Sistem za upravljanje saobraćajem

Članovi tima

- Vladislav Radović SV 27/2021
- Nikola Mitrović SV 18/2021

Motivacija

Saobraćaj u modernim gradovima je postao složen i teško upravljiv zbog velikog broja vozila, gužvi, nesreća i nepredviđenih situacija. Trenutni sistemi za upravljanje saobraćajem uglavnom ne koriste dovoljno podatka kako bi regulisali saobraćaj optimalno.

Cilj projekta je razvoj inteligentnog sistema zasnovanog na znanju, koji automatski analizira podatke o saobraćaju, detektuje kritične situacije i predlaže optimalne odluke za regulaciju prometa.

Pregled problema

Gradovi se suočavaju sa problemima poput gužvi, nesreća i nepredviđenih situacija, dok postojeći sistemi za regulaciju saobraćaja uglavnom ne koriste dovoljno dostupnih podataka i nisu prilagodljivi promenama u realnom vremenu.

Neophodno je razviti sistem koji prikuplja i analizira podatke iz senzora, kamera i GPS uređaja, otkriva kritične situacije i donosi optimalne odluke. Zbog velike količine podataka, sistem mora omogućiti obradu u realnom vremenu i pregled trendova kroz vreme.

Postojeća rešenja su često ograničena i teško se integrišu sa drugim sistemima, pa je cilj razviti proširiv i konfigurabilan sistem zasnovan na znanju, koji povećava automatizaciju, sigurnost i efikasnost saobraćaja.

Metodologija rada

Ulaz u sistem:

Ulazi uključuju:

- `TrafficDensity(crossroad, value)`
- `Vehicle(crossroad, speed, type)`
- `Accident(crossroad, severity)`
- `Weather(type, intensity)`
- `EmergencyVehicle(location, direction)`
- `IllegalParking(crossroad)`
- `EventDay(eventType, expectedTraffic)`
- `PedestrianDetected(crossroad)`
- `PublicTransportDelay(line, minutes)`
- `Crossroad(id, connections)`

Svaki od ovih ulaza utiče na trenutnu procenu stanja saobraćaja, a odluke donete u jednom trenutku služe kao ulaz za narednu optimizaciju semaforskih režima i rute vozila.

Baza znanja:

Baza znanja je strukturisana po kontekstima i pravila su grupisana prema vremenskim periodima i tipovima događaja.

Primer konteksta i pravila:

- **Rush hour – produži zeleno**

rule "Rush hour - extend green"

when

TrafficDensity(crossroad == $c, value > 70) // gužva veća od 70%

TimeOfDay(hour >= 7 && hour <= 9) // jutarnji špic

```
then
   insert(new TrafficAction($c, "extendGreen", 30)); // produži zeleno za 30 sekundi
end
```

- **Detekcija nesreće – blokada raskrsnice**

```
rule "Accident detected - block crossroad"
when
   Accident(crossroad == $c, severity == "high")
then
   insert(new TrafficAction($c, "blockCrossroad"));
   insert(new Notification("Nesreća na " + $c + " - raskrsnica blokirana"));
end
```

- **Prioritet za hitna vozila**

```
rule "Emergency vehicle priority"
when
   EmergencyVehicle(location == $c, direction == $d)
then
   insert(new TrafficAction($c, "allowPass", "priority"));
end
```

- **Loše vreme – produži zeleno**

```
rule "Bad weather extend green"

when

  Weather(type == "rain" || type == "snow", intensity == "medium" || intensity == "high")

then

  // Produži zeleno za sve raskrsnice zbog lošeg vremena

  insert(new TrafficAction("All", "extendGreen", 15));

  insert(new Notification("Loše vreme - produženje zelenog svetla na svim raskrsnicama"));

end
```

- **Nepropisno parkiranje – obaveštenje**

```
rule "Illegal parking notification"

when

  IllegalParking(crossroad == $c)

then

  insert(new Notification("Nepropisno parkiranje detektovano na " + $c));

end
```

- **Pešaci na prelazu – crveno za vozila**

```
rule "Pedestrian crossing - red light"

when

  PedestrianDetected(crossroad == $c)

then

  insert(new TrafficAction($c, "setRedForVehicles"));
```

```
end
```

- **Zauzetost semafora – balansiranje traka**

```
rule "Semaphore congestion - balance lanes"
when
  TrafficDensity(crossroad == $c, value > 80)
then
  insert(new TrafficAction($c, "balanceLanes"));
  insert(new Notification("Zauzetost semafora visoka na " + $c + " - balansiranje traka"));
end
```

