Traffic simulation on HPC platform

Victor TIAN
Mentor: Cheng LIU and Kwai WONG
TRANSIMS

- Developed by Argonne National Laboratory
- Famous software in 2010
- NO update now
- Serial software code
- Agent-based simulation
Structure of TRANSIMS

- **SET UP**
  - Network Preparation
    - Highway Network Conversion
    - Transit Network Conversion
    - Network Editing
  - Trip Table Conversion
    - Activity Generation
  - Activity Generation
    - Census Data Conversion
    - Population Synthesis
    - Activity Generation

- **RUN**
  - Router and Router Feedback
  - Microsimulator
  - Stabilization and User Equilibrium

- **REVIEW**
  - Review and Presentation of Results
    - Summarization
    - Visualization
• Node
• entry/exit point
• signal nodes
• Link

created by manual, for analysis

Alexandrian
Output data

- Delay, for feedback
- Table, including position and time
  - Can be converted to animation
RepastHPC

- Argonne National Laboratory
- Agent-based simulation platform
- Only need to design the model
- No need for control sending/receiving message
Idea

• Each process control one area.

• Buffer area

• Each vehicle is an individual agent, move from left to right
Agent

- Each agent is a kind of vehicle
- Each has a unique ID
- Each agent has its own max speed, 2, 4, 6,
- Each agent has its own safety distance, 1, 2, 3
- Each agent know its neighbours
Algorithm

1. init
   1. read prop/config
   2. create Grid
   3. initial some agents

2. Play
   1. decide next position
   2. remove the agent outsides.
   3. after all agents decide next position, move
   4. add new agent
   5. synchronise between processes

3. Save data to file

4. Done, record
How to decide (straight)

- In each step, it will query and know its neighbours.
- It will know which car is in front and the distance.
- It will move to the position.
  - Keep safety distance.
  - As much as it can.
Decide (straight)

Assume A’s speed is 4

Keep safety distance
Go faster as much as it can
How to decide (line changing)

- After it query and know its neighbours, it will know
- Any car is in left/right front and the distance
- Any car is in left/right position
- It will move to the position
  - Keep safety distance
  - as much as it can
Decide(lane changing)

B(t2) = B(t1) + B's speed

<table>
<thead>
<tr>
<th>B(t1)</th>
<th>B(t2)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>A(t1)</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

A can change line

<table>
<thead>
<tr>
<th>B(t1)</th>
<th>B(t2)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>A(t1)</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

A cannot change line
Output data

- txt file, from t0001.txt to t3000.txt
- Each file describe the position of the agent at that time

```
id  AgentId(2, 0, 0, 0)  AT  Point[0, 50]
id  AgentId(0, 3, 1, 3)  AT  Point[0, 148]
id  AgentId(1, 3, 0, 3)  AT  Point[0, 149]
id  AgentId(1, 0, 1, 0)  AT  Point[0, 49]
id  AgentId(3, 3, 2, 3)  AT  Point[0, 151]
id  AgentId(3, 0, 3, 0)  AT  Point[0, 51]
id  AgentId(3, 3, 2, 3)  AT  Point[0, 150]
id  AgentId(2, 3, 3, 3)  AT  Point[0, 150]
id  AgentId(0, 0, 1, 0)  AT  Point[0, 48]
```
Output data
Output data
Video Demo
Future work

- 2 car move to one position
  - The agent only know others’ position at this moment
- Road -> Agent also
- Traffic signal
- Flow intersection
- more complex algorithm
Q&A