 Transitions

Proving the transitions segments for safety, as the trains can be potentially traveling in opposite directions, can provide what would intuitively seem to be a more complicated case to handle. However, the model can be designed with certain assumptions that make the safety of the operation simpler to draw out. In our model, the train on the two way track (Train B) has the freedom to choose whatever acceleration it wants, given it is between our set max braking and acceleration capability. Thus, we are designing the controller for the train that intends to enter the single lane track (Train A). The safety here depends on that both trains cannot exist simultaneously on the one-lane track. As they are moving in opposite directions, progressing any further would result in a disastrous head-on collision. Train A should then only enter the two-way track once Train B has left it. Otherwise, Train A should come to a halt before the transition and wait for its turn.

There are additional efficiency considerations, such that Train A should only need to brake if it will cross paths with Train B on the one way segment. This level of efficiency would only be possible if Train A has information about Train B's speed and acceleration. As Train A only gets periodically updated about Train B, we can choose to accelerate only when we are far enough to brake in time for the transition or if Train B has left the one-lane area. This actually handles both of our transition scenarios, where the track is becoming a two-way track and where it is becoming two one-way tracks. The train attempting to enter the one lane will always be Train A, while the train on the one-way is always Train B. Something to note is that the position of Train B can be set to be whatever distance behind the train that we do not want Train A to cross paths with. Additionally, there could very well be multiple “Train B’s” on the track, and we should only care about the passage of the final train. Buffers can also be significantly reduced because there is no reason to wait long after Train B has left the one way.

Some shortfalls do emerge however when attempting to model train traffic in this way. Having split the transitions into their two different ends does not describe the situation in which two trains are approaching the single lane at the same time. Neither has entered, so neither train gets automatic priority. Furthermore, as each train is not aware of other trains’ velocities and accelerations, but only their positions, it not possible to ensure that a collision would not occur within the one lane track without simply coming to a halt outside of the zone. Help is needed in the form of traffic signals to let the train know it is safe to enter.

5.5.3 Rail Splits and Merges

When splitting a rail to become two separate lanes, a train’s controller still reduces to our original simple segment controller because the logic is behind which train we choose to follow. There are two scenarios if we know the present positions of all our trains. If there are multiple trains on the single lane before the split, then we are in the same scenario as the single lane from before. Choosing the closest train as our leader leads to needing the same proof. If there are no trains in the single lane, then we should only care about trains that are on the lane we intend to split towards. The closest train there becomes our leader, and all the other trains are irrelevant to us. We’ve only used information that is readily available to the train to organize this strategy.

Two one-way tracks merging into a single track provides a more difficult scenario. If two trains are heading towards the merge point and are expected to arrive at the same time, which train should slow down to allow the other to pass? If our controller always yields to the other merging lane, and all our trains use the same controller, then our system will get locked up with no trains merging. Somehow, there has to be communication with the other train approaching to indicate who will be merging first.

An additional issue that emerged in our approach is that it is difficult for us to avoid dealing with just a single other train. In reality, we encountered a situation where it would be very useful to have differential dynamic logic (QdL) available to us in the current version of KeyMaeraX. The merging of train traffic into a single lane would map very well to the demonstrated examples of cars switching lanes safely. In the QdL syntax, where we additionally have quantified ODEs and quantified assignents, we could describe merging as

Merging of Train Traffic in QdL

$$[\forall T(n := \text{new Train}; \forall i\ a(i) := ctrl;\ \{\forall i,\ x(i)' = v(i), v(i)' = a(i))^*]$$
$$\forall T, \forall i, j(i \neq j, i << j)$$

This says that for all of our tracks (any one lane track could eventually have another track merged with it), all of our trains are safe, even if another train has merged in. Each train i

has its control set and evolved in our ODE. Our control for each train is the same as that of the simple segment following, with appropriate separation between trains to allow for merging. This would also mean that we are considering the newly created lane to actually be a continuation of one of the older lanes, with the other lane feeding into this primary lane. The primary difference here is that, though the entry of cars on a highway lane can normally happen at any point along the highway, but in the case of railways, these merges happen at fixed points. Still, given that the trains provide a safe enough buffer between themselves and are always prepared for a train to enter the lane, it shouldn't be an issue that this happens at a specific point on the track.