- **Kašnjenje javnog prevoza – prioritet autobusu**

```
rule "Bus delay - priority"
when
  PublicTransportDelay(line == $line, minutes > 10)
then
  insert(new TrafficAction("All", "prioritizeBus", $line));
  insert(new Notification("Linija " + $line + " kasni - prioritet na semaforima"));
end
```

- **Masovni događaj – preusmeravanje rute**

```
rule "Mass event - reroute traffic"
```

when

EventDay(expectedTraffic > 1000, eventType == "concert" || eventType == "sports")

then

insert(new TrafficAction("All", "rerouteTraffic"));

insert(new Notification("Masovni događaj danas - preusmeravanje saobraćaja"));

end

- **Noćni režim – skraćivanje zelenog svetla**

rule "Night mode - shorten green"

when

TimeOfDay(hour >= 23 || hour <= 5)

then

insert(new TrafficAction("All", "shortenGreen", 15)); // smanji zeleno za 15 sekundi

End

- Sva pravila u sistemu koriste **forward chaining**, gde se factovi i akcije propagiraju kroz lanac pravila.
- **Nivoi ulančavanja**:
  - **Prvi nivo** – detekcija osnovnog događaja (npr. gužva, nesreća, kašnjenje autobusa).
  - **Drugi nivo** – aktivacija pravila koja kombinuju više ulaza i analiziraju uzroke (npr. blokada raskrsnice, loše vreme, masovni događaji).
  - **Treći nivo** – generisanje optimizovanih odluka i akcija (produžavanje zelenog svetla, preusmeravanje saobraćaja, prioritet hitnim vozilima i javnom prevozu).

Primer ulančavanja pravila:

1. `TrafficDensity > 80%` → insert `HeavyTraffic` fact
2. `HeavyTraffic + Accident` → insert `CriticalCongestion` fact
3. `CriticalCongestion` → insert `TrafficAction(extendGreen)` i `Notification`

`Akumulaciona funkcija:`

rule "Accident frequency in 15 min"

when

accumulate($a : Accident() over window:time(15m), $count : count($a))

then

if ($count > 3) insert(new Notification("Previše nesreća u poslednjih 15 min"));

end

Slično se može koristiti za: prosečnu brzinu vozila, broj pešaka, maksimalnu zauzetost raskrsnica.

Baza znanja se popunjava kombinacijom:

- istorijskih obrazaca gužvi;
- pravila eksperata (saobraćajnih inženjera);
- real-time senzorskih podataka;
- automatski generisanih pravila prema trendovima saobraćaja.

Backward chaining:

Ako postoji sumnja na **kritičnu gužvu** ili incident, sistem koristi istorijske podatke i trenutne senzorske ulaze da postavi dijagnozu problema:

- **Rekurzivni query – upstreamCause** (glavni za uzrok gužve)

query upstreamCause($crossroad: String, $cause: String)

// Nesreća blokira raskrsnicu

Accident(crossroad == $crossroad, $cause := "accident")

```
or
  // Hitno vozilo blokira raskrsnicu
  EmergencyVehicle(location == $crossroad, $cause := "emergencyVehicle")
or
  // Prevelika gustina saobraćaja
  TrafficDensity(crossroad == $crossroad, value > 80, $cause := "heavyTraffic")
or
  // Loše vreme
  Weather(intensity == "high", $cause := "badWeather")
or
  // Masovni događaj
  EventDay(expectedTraffic > 1000, $cause := "massEvent")
End
```

Sistem ne reaguje samo na trenutne događaje, već koristi **backward chaining** i **rekurzivne querije** da postavi dijagnozu problema i identifikuje primarni uzrok kritične gužve ili incidenta. Kada se detektuje problem, kao što je gužva, kašnjenje autobusa ili hitno vozilo na raskrsnici, sistem pokreće query `upstreamCause`, koji istražuje sve moguće uzroke: nesreće, gužvu, loše vreme, masovne događaje ili blokade od strane vozila.

Ovaj query se rekurzivno poziva za sve povezane raskrsnice i ulaze dok ne pronađe **glavni uzrok**. Na taj način se formira **stablo dijagnoze**, gde svaki čvor predstavlja potencijalni uzrok, a grane povezuju posledice sa stvarnim događajima u mreži saobraćajnica.

Primer toka:

1. Sistem detektuje kašnjenje autobusa (fact `PublicTransportDelay`).
2. Upit `whyBusDelay` aktivira `upstreamCause` za raskrsnicu gde je problem primećen.
3. Query proverava svaki potencijalni uzrok: gužvu, nesreću, hitno vozilo, loše vreme ili masovni događaj.

