Modelowanie Systemów Dyskretnych

Lab 4 - Extended Nagel–Schreckenberg model

MODELOWANIE RUCHU NA DWUPASMOWEJ AUTOSTRADZIE

Znamy model Nagela-Schreckenberga oparty na automatach komórkowych został rozszerzony o dodatkowe pasmo ruchu. Opracowano nowy zestaw reguł zmiany stanów umożliwiający manewr zmiany pasa ruchu-wyprzedzania oraz powrót na pas przeznaczony do jazdy z mniejszą prędkością.

ZADANIA

Zaimplementuj rozszerzony model Nagela-Schreckenberga z poprzednich zajęć na podstawie poniższych zasad. Rozszerzenie można wykonać bez zmiany podstawowych właściwości modelu jednopasmowego. Poza zmianą jednej zasady. Nie należy uwzględniać zasady „losowego hamowania”, ponieważ używamy teraz deterministycznego modelu CA z okresowymi granicami.

Przygotowanie panelu:

Zmień rozmiar planszy:

- rozmiar punktu na 25 (size w klasie Board)
- rozmiar planszy na na szerszą i wąską (np. 1800x220) w 3 miejscach
    - w Program.java

      this.setSize(1800, 220);
    - w GUI.java -

      container.setSize(new Dimension(1800, 220));
    - w GUI.java -

      board = new Board(1800, 220 - buttonPanel.getHeight());

Plansza powinna mieć 6 wierszy
Dwa środkowe wiersze będą drogą, natomiast reszta będzie trawą (patrz rozpiska typów).

Należy zmodyfikować metodę initialize która dla zewnętrznych wierszy powinna zmienić typ na 5 (trawa).

Należy zmodyfikować metodę drawNetting tak aby każdy typ miał odpowiedni kolor (patrz rozpiska typów).

Rozpiska typów:
0 - droga - kolor biały
1 - samochód (max prędkość 3) - kolor żółty
2 - samochód (max prędkość 5) - kolor niebieski
3 - samochód (max prędkość 7) - kolor czerwony
5 - trawa - kolor zielony

** Prędkość początkowa samochodu (po kliknięciu) powinna wynosić tyle ile maksymalna **

** Należy pamiętać, żeby w odpowiednim miejscu ustawić wartość prędkości maksymalnej w zależności od typu pojazdu **

Dodanie opcji wyboru typu:

Podmienić metodę mouseClicked w klasie Board na poniższe:

```
public void mouseClicked(MouseEvent e) {
    int x = e.getX() / size;
    int y = e.getY() / size;
    if ((x < points.length) && (x > 0) && (y < points[x].length) && (y > 0)) {
        if(editType==0){
            points[x][y].clicked();
        }
        else {
            points[x][y].type= editType;
        }
        this.repaint();
    }
}
```

Podmienić metodę mouseDragged w klasie Board na poniższe:

```
public void mouseDragged(MouseEvent e) {
    int x = e.getX() / size;
    int y = e.getY() / size;
    if ((x < points.length) && (x > 0) && (y < points[x].length) && (y > 0)) {
        if(editType==0){
            points[x][y].clicked();
        }
        else {
            points[x][y].type= editType;
        }
        this.repaint();
    }
}
```

Dodać w klasie Point atrybut types:

```
public static Integer[] types = {0, 1, 2, 3, 5};
```

** Warto rozważyć rozbudowę sąsiedztwa poprzez dodanie atrybutu prev wskazującego na poprzedni punkt **

Podmienić metodę clicked w klasie Point na:

```
public void clicked() {
    this.type = 0;
}
```

Dodać / zmodyfikować następujący kawałek kodu w metodzie initialize w klasie GUI:

```
drawType = new JComboBox<Integer>(Point.types);
drawType.addActionListener(this);
drawType.setActionCommand("drawType");

buttonPanel.add(start);
buttonPanel.add(clear);
buttonPanel.add(drawType);
buttonPanel.add(pred);
```

Manewr wyprzedzania (3 pkt)

Dodany lewy pas będzie używany tylko do wyprzedzania. Manewr wyprzedzania wykorzystuje rozszerzone sąsiedztwo i obejmuje wolne przestrzenie za i przed pojazdem, na obu pasach. Zakładamy, że kierowca wykrywa jedynie zajętość przestrzeni w swoim sąsiedztwie. Prędkość drugiego pojazdu na autostradzie pozostaje dla niego nieznana. Aby samochód mógł wyprzedzić inny pojazd wymagana jest pusta przestrzeń przed oraz za pojazdem na lewym pasie.

$$1) V_{0,j} < V_{max}$$
$$2) D^{-}_{0,j} \geq D_{max}$$
$$3) D^{-}_{1,j} \geq D_{max}$$
$$4) D^{+}_{1,j} \geq D$$
$$\rightarrow L_{1,j+(D-1)} = V_{0,j} + 1$$

gdzie:

$V_{0,j}$ - prędkość pojazdu na prawym pasie we względnej pozycji j,

V_{max} - maksymalna prędkość pojazdu,

$D^{-}_{0,j}$ - odległość od najbliższego pojazdu jadącego z tyłu na prawym pasie, w pozycji j,

$D^{-}_{1,j}$ - odległość od najbliższego pojazdu jadącego z tyłu na lewym pasie, we względnej pozycji j,

D_{max} - odległość pokonana przez pojazd rozwijający maksymalną prędkość (na jedną iterację),

$D^{+}_{1,j}$ - odległość od najbliższego pojazdu jadącego z przodu na lewym pasie, w pozycji j,

D - odległość pokonana przez pojazd z prędkością punktową,

$L_{1,j+(D-1)}$ wartość komórki w pozycji względnej j + D 1, w następnym kroku czasowym.

Manewr powrotu (3 pkt)

Zasada jest podobna do tej stosowanej w manewrze wyprzedzania.

$$1) D^{-}_{0,j} \geq D_{max}$$
$$2) D^{-}_{1,j} \leq D_{max}$$
$$3) D^{+}_{0,j} \geq D$$
$$\rightarrow L_{1,j+(D-1)} = V_{0,j}$$

Zasady przetwarzane są w następującej kolejności:
manewr powrotu→manewr wyprzedzania→ruch (zestaw reguł NaSch), ale w każdej iteracji stosowana jest tylko jedna reguła. W większości sytuacji pojazdy po prostu poruszają się dalej tym samym pasem. Powyższy zestaw reguł jest minimalny w tym sensie, że prowadzi do realistycznego zachowania i tzw. schematu podstawowego, czyli zależność między przepływem a gęstością, jest odtworzona poprawnie.