5.5.4 Simple Event Triggering

For simple event triggering, we have a single train continuing from signal to signal. As a simple train, it is traveling at a constant velocity to a first station, to a second station. Primarily, this event trigger example serves as a test for us to detect that, after passing a signal, our train will be able to continue onward to reach the next one without issue.

5.5.5 Event Triggering with Following

We prove that it is possible to model two trains, one following the other safely, using an event trigger model instead of a time trigger model.

Our original intention was to use this to prove ETCS Level-1 control systems safe on our tracks, but a fundamental issue is that to prove a more complex model we would need access to the the continuous value of the other trains' position.

5.5.6 Delay Recovery

We were not able to fully prove delay recovery.

To prove delay recovery is possible at a given stations, we need to prove our delay recovery invariants are preserved by the time the follower train safely reaches the station, behind the lead. This means at the next station, the follower train will not be delayed by the lead train if another delay occurs, eliminating delay propagation.

Our models' preconditions can be justified as follows;

Our buffer allows the follower train to travel $maxDelay$ time before braking at the station, without hitting the lead;

$$buffer >= maxDelay * targetVel$$

Our constants $targetVel$, acceleration A, braking B, and buffer allow the lead train to accelerate to regain its' original buffer from the follower by the time the following train comes to a stop at the station. The time for the follower to brake is less that or equal to the time it takes for the lead to regain its' buffer, as follows;

$$\frac{buffer + targetVel^2}{(2 * B)} >= (1/2) * A * (\frac{B}{(targetVel)^2})$$

Finally, the *maxVel* is large enough to allow the lead to accelerate for the time the follower brakes,

$$\frac{targetVel}{B}$$

.

$$maxVel >= \frac{(A * targetVel)}{B}$$

Though we were unable to prove these relationships with KemmaeraX, we are convinced they ensure safety given the *maxDelay*, where safety is:

1. The lead is in front of the follower at all times
2. The lead and follower stop at the station
3. The lead and follower are always more than breaking distance apart, unless the lead has left the station (the follower does not have to break early).

6 Discussion

In this paper we have modeled several track scenarios that often appear in real world subway systems. Though tracks come in many shapes and forms, the motion from track to track remains the same, and takes on a relatively simple ODE. These simpler governing laws of our trains' motion have in turn produced controllers that themselves do not rely on complex physical conditions. Most of the complexity we believed, was found in choosing what train to follow. If it could be narrowed down to one train (or station), then basically control was the same. Aside from a few modifications as to what safety was, be it not physically colliding with another train or not being on the same one-way track as a train, control was not affected. Having practically the same control strategy across different scenarios is pretty convenient and saves a lot of confusion in proofs.

However, unfortunately there are oftentimes when situations cannot be narrowed down to a lead train, follow train scenario for an efficient outcome. These are more difficult to model. What was tough though was that, it was not any increased difficulty in the motion of trains, this fortunately stayed the same. Instead, some scenarios carried with them sub-cases that required the train to have additional knowledge than what was initially offered. Specifically, this occurred when a train sought to merge into another track. When attempting to merge into a track traveling in the same direction, it is easy for a train to stay safe. Never start moving. This can be observed with cars when a driver gets too nervous to enter a busy highway, or when their driveway comes out on a main road. No progress can be made by the vehicle seeking to enter, and there can be a lot of uncertainties. When coming into traffic, there has to be space to come in, and the vehicle should be moving fast enough such that traffic behind does not collide soon after entry.

With cars perhaps, their drivers can see each other, and, relative to trains, they are much easier to accelerate or decelerate with the situation. Subways are worse off, and with usually no visual information within the dark tunnels, they must rely on periodic updates for instructions and locations. Even if a train was granted knowledge of a single train, to merge, it would have to choose the final train that comes by, and wait for it to pass. This could be grossly inefficient because it abandons any opportunity to merge within gaps between trains beforehand. KeyMaeraX does not offer to us the ability to practice safety between an arbitrary number of trains, though we discussed the possibility of QdL.

We also spoke of the option of using ground signals to direct train movement. This would also assist our merging issue, as a system that has more total knowledge of all trains would be able to give directives to trains about their speed and whether it is safe to move forward. We began to explore this possibility, and it required to think with event triggered control rather than the time triggered controllers that we had been previously designing. Rather than waiting for some maximum time delay T, trains can receive new movement authority at fixed distances. Though there would have to be some additional guarantees, such that there is only one subway between two signals at a time.

Finally, when considering proving limits on delay propagation using an invariant preserved at every station, we found a fundamental flaw. Delay propagation can be prevented by allowing the lead train to travel at a larger speed to recover a buffer distance after its' delay. However, this means the follower train has a limited speed, which negatively impacts the other efficiency measures we originally considered. A slower max speed for the follower especially means the time of travel will be much longer for that following train to the station of interest.

In the future, it would likely be more worthwhile to explore the event triggered models that we were able to touch upon in this paper. With a more efficient master controller that can prevent most bad behavior, our trains could possibly be more free to think less (by just following new directives at every balise), and produce an overall more efficient railway. Of course, designing the protocols to assign these movement authorities to each train is undoubtedly difficult to get right.

One goal that we had hoped to achieve but were not able to was designing our track efficiency interface. With this program, the user could select their number of trains, following distance, track layout and receive data on the expected efficiency of the overall system. This application would also have been useful for demonstrating that the smaller pieces of track that we have described in this paper can actually be stitched together to form a railway.

Some complications arose in our previous goals before we started working on this additional task. For example, working on the need for signaling in the more complicated series of transitions and merging tracks took up some of the late stages of research that ideally would be spent on this goal. However, this change in priority we believe was necessary because the application would have been flawed from the start otherwise.

7 Conclusion

We discovered that the information provided by a ETCS Level-2 closely aligned with that needed for efficient controllers for station stops and track transitions. We also found that proving definite limits on delays in trains implies control decisions that negatively impact other measures of efficiency, namely throughput and time of travel.

Trains, although individually are simple to track, can quickly come together to form a pretty daunting system. By breaking down their tracks into smaller sub-pieces, we hoped to somewhat reduce this complexity into manageable portions. By proving these sub-problems to be safe, it becomes easier to once again image larger scale subway lines, from New York to Tokyo. It is incredibly important to get such things right, as millions depend on the ensured safety and efficiency each day of their lives.

8 Per Partner

Equal work was performed by both project members.

9 Project Files

The simple train segments proof for two train following is kept as `2-Train-Following.kyx`.

The simple segments when a train approaches a stop is kept as `2-Train-Station.kyx`.

The transition between one-way and two-way segments is kept as `transitions.kyx`.

Additionally, our event triggered control files for Simple Event Triggering, Event Triggering with Following, and Delay Recoevery are kept as `Train-Event-Trigger-Simple.kyx`, `Train-Event-Trigger-Following.kyx`, and `Delay-Recovery.kyx` respectively.

References

[NH04] Andrew Nash and Daniel Huerlimann. "Railway Simulation Using Open Track". In: *Computers in Railways IX* (2004).

[Pla10] André Platzer. "Quantified Differential Dynamic Logic for Distributed Hybrid Systems". In: *CSL*. Ed. by Anuj Dawar and Helmut Veith. Vol. 6247. LNCS. Springer, 2010, pp. 469–483. ISBN: 978-3-642-15204-7. DOI: `10.1007/978-3-642-15205-4_36`.

[Boh+18] Brandon Bohrer et al. "CoasterX: A Case Study in Component-Driven Hybrid Systems Proof Automation". In: *IFAC-PapersOnLine* (2018). Analysis and Design of Hybrid Systems ADHS.

[Gor18] Aaron Gordon. *'The Trains Are Slower Because They Slowed the Trains Down'*. Mar. 2018. URL: `https://www.villagevoice.com/2018/03/13/the-trains-are-slower-because-they-slowed-the-trains-down/`.

[Pea18] Adam Pearce. *How 2 M.T.A. Decisions Pushed the Subway Into Crisis*. May 2018. URL: `https://www.nytimes.com/interactive/2018/05/09/nyregion/subway-crisis-mta-decisions-signals-rules.html`.

[con19] Wikipedia contributors. "European Train Control System". In: *Wikipedia, The Free Encyclopedia* (2019).

[Sch19] Robert Schrader. *The World's Busiest Subway Systems*. July 2019. URL: `https://www.tripsavvy.com/the-worlds-busiest-subway-systems-4146031`.