4. Ako neki uzrok vodi ka drugoj raskrsnici, query se rekurzivno aktivira i proširuje stablo dijagnoze.
5. Kada se primarni uzrok identifikuje, insertuju se **TrafficAction** i **Notification** objekti koji optimizuju saobraćaj: produžavanje zelenog svetla, preusmeravanje rute, prioritet hitnim vozilima ili javnom prevozu.

- ***Zašto gužva?***

```
query whyTrafficJam($crossroad: String, $cause: String)
    upstreamCause($crossroad, $cause)
end
```

- ***Zašto hitna vozila kasne?***

- ```
  query whyEmergencyDelayed($crossroad: String)
      EmergencyVehicle(location == $crossroad, direction != "free")
      upstreamCause($crossroad, $cause)
  end
  ```

- ***Zašto autobus kasni?***

- ```
  query whyBusDelay($line: String, $crossroad: String, $cause:
  String)
      PublicTransportDelay(line == $line, minutes > 10)
      upstreamCause($crossroad, $cause)
  end
  ```

- ***Da li je gužva posledica lošeg vremena?***

- ```
  query isWeatherCause($crossroad: String)
      TrafficDensity(crossroad == $crossroad, value > 80)
      Weather(intensity == "high")
  ```
  ```

```
end
```

- ***Da li je gužva posledica masovnog događaja?***

- ```
  query isEventCause($crossroad: String)
      TrafficDensity(crossroad == $crossroad, value > 80)
      EventDay(expectedTraffic > 1000)
  end
  ```

- ***Ko blokira raskrsnicu? (vozilo ili nesreća)***

- ```
  query whoBlocksCrossroad($crossroad: String, $cause: String)
      upstreamCause($crossroad, $cause)
  end
  ```

- ***Koja raskrsnica je izvor problema?***

- ```
  query sourceCrossroad($cause: String, $crossroad: String)
      upstreamCause($crossroad, $cause)
  end
  ```

- ***Da li postoji alternativna ruta slobodna?***

- ```
  query isAlternativeRouteFree($crossroad: String, $alternative:
  String)
      Crossroad(id == $crossroad, connections contains
  $alternative)
      TrafficDensity(crossroad == $alternative, value < 50)
  end
  ```

- ***Da li je pešački prelaz uzrok usporenja?***

- ```
  query isPedestrianCause($crossroad: String)
      PedestrianDetected(crossroad == $crossroad)
  ```
  ```

```
        TrafficDensity(crossroad == $crossroad, value > 70)
    end
```

-
- **Da li postoji alternativna ruta slobodna?**
- **Da li je pešački prelaz uzrok usporenja?**

Complex Event Processing (CEP):

- **Prosečna brzina u 5 minuta**
- **Najduži period gužve (30 min)**
- **Broj nesreća u 15 minuta**
- **Broj pešaka u 10 minuta**
- **Detekcija naglog rasta gužve**
- **Prosečno kašnjenje autobusa u 1h**
- **Maksimalna zauzetost raskrsnice u 20 min**
- **Broj zaustavljenih vozila u 10 min**
- **Ukupan broj obaveštenja vozačima u 1h**
- **Trend pogoršanja vremena u 30 min**

```
rule "Accident frequency in 15 min"
when
accumulate( $a : Accident()
over window:time(15m),
$count : count($a) )
then
if ($count > 3) insert(new Notification("Previše nesreća u poslednjih 15 min"));
end
```

Izveštaji i monitoring:

Sistem podržava tri vrste izveštaja:

1. **Normal izveštaj** – trenutna statistika gužvi, brzine i broja nesreća po raskrsnici i vremenu.

2. **Trend izveštaj (AtSomeTime)** – analiza saobraćaja u određenim periodima, npr. jutarnja gužva u poslednjih mesec dana.
3. **MaxPeriod izveštaj** – pronalaženje najdužih perioda kritične gužve ili nesreća za planiranje saobraćajnih intervencija.

Template-i:

Template (`rush_hour_rule.drt`):

```
rule "Rush hour - @crossroad"
when
    TrafficDensity(crossroad == "@crossroad", value > @threshold)
then
    insert(new TrafficAction("@crossroad", "extendGreen",
@extendSeconds));
end
```

Primer CSV fajla koji generiše pravila:

crossroad	threshold	extendSeconds
Crossroad 1	70	30
Crossroad 2	75	25
Crossroad 3	80	20

Primer template-a za fact

Ako imamo puno raskrsnica sa različitim gustinama saobraćaja:

Fact template (`traffic_density_template.drt`):

```
declare TrafficDensity
    crossroad: String
    value: int
```

end

CSV primer:

crossroad	value
Crossroad 1	85
Crossroad 2	60
Crossroad 3	90

Klasni dijagram: