DIMACS
Center for Discrete Mathematics &
Theoretical Computer Science

DIMACS EDUCATIONAL MODULE SERIES

MODULE 04-2
Modeling Traffic Jams with Cellular Automata
Date prepared: September 9, 2004

Stephanie Edwards
University of Dayton
Dayton, OH 45469-2316
email: sedwards@udayton.edu

Mark Ginn
Appalachian State University
Boone, NC 28608-2092
email: ginnmc@appstate.edu

John Harris
Furman University
Greenville, SC 29613
email: john.harris@furman.edu

Gregory Rhoads
Appalachian State University
Boone, NC 28608-2092
email: rhoadsgs@appstate.edu

DIMACS Center, CoRE Bldg., Rutgers University, 96 Frelinghuysen Road, Piscataway, NJ 08854-8018
TEL: 732-445-5928 • FAX: 732-445-5932 • EMAIL: center@dimacs.rutgers.edu
Web: http://dimacs.rutgers.edu/

Founded as a National Science Foundation Science and Technology Center and a Joint Project of Rutgers University, Princeton University, AT&T Labs - Research, Bell Labs, NEC Laboratories America and Telcordia Technologies with affiliated members Avaya Labs, HP Labs, IBM Research, Microsoft Research.

Module Description Information

- **Title:**

  Modelling Traffic Jams with Cellular Automata

- **Authors:**

  Stephanie Edwards, University of Dayton
  Mark Ginn, Appalachian State University
  John Harris, Furman University
  Gregory Rhoads, Appalachian State University

- **Abstract:**

  The module touches on several topics of mathematical interest. *Mathematical modeling* in general is discussed briefly at the start, and then some details regarding traffic models are given. *Cellular automata* (CA), both deterministic and probabilistic, are introduced as a method for modeling traffic flow. The software package *Mirek's Cellebration* (MCell) is presented as a way of handling the long-run forecasting of different CAs.

- **Informal Description:**

  The concept of cellular automata is motivated by easy-to-follow examples. While much of the material in the module concerns simple deterministic CAs, there is a section at the end on probabilistic CAs. Instructors could spend as much or as little time as desired on each type. MCell is a software package that is available for free download, and students will enjoy trying out its many interesting features. Regarding traffic, students can easily experiment with initial densities and other features that affect the way traffic flows. There are plenty of exercises throughout the module that give students the opportunity to solidify the concepts for themselves.

- **Target Audience:**

  This module is designed for advanced high school students and lower level college students. It would be a good module for a liberal arts mathematics class.

- **Prerequisites:**

  This module assumes some familiarity with computers and downloading software from the internet. It assumes that the students have had high school algebra.

- **Mathematical Field:**

  Probability, Discrete Mathematics, Cellular Automata

- **Applications Areas:**

  This module uses cellular automata and technology to model traffic jams.

- **Mathematics Subject Classification:**

  Primary: 37B15
  Secondary: 60K30

- **Contact Information:**

Stephanie Edwards
University of Dayton
Dayton, OH 45469-2316
email: sedwards@udayton.edu

Mark Ginn
Appalachian State University
Boone, NC, 28608-2092
email: ginnmc@appstate.edu

John Harris
Furman University
Greeneville, SC, 29613
email: john.harris@furman.edu

Gregory Rhoads
Appalachian State University
Boone, NC, 28608-2092
email: rhoadsgs@appstate.edu

- **Other DIMACS modules related to this module:**

  04-1: Probability and Chip Firing Games

Contents

ABSTRACT

The module touches on several topics of mathematical interest. *Mathematical modeling* in general is discussed briefly at the start, and then some details regarding traffic models are given. *Cellular automata* (CA), both deterministic and probabilistic, are introduced as a method for modeling traffic flow. The software package *Mirek's Cellebration* (MCell) is presented as a way of handling the long-run forecasting of different CAs.

1 Introduction

What comes to mind when you hear the term "traffic jam?" Congestion, brake lights, aggravation, road rage, horn blowing — the list goes on and on. Traffic jams are indeed unpleasant.

What causes traffic jams? Accidents, construction work, lane closings, and incompetent drivers are often to blame. But why is it that some jams just seem to appear out of nowhere? In this module we will use mathematics to help us to understand — that is, to model — this highway phenomenon.

Before beginning, let us first discuss mathematical models in general. A mathematical model is a type of mathematical construction that is used to simulate real world phenomena. The mathematical model can then be analyzed in the hope that the information obtained from the model will give some information about the real world phenomena. Mathematical models are used in a wide variety of situations, from weather forecasting and financial markets to epidemiology, medicine, and traffic.

Mathematical models can be used for many different purposes. Often they are used to make predictions about the future, as is the case in weather and economic forecasts. Sometimes they are used not to make predictions but rather to try to understand the phenomena being modelled. This is often the case in epidemiology where analyzing data about the number of people that have a disease at a given time can give some information about how the disease is spread. While many people have made mathematical models of traffic flow for forecasting purposes, our model will be of the second type. We will try to build a model to help us understand how traffic jams form and then maybe what can be done to help alleviate them.

A mathematical model may be as simple as a single equation or a system of equations or something much more complex. Typically, the more complex the model the better it simulates the real world situation being modelled. However complex models tend to be more difficult to analyze so simpler models are used instead. Often there are trade-offs to be made between complexity and accuracy. In our situation, we will be trying to come up with the simplest model we can that exhibits the desired phenomena.

1.1 Modeling Traffic

Several different mathematical models for traffic have emerged in recent years. Physicists and mathematicians alike have applied numerous methods to the study of traffic flow [2]. One model relates traffic to liquid. Think about how traffic flow looks from an airplane — at times, the traffic down below almost looks like fluid flowing down the highway. Kinetic theories, Newtonian mechanics, and differential equations are sophisticated mathematical elements of other investigations. An August, 1999 *Washington Post* article [7] described results of recent traffic research in the following way.

> Scientists have identified "phase changes" in traffic, similar to the sudden transitions that occur when steam turns to water or water to ice. Understanding the timing and dynamics of phase changes in traffic, like those in nature, poses a challenge for physicists.
> ...
>
> *Phase 1.* When traffic is light, motorists drive much as they like, moving at the speed they want and changing lanes easily. Motorists are comparable to steam particles with great freedom of movement.
>
> *Phase 2.* As the road becomes crowded, motorists suddenly lose much of their freedom and are forced to drive as part of the overall traffic stream, moving at the

speed of the general flow and often unable to change lanes. This phase, similar to water, has been called "synchronized" flow.

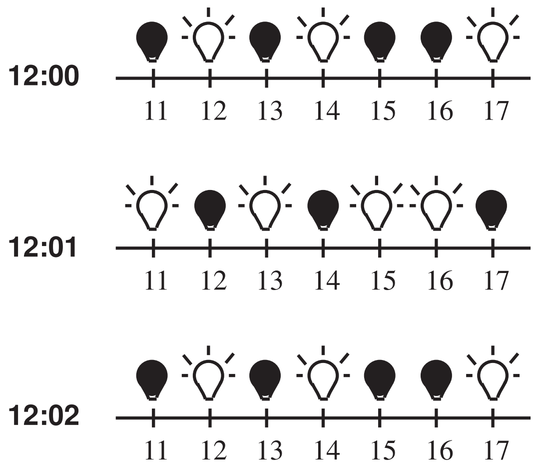
Phase 3. In heavy congestion, traffic is stop-and-go. Like water freezing into ice, the motorists are stuck in place.

In 2001 Larry Gray and David Griffeath [3] developed the first simple, discrete model in which each of the three phases mentioned in the *Post* article can be observed. We will investigate their model in this module. Their model uses one dimensional cellular automata. We will study cellular automata in the next section.

1.2 One Dimensional Cellular Automata

Let's begin this section by looking at some examples.

Example 1.1. Think about a number line which extends infinitely in both directions, and imagine that there is a little light bulb situated on the top of every whole number. Here is a small piece of the number line.



Light Bulbs

At 12:00 noon, some of the light bulbs are *on* and some are *off*. At 12:01, the lights have switched states. Again at 12:02, the lights have switched again. Do you see the pattern developing? Can you predict how the lights will look at 12:03?

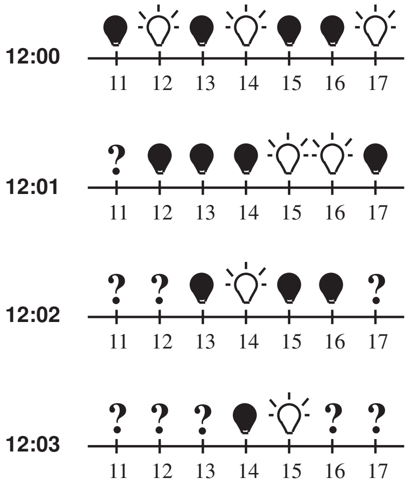
In this example, the lights are really following two simple rules:

- **Rule 1:** If a light bulb is *on* during the present time unit, then it will switch to *off* one time unit later.
- **Rule 2:** If a light bulb is *off* during the present time unit, then it will switch to *on* one time unit later.

Note: our time units are represented by minutes in this example.

These two rules can be altered to form other interesting sequences of lights. The next example illustrates this idea.

Example 1.2. Consider the following figure. Again, there are light bulbs on top of each number along the infinite number line. We are shown only a small piece of the line.



Light bulbs turning *on* and *off*.

Notice that with each passing minute, some lights switch states, and some do not. Why is this? Can you detect the pattern?

Again, there are just two simple rules that govern the action of the lights in this example. Notice how Rule 2 has a bit of a new look.

- **Rule 1:** If a light bulb is *on* during the present time unit, then it will switch to *off* one time unit later.
- **Rule 2:** If a light bulb is *off* during the present time unit, then it will switch to *on* one time unit later *only if* exactly one of the two adjacent lights is presently *on*. Otherwise, the light stays *off*.

Let's examine this particular example more closely. At 12:00, the lights at 12, 14, and 17 were *on*. So, according to Rule 1, at 12:01, each of these lights switched to *off*. The light at 13 was *off* at 12:00, and *both* of its adjacent lights (neighbors), 12 and 14, were *on* at that time. Thus according to Rule 2, 13 stayed *off* at 12:01. (The only way that a light which is *off* can switch to *on* is if exactly one of the two adjacent lights (neighbors) is *on*). The light at 15 was also *off* at 12:00, and exactly *one* of its neighbor lights, namely 14, was *on*. Therefore, according to Rule 2, the light at 15 switches *on* at 12:01.

So why the question marks? Take for example the light at 11, which is *off* at 12:00. Since we cannot see the light at 10, there is no way for us to know how many of 11's neighbor lights are *on*. For this reason, we don't know how the light at 11 will look at 12:01 — hence the question mark. The other question marks in the figure arise from similar reasons. See if you can figure out why they are there.

There is an interesting difference to note between these two examples. In Example 1.1, the future status of a particular light bulb depended only on the current status of *that* light bulb. In Example 1.2, however, a light bulb's future status depends not only on its own current status, but also on the current status of light bulbs near it.

Example 1.3. In this example, we will use 0's and 1's to denote *off* light bulbs and *on* light bulbs respectively. We will use the same rule as we used in Example 1.2 but with a different configuration of *on* and *off* light bulbs.

$t=0$	0	1	0	1	1	0	0	0
$t=1$								
$t=2$								
$t=3$								

Notice how we don't know where on the number line these light bulbs are sitting - but we actually don't care! Now, we can use the rule to fill in the rest of the table.

$t=0$	0	1	0	1	1	0	0	0
$t=1$	?	0	0	0	0	1	0	?
$t=2$	?	?	0	0	1	0	?	?
$t=3$	?	?	?	1	0	?	?	?

Example 1.4. In this example, we again use 0's and 1's to denote *off* light bulbs and *on* light bulbs respectively. We will use the same rule as we used in Example 1.3 but the only difference is the series of dots before the initial 0 and after the last 0. These dots mean that the 0's continue infinitely to the left and to the right (i.e. no more 1's).

$t=0$	$\cdots$ 0	1	0	1	1	0	0	0 $\cdots$
$t=1$								
$t=2$								
$t=3$								

Now, we can use the rule to fill in the rest of the table. The dots (or knowing that 0's continue infinitely to the left and the right) allows us to completely fill in the table.

$t=0$	$\cdots$ 0	1	0	1	1	0	0	0 $\cdots$
$t=1$	1	0	0	0	0	1	0	0
$t=2$	0	1	0	0	1	0	1	0
$t=3$	1	0	1	1	0	0	0	1

Example 1.5. Again, in this example, we will use 0's and 1's to denote *off* light bulbs and *on* light bulbs respectively. Our rules are the following:

- **Rule 1:** If a light bulb is *on* during the present minute, then it will switch to *off* in the next minute *only if* its right neighbor is *off* during the present minute.
- **Rule 2:** If a light bulb is *off* during the present minute, then it will switch to *on* in the next minute *only if* its left neighbor is *on* during the present minute.

Now, consider the following initial configuration of *on* and *off* light bulbs.

$t=0$	0	1	1	1	0	1	0	0	0	1	1	1	1	0	0

After the first time step, our table will look like the following.

$t=0$	0	1	1	1	0	1	0	0	0	1	1	1	1	0	0
$t=1$	?	1	1	0	1	0	1	0	0	1	1	1	0	1	0

And now we see what happens at time steps 2 and 3.

$t = 0$	0	1	1	1	0	1	0	0	0	1	1	1	1	0	0
$t = 1$	?	1	1	0	1	0	1	0	0	1	1	1	0	1	0
$t = 2$	?	1	0	1	0	1	0	1	0	1	1	0	1	0	1
$t = 3$	?	0	1	0	1	0	1	0	1	1	0	1	0	1	?

These examples are illustrations of a mathematical construction called **cellular automata** (or CA for short). A CA can be thought of as an intricate collection of identical objects, each of which is said to be in some particular state (*on* or *off*, for example). As distinct time periods pass, the objects (or *cells*), may change from state to state according to a particular set of rules. These rules and the list of possible states are the same for each object.

These examples are very simple examples of a cellular automata. There are several aspects of these CAs that can be varied to form other CAs. Here are some examples.

- The set of possible states could be different. Instead of two (*off* and *on*), there could be three (say, *off*, *dim*, and *bright*), or more.
- The rule can vary in terms of the neighboring cells' status. For instance, the 0's might change to 1's if one *or* two neighboring (adjacent) cells have 1's in them.
- The rule could be different in that the relevant neighborhood might be larger. In the example above, it was just the status of the two closest neighbors that affected the center cell. It very well could be that this "range" of cells is bigger.
- The playing field might be different. Instead of the one dimensional number line, grids of higher dimensions could be used. The most famous two dimensional cellular automata is John Conway's "Game of Life".

For more information about cellular automata, we refer the reader to S. Wolfram's book [9].

1.3 Exercises

As in the examples in this section, $\cdots 0$ means that 0's continue infinitely to the left and $0 \cdots$ means that 0's continue infinitely to the right. If no dots are present, it is unknown what is happening to the left and/or the right.

(1) Consider the following rule: An *off* light turns *on* if it has at least one *on* neighbor. An *on* light bulb goes *off* if it has exactly 2 *on* neighbors. The following represents an initial configuration (at $t = 0$). Fill in the boxes for $t = 1, 2, 3$. Use 1 to denote an *on* light and 0 to denote an *off* light.

$t = 0$	0	0	0	1	1	0	0	0
$t = 1$								
$t = 2$								
$t = 3$								

(2) Using the same rule as in Exercise (1), fill in the boxes for $t = 1, 2, 3$. Use 1 to denote an *on* light and 0 to denote an *off* light. The dots that appear next to the zeros mean that 0's continue to the left and to the right (i.e. no more 1's).

$t=0$	$\cdots 0$	0	0	1	1	0	0	$0\cdots$
$t=1$								
$t=2$								
$t=3$								

(3) Using the same rule as in Exercise (1), fill in the boxes for $t = 1, 2, 3$. Use 1 to denote an *on* light and 0 to denote an *off* light. The dots that appear next to the zeros mean that 0's continue to the left and to the right (i.e. no more 1's).

$t=0$	$\cdots 0$	0	0	1	1	0	0	0	1	0	0	$0\cdots$
$t=1$												
$t=2$												
$t=3$												

(4) Using the rule from Example 1.2, fill in the boxes for $t = 1, 2, 3, 4, 5, 6, 7$. Use 1 to denote an *on* light and 0 to denote an *off* light. The dots that appear next to the zeros mean that 0's continue to the left and to the right (i.e. no more 1's).

$t=0$	$\cdots 0$	0	0	0	0	0	0	1	0	0	0	0	0	0	$0\cdots$
$t=1$															
$t=2$															
$t=3$															
$t=4$															
$t=5$															
$t=6$															
$t=7$															

Note: These rules, together with this initial configuration, give rise to a CA called "Pascal's Triangle". Can you see why?

(5) Use the rule from Example 1.2 with the following exception: If a light is *on* and both of its neighbors are *off*, the light remains *on*. Fill in the boxes for $t = 1, 2, 3, 4, 5, 6, 7$. Use 1 to denote an *on* light and 0 to denote an *off* light. The dots that appear next to the zeros mean that 0's continue to the left and to the right (i.e. no more 1's).

$t=0$	$\cdots 0$	0	0	0	0	0	0	1	0	0	0	0	0	0	$0\cdots$
$t=1$															
$t=2$															
$t=3$															
$t=4$															
$t=5$															
$t=6$															
$t=7$															

How does this CA differ from the one in the previous exercise?

(6) Can traffic lights and/or digital clocks be thought of as cellular automata? Explain!

