Wang, Jenny
23032736
HW #6
BML comments

The Biham-Middleton-Levine traffic model is a traffic flow model used to observe which conditions result in free flow and/or gridlocks. Four parameters were observed in this simulation: density of cars, number of steps, number of rows, and number of columns. As shown in bml_figures.pdf, varying the parameters effects traffic flow and gridlock. The most influential parameter of traffic flow is density p. By fixing r, c, and n, three densities were observed. As expected, a low density (p=0.2) did not result in gridlock and cars still had space to move. A high density resulted in staggered diagonal gridlocks across the entire grid indicating a complete traffic jam. This observation makes sense since the higher the density of cars, the lower the density of empty spaces. More cars occupy the grid leaving less room for cars to move into empty spaces. Likewise, a lower density allows for continuous traffic movement.

Increasing the number of steps resulted in more geometric patterns and lines. The highest number of steps was n=20,000 and the final lattice displayed noticeably greater pattern formation than did the final lattice for the lowest number of steps, n=1,000. The lower the number of steps, the more random the grid appears.

Unlike density and number of steps, the shape of the grid did not appear to influence traffic. For the observed density and number of steps, p = 0.35 and n = 10,000 respectively, all final states exhibited similar traffic flows with both areas of free flow and traffic jams.

From varying multiple densities, it appears that initial signs of gridlock occur around densities p=0.25 – 0.30. Free flow occurs at densities under p=0.25. Densities p=0.30-0.40 exhibit both free flow and traffic jams. Densities above p=0.45 displayed traffic jams.

The last portion of my bml_simulation R files has information on how many steps until gridlock for various densities.

Note: My bml.step() function moves red cars one spot east and then blue cars one spot north. I considered one odd timestep (reds move) and one even timestep(blues move) as one whole step. Thus, the steps.until.gridlock displays the number of moves made by both color cars until gridlock occurred. For example, if steps.until.gridlock returns 182, both red and blue cars went through 182 whole iterations. If steps.until.gridlock = n, the system never hits gridlock.