2 MCell

In Section 1 we calculated how certain initial configurations behave under a given rule for a few time periods. It is easy to see how difficult it would be to calculate what happens "in the long run," or as time increases. Fortunately, there is an amazing software program available called Mirek's Cellebration, or *MCell*, for short, created by Mirek Wojtowicz [8]. It can be downloaded for free from http://psoup.math.wisc.edu/mcell/. You will need internet access to download the program and once it is installed, you will not need the internet. In this section we will introduce the reader to some of the features of MCell.

We will use MCell to explore Example 1.2. The rules used in the example were as follows:

- **Rule 1:** If a light bulb is *on* during the present time unit, then it will switch to *off* one time unit later.
- **Rule 2:** If a light bulb is *off* during the present time unit, then it will switch to *on* one time unit later *only if* exactly one of the two adjacent lights is presently *on*. Otherwise, the light stays *off*.

We could represent these rules in another way. Let us begin by exploring rule 1: If the light bulb is *on* at the present time unit then it will switch to *off* one time unit later. Suppose a light bulb is *on* and both of its neighbors are *off*. Using 0's and 1's as before we can denote this situation in the following way:

left neighbor	light bulb	right neighbor
0	1	0

Or simply: 010

If exactly one neighbor light is *on*, the situation would be: 110 or 011. And if both neighbor lights are *on* we have 111. In each of these situations, the middle light (the light of interest) will go *off*. So we can denote this occurrence in the following way:

present time unit				next time unit		
0	1	0	→		0	

Then the other scenarios are

present time unit				next time unit		
0	1	1	→		0	
1	1	0	→		0	
1	1	1	→		0	

We can extend this idea to include rule 2 - If a light bulb is *off* during the present time, then it will switch to *on* in the next time *only if* exactly one of the two adjacent lights is presently *on*. Otherwise, the light stays *off*. We can write both of the rules in one table in the following way.

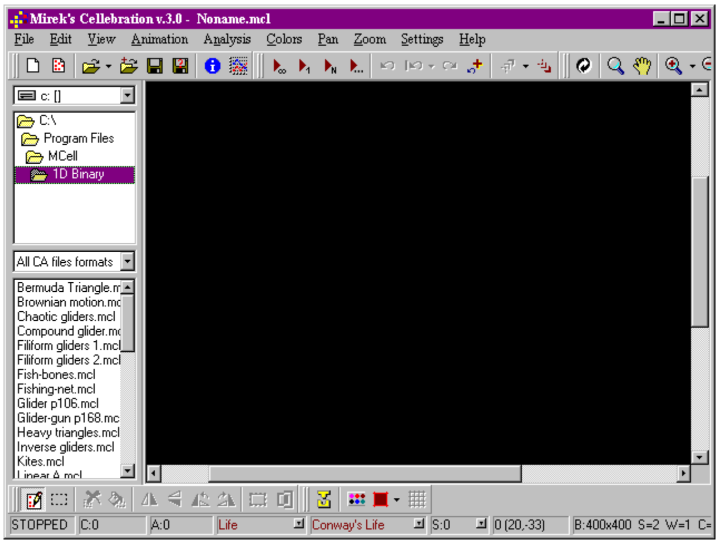
initial time unit	next time unit
010	0
011	0
110	0
111	0
000	0
001	1
100	1
101	0

This table shows the cell in question and each of its two neighbors. We can see that all of the *on* lights go *off* and the *off* lights come *on* only when exactly one of its neighbors is *on*. This is the way our computer software represents the rule. Recall, we mentioned in the exercises in the previous section, that these rules were called "Pascal's Triangle".

2.1 To find this rule in MCell:

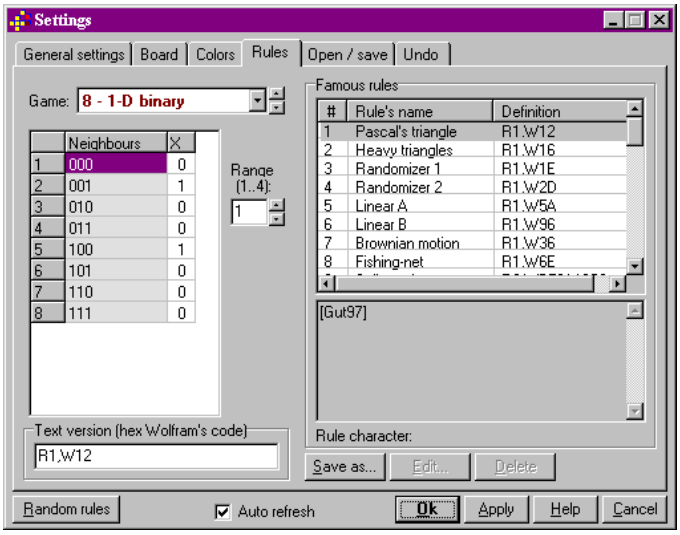
Recall, MCell is available to download for free from http://psoup.math.wisc.edu/mcell/. Once installed, it does not require internet access to run.

- Open up MCell
- You will see a list of folders in the upper left-hand corner of the screen. Click on the MCell folder and then open the folder called "1D Binary."



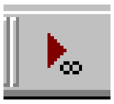
MCell Screen

- Below the list of folders, there is now a list of *.mcl files - open the one called "Pascal's Triangle.mcl."
- Go to the 'Rules' menu (along the top of the screen) and open 'Rules - setup'



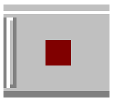
Mcell - Setting - Rules Screen

- On the left-hand side of the dialog box will be the rules, in a format similar to the table we just created above. Click OK after you've looked at the rule. It is possible to change the rules here.
- Press the "continuous play" button and see MCell run the rule! If you want to see one step at a time, press the "slow play" button which is located immediately to the right of the "continuous play" button.



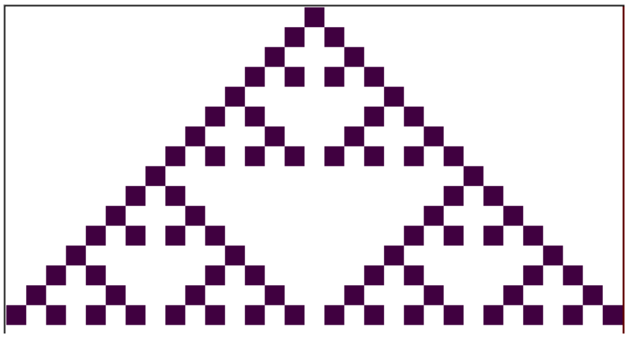
Continuous Play Button

- You can enlarge and shrink the viewing area by using the magnifying glass buttons (+ to enlarge, - to shrink). The button on the far right is a "Shrink to fit" button which picks a nice size for you.
- To stop MCell from drawing, press the "stop" button (the square).



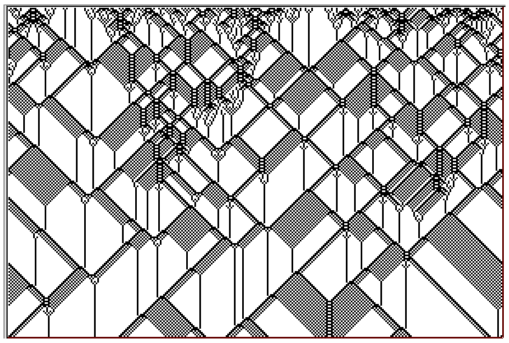
Stop Button

Example 2.1. Using MCell, 1D Binary, try out Pascal's Triangle and see if you get the following picture. (Use "Shrink to fit".) Note - you may have colors other than black and white in your screen - it is OK!



MCell - Pascal's Triangle

Example 2.2. Using MCell, 1D Binary, try out Kites.mcl and see if you get the following picture. (Use "Shrink to fit".)



MCell - Kites

2.2 Exercises

Note: once you have run a rule, to reset the screen, click on that rule again.

(1) Run the rule "SolitonsA.mcl". Look at the rules and change the first rule so that you have the light turning *on* instead of staying *off*. Does the picture look different now? Describe.

(2) Run the rule "SolitonsA.mcl". Look at the rules and change the fourth rule so that you have the light turning *off* instead of staying *on*. Does the picture look different now? Describe.

(3) Run the rule "SolitonsD1.mcl". Look at the rules and change the first rule so that you have the light turning *on* instead of staying *off*. Does the picture look different now? Describe.

(4) Run the rule "SolitonsD1.mcl". Look at the rules and change the fifth rule so that you have the light turning *off* instead of staying *on*. Does the picture look different now? Describe.

(5) Run the rule "Brownian motion.mcl". Look at the rules. These rules can also be stated as: If a light has both neighbors *off*, then the light does not change. Otherwise, the light changes. Change the rule to one that can be stated as: If a light has both neighbors *off*, then the light turns *on*. Write down the rule in the form of a table. Run the rule and describe the differences.

(6) Run the rule “Brownian motion.mcl”. Look at the rules. These rules can also be stated as: If a light has both neighbors *off*, then the light does not change. Otherwise, the light changes. Change the rule to one that can be stated as: If a light has both neighbors *off*, then the light turns *off*. Write down the rule in the form of a table. Run the rule and describe the differences.

3 The Slow Lane: Deterministic Traffic Model

Back in Section 1, we saw that

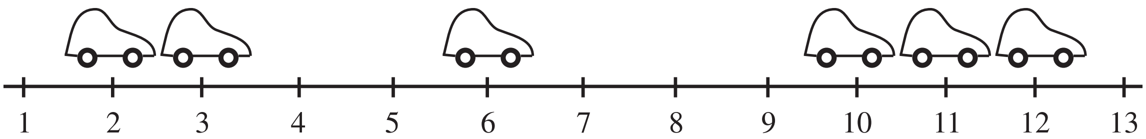
- Cellular automata are collections of identical objects having one of a finite number of states and a set of rules for determining how the state of an object changes from one moment to the next.
- An example of a cellular automata is a sequence of light bulbs which can have one of two states *on* or *off*.

One of the goals of modelling is to find a mathematical representation of a real-life situation that we wish to study. The model must also be simple enough to be easily analyzed. No one will argue that our light bulb example is simple, and in this section, we show that we can use the same ideas to model traffic flow on a one-lane highway. We will study the model built by Gray and Griffeath [3]. There are other models of traffic out there (see [1], [4], [5], [6]).

Our one-lane highway will not include on-ramps, off-ramps, or other ways for cars to enter or exit the road. This assumption may seem unrealistic, but our goal is to find the simplest “interesting” model which contains traffic jams. We will also assume our highway is infinitely long.

Let us represent our highway by the infinite real number line with the integers denoting the possible locations of the cars. At each integer, we will assign a value of either 0 or 1. Assigning a value of 1 will indicate a car occupies that location, and a 0 will indicate the location is empty. Notice the similarities between the highway example and the light bulb example. The light bulbs correspond to the locations on the highway and each has two possible states: *on/off* for the light bulbs and *car/no-car* for the highway example.

Example 3.1. This is an example of a possible configuration of a highway.



Cars

We will assume cars are located in locations 2, 3, 6, 10, 11, and 12 while locations 1, 4, 5, 7, 8, 9, 13, 14, and 15 are empty. We will represent our highway with the following table (note: the picture only goes up to location 13, however, our table representation goes up to location 15):

location	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$t = 0$	0	1	1	0	0	1	0	0	0	1	1	1	0	0	0

As seen in the picture, each car is **traveling to the right (towards higher numbers)**. From here on out, we assume that the cars will be traveling to the right.

Being infinitely long, our highway is difficult to represent. We will get around this by using a loop to represent the highway and have the cars move continuously around the loop. Thus, in the previous example, position 15 and 1 are adjacent and the 6 cars represented will move within the 15 positions. MCell allows one to specify the length of the loop - by default it is 600 positions.

3.1 "Move if you can"

We need to decide how our cars will move. What goes through your mind when you are deciding whether or not to press on the accelerator and move ahead? The distance between you and the next car? The tempo of the music on the radio? How late you are?

There are many factors that determine your driving habits but we want to keep our model simple. Therefore, we establish one rule:

> **"Move if you can" rule:** A car will move to the next (to the right) location if and only if the next location is empty (has no car in it).

Let us see what our rule implies about the cars in Example 3.1 above. Remember that locations 15 and 1 are adjacent — so using the "Move if you can" model, a car in cell 15 will move to cell 1 if there is no car in cell 1. Here is a table representing the flow of traffic for time $t = 0, 1, 2, 3, 4$. Note that the car in position 15 at time $t = 3$ moves to position 1 at time $t = 4$.

location	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$t = 0$	0	1	1	0	0	1	0	0	0	1	1	1	0	0	0
$t = 1$	0	1	0	1	0	0	1	0	0	1	1	0	1	0	0
$t = 2$	1	0	1	0	1	0	0	1	0	1	0	1	0	1	0
$t = 3$	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1
$t = 4$	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0

Notice that Example 1.5 is exactly the "Move if you can" model applied to light bulbs.

Now we have a model of the highway set up and would like to see what happens to cars as time proceeds. Specifically, we are interested in seeing if traffic flows freely or if it "jams" up. We need first to define these terms.

Definition 3.2. For the "Move if you can" model, if $n \geq 2$, we will define a **jam of size** n to be n consecutive 1's with a 0 in the location immediately before and after the sequence of 1's.

A jam will be a group of cars in adjacent locations. In Example 3.1, notice that initially there is a jam of size 2 in locations 2-3 and a jam of size 3 in locations 10-12. In the "Move if you can" model, the lead car in a jam will automatically move ahead at the next time step while the other cars in the jam remain stationary. In the course of driving, a specific car may be a member of many different jams. For example, it could encounter the back end of a jam while driving along, have to wait for the cars ahead to move out, proceed driving again only to encounter the back of a new jam and start the process over again.

These initial jams may have been caused by an accident, a police officer with radar, an animal running across the road, or many other possible reasons. We are not interested in why the jams occurs but in how they dissipate and if they reform.

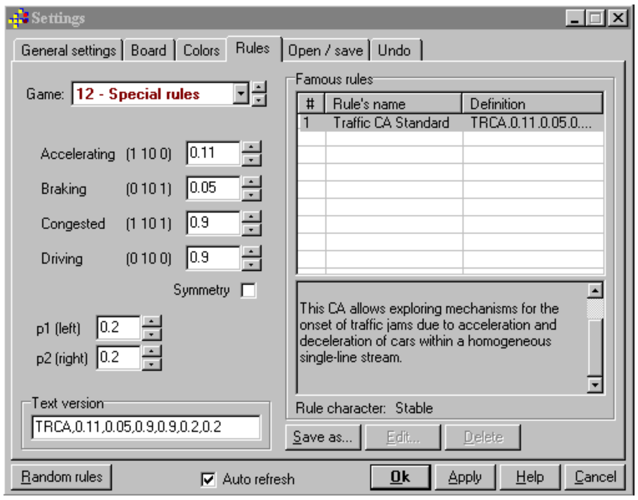
Looking back at table above, we see that the initial jam at locations 2-3 was gone at $t = 1$ and the jam at locations 10-12 was completely gone at $t = 2$. In fact, at $t = 2$ every car has an open space in front of it and proceeds forward at $t = 3$. This situation where every car continues to move forward will happen at every future time step; the cars are flowing freely from one location to the next. (You will be asked to explain why this occurs in the exercises.) This jam-free behavior will be given a special name.

Definition 3.3. A configuration for the "Move if you can" model is said to be in **free flow** if every car has an open space in front of it at the end of a time step.

3.2 Modeling Traffic Using MCell

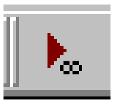
One question we have been trying to answer is, "When do we get jams?" It takes SO long to do each step and this makes it difficult to see what will happen in the long run - will a jam occur? Fortunately, MCell helps us with this. We will now begin a brief introduction to MCell and talk about some of its features.

- Open up MCell.
- You will see a list of folders in the upper left-hand corner of the screen Click on the MCell folder and then open the folder called "Special Rules."
- Below the list of folders, there is now a list of *.mcl files - open the one called "Traffic CA.mcl."
- Go to the 'Color' menu (along the top of the screen) and click on the 'color bar'. On the left side of the color bar window are left/right buttons with '1/25' underneath the buttons. Press on the left button until it reads '1/2' underneath the buttons. The will allow you to use 2 states - on or off, occupied or empty.
- Go to the 'Rules' menu (along the top of the screen) and open 'Rules setup'
- You will see the words 'Accelerating', 'Braking', Congested', and 'Driving'. The numbers in the white box represent the rules. We will talk about their exact meaning in Section 4.



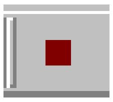
MCell - Traffic Rules

- Press the "Play" button and see MCell run the rule!



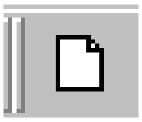
MCell - Play button

- You can enlarge and shrink the viewing area by using the magnifying glass buttons (+ to enlarge, - to shrink) The button on the far right is a “Shrink to fit” button which picks a nice size for you.
- To stop MCell from drawing, press the “stop” button (the square).



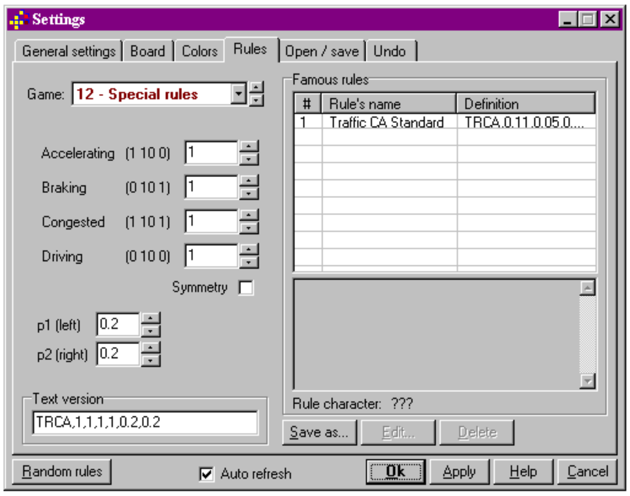
MCell - Stop button

- You can clear the screen and put your own initial conditions in there. Press the “new screen” button and then click on the spots where you would like to put cars. Remember, only put cars on the first row! It will help immensely if you enlarge the viewing area.



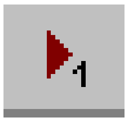
MCell - New screen button

Now, the “Rules...” screen looks very different from the ones we have used up to now. The difference is due to the fact that MCell allows the user to insert various probabilities into this model. The probabilities relate to different possible driving methods, and we will talk more about them in Section 4. For now, let us set each of the four probabilities to 1 (which is actually the defining feature of the “Move if you can” model).



MCell - Traffic Rules screen with Probabilities set at 1

MCell will allow us to look at MANY cars at a time, and that is certainly helpful. Next to the play button is another play button with a 1 next to it. Clicking this button once will show one time period passing. We'll call this button the **slow play button**. If you enlarge the viewing area many many times, you will be able to see each distinct car. Clicking successively will allow you to see each car over several distinct time periods. This is MUCH faster than doing it by hand!

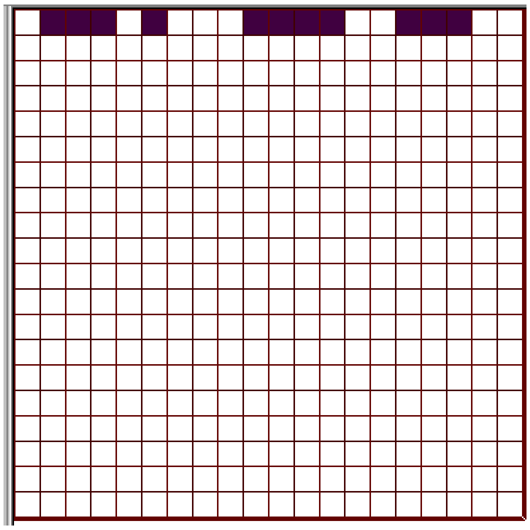


MCell - Slow play button

Example 3.4. Think back to Example 1.5. As we mentioned before, the rules we used for the lights turning *on* and *off* are the same rules for the "Move if you can" model. In this example, we will use MCell to look at how the traffic moves through multiple time periods. Recall the initial conditions:

$t = 0$	0	1	1	1	0	1	0	0	0	1	1	1	1	0	0	1	1	1	0	0

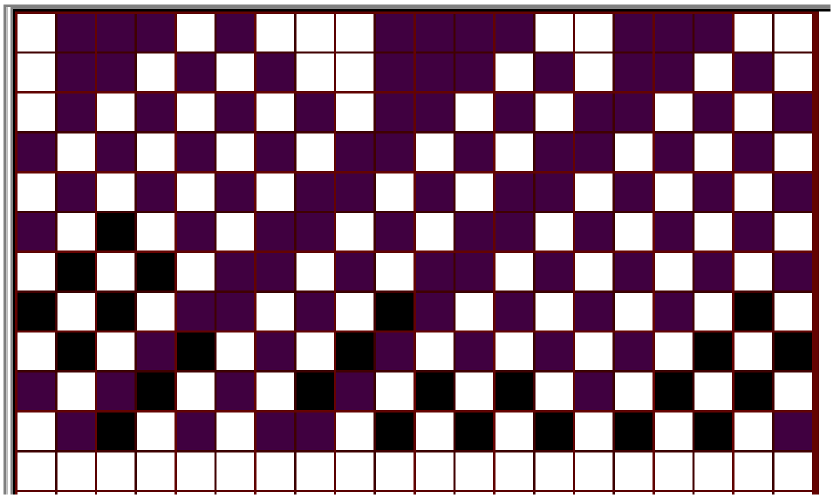
Since we have only 20 locations, we will need to adjust the board size for MCell. To do this we go to "Settings - Board..." and the Board Size will be in the upper left hand corner. Click on "Other" and set the board size to be 20 by 20. (Mcell forces the size to be a square and the lengths/widths must be multiples of 20.) When we put this into MCell, we get:



MCell - "Move if you can" initial configuration

For clarity, let us reiterate a few points. First, the black squares in the first row of the above figure represent cars, and the white squares represent spaces without cars. Second, *cars move from left to right*. Third, as a car moves off of the right side of the screen, it reappears on the left side of the screen.

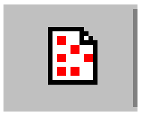
In this example, the initial configuration contains 3 distinct "jams" of cars. Reading from left to right, there are jams of size 3, 4, and 3 respectively.
Using the "slow play" button we get the following picture after 9 time periods. Notice that the number of cars on each line remains the same. This is because when a car exits the screen on the right hand side, it reappears on the left hand side of the screen.



MCell - "Move if you can" after 9 time periods

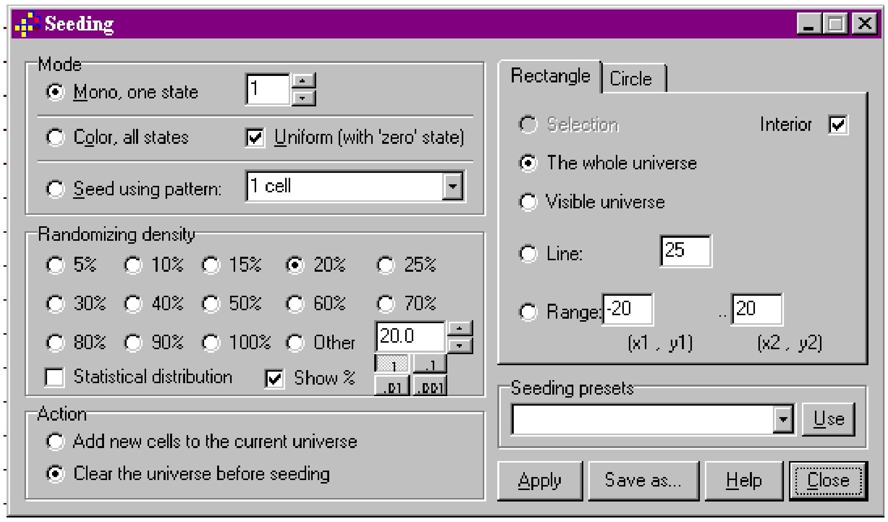
Look at the big picture here. At the beginning there are three jams, and over time they seem to (more or less) work themselves out. Why is it that the jams themselves look like they are moving backward? Maybe we can answer that question after looking at another example.

Example 3.5. In this example, we let MCell itself randomly choose the initial configuration of cars. To do this, simply press the "random" key. (Our board size is 600 by 500.)



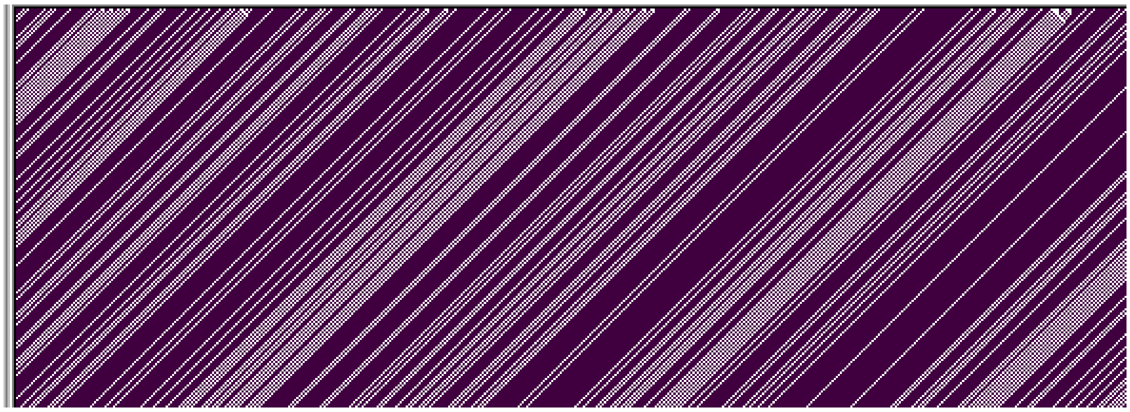
MCell - Random button

Before setting up the initial configuration, MCell asks us to choose the percentage of spots that will be occupied by cars.



MCell - How to chose a certain density

For our example, we will choose 80% (think of this as the first row of spots being 80% filled with cars). Make sure that you have clicked the button to the left of "Mono, one state". Press the Apply and then Close buttons and the screen will have the cars it. Now play the rule and look at the long term behavior.



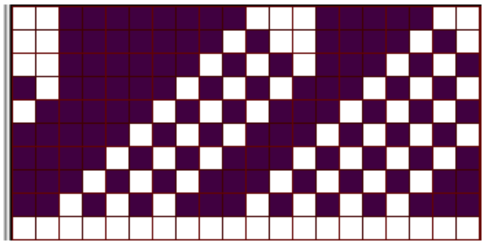
MCell - long term behavior of the rule above

So what is this figure showing? Each horizontal slice represents one time period, and the solid black sections are the "jams." Notice again that it appears the jams are moving backwards!

To understand what is happening, let us take a smaller version of this example and concentrate on a single jam. We make our board 20 by 20. To do this, go to the 'Settings' menu and open 'Settings - board". Then make the board 20 x 20. We will begin with the following initial conditions:

$t = 0$	0	0	1	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	0	0

We have a jam of size 8 and a jam of size 5. As we look what happens over time, we again see that the jams tend to move to the left. One way to think of why this occurs is that the car in the front of the jam leaves because there is an empty space ahead of it. The rest of the jam does not move, and another car catches up with the jam and then becomes part of the jam (and thus creating a jam that is the same size as the one during the previous time period).

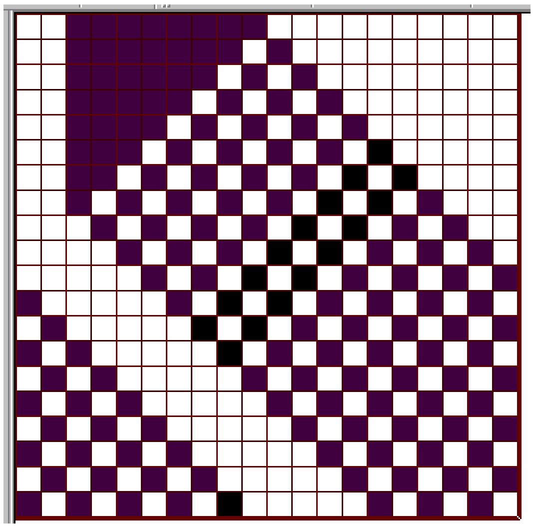


MCell - Traffic Jams

Example 3.6. Now that we have seen a bit about the behavior of jams, let us consider another situation. Again we will be using a 20 by 20 board and the "Move if you can" rule. But this time our initial configuration will have fewer cars in it:

$t = 0$	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0

We have a jam of length 8. As we run MCell we see that the jams dissolves and we are in free flow.



What makes Example 3.2 and 3.6 different is the number of cars in the initial configuration (the *density*). In the next section we will talk about the role on the initial densities.

3.3 Densities in the move-if-you-can model

In the last section, we looked at the long range behavior of the "move if you can" model under different initial conditions. In Example 3.1, we had 15 locations and 6 cars and noticed that no matter where the 6 cars were initially placed, eventually all cars would be in free flow. Then we used MCell and started with 20 locations and 11 cars and found that again, eventually the cars would go into free flow. In Example 3.5 we changed the initial number of cars so that 80% of the locations were filled. Even after many iterations of MCell, traffic jams existed at every time step. We would like to determine how the number of cars affects the long range behavior of the traffic pattern. First, let us give a definition for this initial number of cars.

Definition 3.7. If $\rho \in [0, 1]$, we say the initial configuration has **density** ρ if the percentage of locations filled with carsin the initial configuration is ρ.

For example, if $\rho = \frac{1}{2}$, then exactly half of the locations will have a car assigned to it. In Example 3.1, we had 15 locations and 6 cars so $\rho = 6/15 = 40\%$. If the density is less than or equal to $\rho = \frac{1}{2}$ (or 50%), the traffic will eventually be in free flow, otherwise there will be at least one jam.

3.4 Exercises

(1) The following table was built in the Example 3.1. (Recall that the right end of the table wraps back around to the left end!)

location		1	2	3	4	5	6	7	8	9	10
state	$t = 0$	0	1	1	0	0	1	0	1	0	0
state	$t = 1$	0	1	0	1	0	0	1	0	1	0

Determine the status of the 10 locations for $t = 2$ and $t = 3$ in the "Move if you can" model. Determine the density of the initial configuration.

location		1	2	3	4	5	6	7	8	9	10
state	$t = 0$	0	1	1	0	0	1	0	1	0	0
state	$t = 1$	0	1	0	1	0	0	1	0	1	0
state	$t = 2$										
state	$t = 3$										

(2) Determine the status of the 10 locations for $t = 1$, $t = 2$ and $t = 3$ in the "Move if you can" model. Determine the density of the initial configuration.

location		1	2	3	4	5	6	7	8	9	10
state	$t = 0$	0	1	1	1	1	1	0	1	1	0
state	$t = 1$										
state	$t = 2$										
state	$t = 3$										

(3) Consider the following initial conditions. Without using MCell, fill in the rest of the table using the “Move if you can” Model. Determine the density of the initial configuration.

$t = 0$	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	0	0	1	1	0
$t = 1$																				
$t = 2$																				
$t = 3$																				
$t = 4$																				
$t = 5$																				

(4) Consider the initial conditions from the previous model. Using MCell, fill in the rest of the table using the “Move if you can” rules. (Use a 20 by 20 board.) Determine the density of the initial configuration.

$t = 0$	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	0	0	1	1	0
$t = 1$																				
$t = 2$																				
$t = 3$																				
$t = 4$																				
$t = 5$																				
$t = 6$																				
$t = 7$																				
$t = 8$																				
$t = 9$																				
$t = 10$																				
$t = 11$																				
$t = 12$																				
$t = 13$																				
$t = 14$																				
$t = 15$																				
$t = 16$																				
$t = 17$																				
$t = 18$																				
$t = 19$																				

(5) Consider the initial conditions from the previous model. Using the play button, look at the long term behavior. Write a paragraph describing what you see. Make sure to comment on jams and free flow. Is free flow ever attained?

(6) Using MCell and a 20 by 20 board, randomly place cars on the screen with a 40% density. Write down your initial configuration:

$t = 0$																				

Will you attain free flow? If so, how many time periods until it occurs.

Do this 5 more times:

$t = 0$																				

$t = 0$																				

$t = 0$																				

$t = 0$																				

$t = 0$																				

(7) Using MCell and a 20 by 20 board, randomly place cars on the screen with a 70% density. Write down your initial configuration:

$t = 0$																				

Will you attain free flow? If so, how many time periods until it occurs.

Do this 5 more times:

$t = 0$																				

$t = 0$																				

$t = 0$																				

$t = 0$																				

$t = 0$																				

(8) Explain why if a system is in free flow in the "Move if you can" model, then every car would move forward at the next time step.

(9) Explain why if a system is in free flow in the "Move if you can" model, then every car would move forward at **every future** time step. (Hence the name: free flow.)

(10) Argue that if $\rho > 50\%$, then there must be a jam in the initial configuration.

(11) The previous exercise implies that if $\rho > 50\%$, then there must be a jam at every time step. Explain why.

(12) Run MCell with various densities $\rho < 50\%$ and look at the long term behavior. Describe what you eventually see. Make a conjecture about the long term behavior of the "Move if you can" model with $\rho < 50\%$. Use a 40 by 40 board and remember to use the "shrink to fit" button.

(13) Use a table to write down all possibilities for the status of a location and its immediate neighbors, then determine the status of the location at the next time step using the "Move if you can" model.

4 The Expressway: Probabilistic Traffic Model

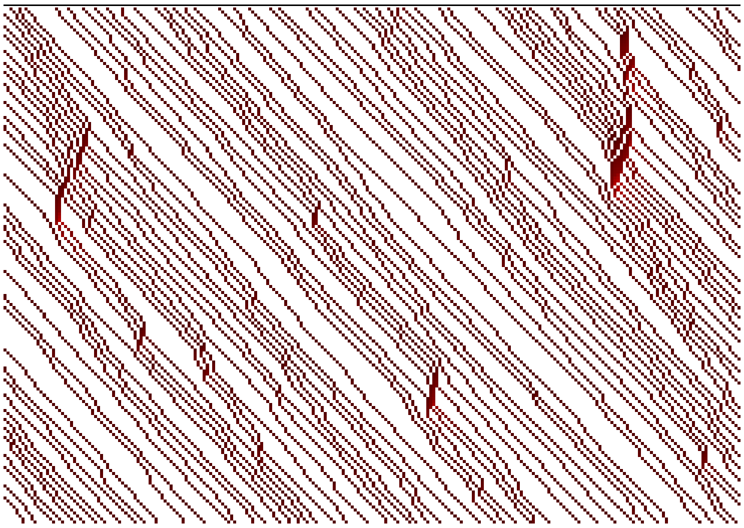
4.1 Introduction

Recall the descriptions of the three phases of traffic mentioned at the beginning of this module.

> *Phase 1.* When traffic is light, motorists drive much as they like, moving at the speed they want and changing lanes easily. Motorists are comparable to steam particles with great freedom of movement.
>
> *Phase 2.* As the road becomes crowded, motorists suddenly lose much of their freedom and are forced to drive as part of the overall traffic stream, moving at the speed of the general flow and often unable to change lanes. This phase, similar to water, has been called "synchronized" flow.
>
> *Phase 3.* In heavy congestion, traffic is stop-and-go. Like water freezing into ice, the motorists are stuck in place.

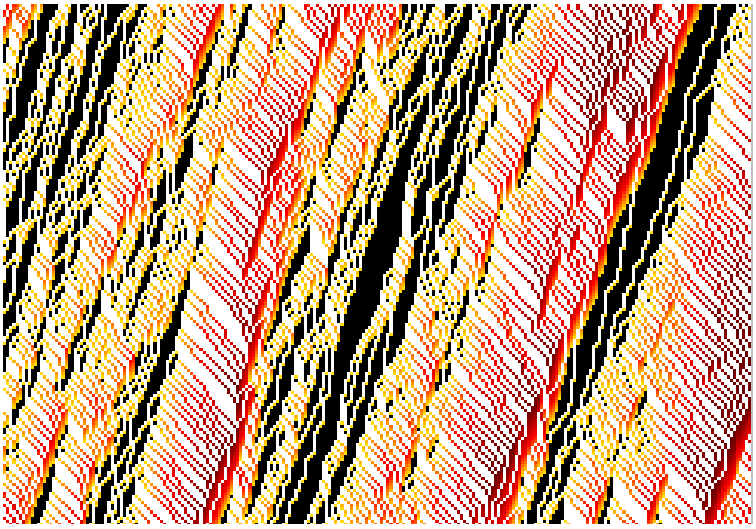
The model developed in the last section was certainly a simple model of traffic flow that we were able to analyze fairly well. However it did not demonstrate the three phases of traffic that we wished to see. Rather, we only got the first and third types of traffic, free flow and persistant traffic jams. Before we go on and develop a more complex model, let's first discuss the three phases a bit more.

Consider the first MCell screen shot shown below.



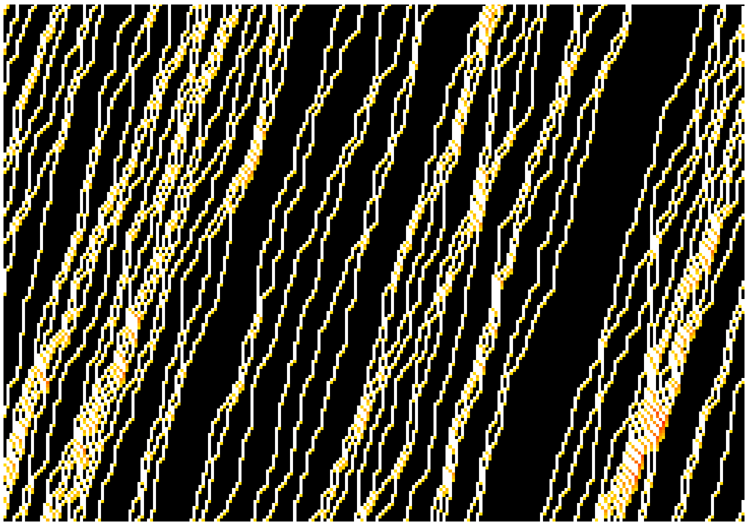
Example Free Flow

This gives a more realistic example of free flow than we saw in the "Move if you can" model. Here cars move, for the most part, independently of one another. While traffic jams do occasionally appear, they quickly dissipate and return to free flow. Notice that our definition of free flow given in Definition 3.3 does not fit this situation. A mathematically precise definition of free flow is quite technical and is beyond the scope of this module [3].



Second Phase of Traffic

The above screen shows the second phase of traffic. Here traffic is moving fairly well and there are no very long traffic jams although virtually every car spends most of its time in contact with the cars adjacent to it. While traffic jams do form in this situation, they tend to eventually go away as new ones form in a different place.



Third Phase of Traffic

Finally, we have the third phase shown in the above screen is a definite traffic jam. Long jams that move backwards along the road with only brief escapes into small pieces of free flow typify this phase.

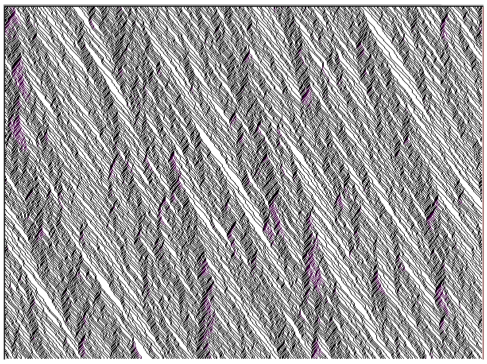
These are the three phases of traffic flow that we would like to model. We will look at various attempts to do this in the next sections.

4.2 A Simple Probabilistic Model

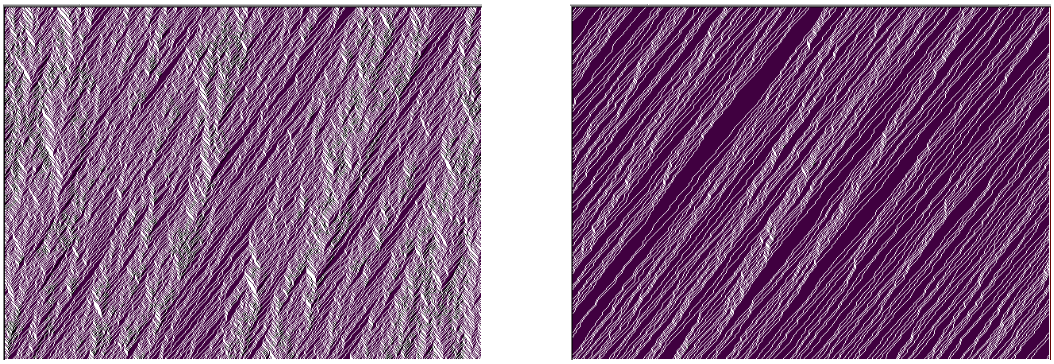
We will attempt to improve our model by adding some randomness to the process. In Section 3, our model made the assumption that all drivers who had an opening ahead of them moved forward immediately. Anyone who has ever watched a long line of cars start after being stopped at a traffic light knows that this is certainly not the case. In this section we will assume that there is some randomness as to whether a person who *can* move ahead *does* by saying that any car with a space ahead of them will move ahead with probability p. Let us first consider the case where $p = \frac{1}{2}$. In this case we can think of each driver with a space ahead of them flipping a coin at each time period. If the coin comes up heads, they move ahead one space, if it comes up tails they stay where they are for that time period. (A more realistic model might consider whether the driver is dialing their cell phone or adjusting the radio, but we will stick to flipping coins.) Now hopefully in reality drivers move ahead more often than half of the time there is an open place in front of them so we will have p greater than $\frac{1}{2}$. In this case, say with $p = .9$, we can just think of flipping a coin weighted so that it comes up heads 90% of the time and tails only 10% of the time. Hopefully it is intuitive to the reader that in this situation the cars would move more quickly down our infinite highway.

MCell also does a nice job of simulating this model. To see this, open MCell and in the Special Rules folder open the Traffic CA rules. Then, under the Settings menu, open the Settings - Rules dialog box. In this box, change the numbers next to Accelerating, Braking, Congested, and Driving to all be the same. Whatever number is placed in the boxes is the probability p that a car with a space ahead of it will move forward. Let us look at what happens with $p = .8$. Recall that to set the initial seeding you press the random key and select one of the percentages given as the percentage of available cells to be filled. Let's look at how the picture changes as we go from a low density to a high density. In the figure below we see screen shots with densities $20\%, 40\%, 60\%$, and 80%.

Density 20%



Density 40%



Density 60% Density 80%

While the third picture with 60% density seems to be the "synchronized" flow of phase 2, a longer look at this model in motion shows that the jams eventually work themselves out. Hence this is just a special case of Phase 1. In fact, a lot of research has been done on this model and it has been proven that synchronous flow never occurs [3]. As this model does not exhibit the behavior we are looking for, we will move once again to a more complex model.

4.3 The Probabilistic Traffic CA

In an attempt to further refine the probabilistic model of the last section, Gray and Griffeath [3] noticed that drivers were concerned with more than just if there was an empty space in front of them, they were also concerned about the situation further ahead of them and behind them. Hence they came up with four different scenarios that a driver with an empty space directly ahead of them can face and assigned the car a different probability of advance for each of the scenarios. These scenarios are *acceleration, braking, congestion,* and *driving*. Here is a table describing the four situations for a driver in position x:

transition type	$x-1$	x	$x+1$	$x+2$	*probability of advance*
acceleration	1	1	0	0	α
braking	0	1	0	1	β
congestion	1	1	0	1	γ
driving	0	1	0	0	δ

Let's clarify this table a bit. The *acceleration* row tells us that if an initial situation is 1100, then in the next time interval, the *second* car will advance *with probability* α. Similarly, the *braking* row tells us that if an initial situation is 0101, then in the next time interval, the second car will advance with probability β.

The four scenarios can be thought of in driving terms in the following manner:

- **acceleration** — In this situation, the car has reached the front of a jam and is accelerating off.
- **braking** — In this situation, the car is coming up behind some congestion and needs to brake.
- **congestion** — In this situation, the car has found an opening in the middle of some congestion.
- **driving** — Here the car is away from other cars and driving on its own.

It is important to note that when β, the braking probability, gets larger, the car is *less* likely to brake — β is actually the probability of moving forward in a braking situation.

This model finally seemed to be getting at what the researchers desired. The three MCell screen shots at the beginning of this section are pictures of MCell running this model with $(\alpha, \beta, \gamma, \delta)$ set to $(.5, .4, .3, .9)$ with ρ taking on the values of .2, .5, and .8 in each of the three pictures respectively. So Gray and Griffeath finally had a fairly simple model that captured the aspects of traffic flow they wanted to analyze. Now all they had to do was analyze the model to determine which values of the parameters gave rise to which of the three phases. Unfortunately, this model proved to be too complex a task. After all, there are five different parameters $\alpha, \beta, \gamma, \delta$, and ρ that can all change. Trying to determine exactly which values of which of these parameters put the traffic flow in which phase was a very complex task. So they tried to simplify the model in a way that preserved the behavior they were looking for while at the same time made it easier to analyze. They achieved this by considering the special case of *symmetric cruise control*.

A traffic CA is said to be in *cruise control* if $\delta = 1$. This is the familiar situation of cars on the open road moving at top speed. This is a nice assumption to have for the analysis as it takes one bit of randomness out of the model.

We say that a traffic CA is *symmetric* if $\gamma = \delta$. This may seem like a strange assumption to make at first. It says that we advance in traffic as fast as we do on the open road. It would certainly make sense to think that drivers would be more cautious and advance more slowly in congestion (although there are freeways in many metropolitan areas where this behavior is exhibited daily). This is a simplification that makes the mathematics much easier. The reason for this comes from considering anti-cars. An *anti-car* is simply a place in the CA that does not contain a car (a blank space). Notice that as cars move down our infinite highway to the right, anti-cars move down it to the left. In the case of a symmetric traffic CA, the probability of a car moving to the right is the same as an anti-car moving to the left. Thus the behavior of cars in a traffic CA with density ρ is the same as the behavior of anti-cars in a traffic CA with density $1 - \rho$. (Assuming that α, β, γ, and δ all remain the same.) This essentially cuts the amount of work that needs to be done in half.

Even with this simplified model, a complete analysis of traffic behavior is not known. As a matter of fact, as of the writing of this module, only the behavior on the boundaries of this model, when both α and β are set to either 0 or 1, is completely understood, and some of these cases were even hard to analyze. We will examine how this model behaves for several different sets of parameters in the exercises and see that it does, in fact, exhibit all three phases.

4.4 Exercises

(1) Recall Example 3.1 where the initial conditions were:

location	1	2	3	4	5	6	7	8	9	10	11	12	13
$t = 0$	0	1	1	0	0	1	0	0	0	1	1	1	0
$t = 1$													
$t = 2$													
$t = 3$													
$t = 4$													
$t = 5$													
$t = 6$													

Complete the table using the simple probabilistic model which allows a car to move forward if there is room, with probability $\frac{1}{2}$. Use a coin to determine if the car will move ahead.

(2) Let $\rho = 0.75$ and assume an initial configuration

location	1	2	3	4	5	6	7	8	9	10
$t = 0$	0	1	1	0	0	1	0	1	0	0
$t = 1$										
$t = 2$										
$t = 3$										

Let the probability that a car moves ahead to an open stop be 0.75. To determine if a car moves ahead to a vacant location, flip two fair coins and move if you get at least one head (the probability of getting at least 1 head is 0.75 with two fair coins). Fill in the table above.

(3) Recall Example 3.1 where the initial conditions were:

location	1	2	3	4	5	6	7	8	9	10	11	12	13
$t = 0$	0	1	1	0	0	1	0	0	0	1	1	1	0
$t = 1$													
$t = 2$													
$t = 3$													
$t = 4$													
$t = 5$													
$t = 6$													

Complete the table using the simple probabilistic model which allows a car to move forward if there is room, with probability $\frac{1}{6}$. Have the car move ahead if a 1 is rolled on a die and there is room. Otherwise the car will stay put.

(4) In the "Move if you can" model, the initial configuration in Exercises 1 and 3 resulted in free flow after 2 time steps (see Example 3.1). In paragraph form, describe your results from Exercises 1 and 3. How does the probability change the outcome? Is there free flow? If so, after how many time periods?

(5) Let's consider the Symmetric Cruise Control model with $\alpha = .6$ and $\beta = .6$. To set the model up this way in MCell, we need to open Mcell and then in the 'Rules' menu at the top open the 'Rules setup' dialog box. In this dialog box, α is the number next to Accelerating and β is the number next to Braking. Recall that since this is the symmetric cruise control, both γ and δ, the numbers next to Congestion and Driving, should be set to 1. Set these numbers and click the OK button.

   (a) Run this model with initial densities set to $30\%, 50\%$, and 70%. Describe what you see in each case and determine which of the three phases is represented.

   (b) Now look at initial densities 40% and 60%. What do you see here? Be sure to let the model run for a while so you are viewing the long term behavior.

(6) Now let's look at the Symmetric Cruise Control model with $\alpha = .2$ and $\beta = .6$

   (a) Before we begin, think about what this set of parameters has changed about our driver's driving habits. How should this model differ from the one in the previous exercise?

(b) Now set up MCell to run this rule and run it with density 40%. Describe what you see. How does this differ from what we saw in the last exercise? Once again, make sure to let the model run for a while before making your observations.

5 Concluding Remarks

Automobile traffic affects almost everyone. As the number of cars on public roads increases, learning more about the nature of traffic becomes even more beneficial. The models discussed in this module are attempts to describe traffic patterns mathematically.

The "Move if you can" model was straightforward — any car that had an open space in front of it would move into that space. In this model the traffic pattern evolved into a freely moving state — as long as there were not too many cars on the road. If the road was more than 50% covered with cars, though, then jams were always present.

In Section 4 we increased the complexity of the model a bit by incorporating some randomness. First we fixed it so that a car would move into an open space ahead of it with probability p. This variation produced both Phase 1 and Phase 3 traffic flows. We then increased the model's complexity once more by assigning probabilities to the four possible actions of a car on the road: acceleration, braking, moving in congestion, and driving. This model finally did show all three phases of traffic flow.

These models, and other similar ones, have helped researchers to learn about traffic patterns. By varying probabilities and initial road conditions, the researchers have shown that "phantom traffic jams" can just appear out of nowhere — even if all drivers are driving similarly to one another. It is interesting to note that the models described here started simple and gradually got more complex. This is typical of mathematical modeling — the more you would like to learn about a phenomenon, the more complex the model needs to be.

What else can be learned about traffic? Is there something that can be done to reduce the number of traffic jams? Is there a way to make the roads safer for drivers? These questions may be very difficult, and solving them just may require us to develop better models.

References

[1] V. Belitsky, J. Krug, E. Neves, and G. Schutz, *A cellular automaton for two-lane traffic*, Preprint (2002).

[2] D. Chowdhury, L. Santen, and A. Schadschneider, *Statistical physics of vehicular traffic and some related systems*, Physics Reports **329** (2000), 199–319.

[3] L. Gray and D. Griffeath, *The ergodic theory of traffic jams*, Journal of Statistical Physics **105** (2001), 413 – 452.

[4] H. Kozuka, Y. Matsui, and H. Kanoh, *Traffic flow simulation using cellular automata under non-equilibrium environment*, IEEE (2001), 1341–1345.

[5] K. Nagel, D. Wolf, P. Wagner, and P. Simon, *Two lane traffic rules for cellular automata: A systematic approach*, Phys. Rev. E **58** (1995), 1425–1437.

[6] D. Redelmeier and R. Tibshirani, *Why cars in the next lane seem to go faster*, Nature (September 2, 1999).

[7] A. Sipress, *Studying the ebb and flow of stop-and-go. los alamos lab using cold war tools to scrutinize traffic patterns*, Washington Post, August 5, 1999; p. A01.

[8] M. Wójtowicz, *Mirek's cellebration: a 1D and 2D cellular automata explorer*, Windows software, available for free download from http://psoup.math.wisc.edu/mcell/.

[9] S. Wolfram, *Cellular automata and complexity*, Addison-Wesley, Menlo Park, CA, 1994.