Universidade do Porto

Faculdade de Engenharia

FEUP

T02
Tráfego numa Cidade

Relatório Final

Agentes e Inteligência Artificial Distribuída
4º ano do Mestrado Integrado em Engenharia Informática e Computação
2017/2018

Elementos do Grupo:

Andreia Rodrigues – up201404691 – up201404691@fe.up.pt
Eduardo Leite – gei12068 – gei12068@fe.up.pt
Francisco Queirós – up201404326 – up201404326 @fe.up.pt

Índice

Objetivo

1.1 Descrição do cenário

No âmbito da unidade curricular de Agentes e Inteligência Artificial Distribuída o grupo propôs-se a desenvolver um programa que simula o tráfego automóvel numa cidade.

Este programa vai simular a interação entre automóveis face ao trânsito que se pode fazer ou não sentir devido à abundância de carros e aos vários semáforos espalhados pelas ruas dessa cidade. Os vários automóveis devem ser capazes de comunicar entre si o estado do trânsito onde circulam e os semáforos devem ser capazes de comunicar o seu estado a todos os automóveis existentes. O objetivo é que os automóveis evitem zonas com maior trânsito quando lhes for possível, chegando mais rápido ao seu destino.

1.2 Objetivo do trabalho

Com este trabalho, espera-se conseguir desenvolver agentes que quando estejam a funcionar em conjunto, permitam simular um ambiente fiel ao que nos foi pedido desenvolver.

Espera-se também conseguir que os agentes que representam os condutores/automóveis naveguem de forma eficaz em função da comunicação feita entre os agentes, e o seu destino.

Especificação

2.1 Identificação e caracterização dos agentes

2.1.1 Veículos

Os veículos terão em cada instante uma posição e um local aonde querem chegar.

Para concluírem o seu objetivo vão evitar usar caminhos congestionados de forma a minimizar o tempo necessário. A estratégia a usar será:

1. Determinar o caminho ótimo para chegar ao seu destino a partir da sua localização atual usando A*.
2. Seguir o caminho determinado no ponto 1.
3. Se a qualquer instante receber uma mensagem a informar sobre um troço congestionado, recalcular o A* mas ignorando a existência do troço em questão, em adição a qualquer outro troço para qual este protocolo tenha sido seguido anteriormente na vida do agente. Se ignorando este novo troço não for encontrado um caminho, mantém o caminho que já seguia.
4. Após percorrer cada troço, se o agente determinar que demorou significativamente mais tempo do que esperaria demorar (usando velocidade e comprimento do troço) informa outro agentes do tipo veículo (aleatórios retirados do conjunto de todos os agentes do tipo veículo) que aquela via se encontra congestionada. Este comportamento deve reduzir o tempo de viagem destes outros agentes.

Estes agentes também irão parar caso agentes do tipo semáforo, que estejam presentes à entrada do troço que pretendem percorrer a seguir, mandem parar.

2.1.2 Semáforos

Os semáforos são agentes que podem ter dois estados: verdes, para indicar avanço ou vermelhos, para indicar paragem. Estes agentes vão regularmente trocar o seu estado de forma a regularizar o trânsito.

Estes agente não terão estratégias complexas, apenas irão trocar o seu estado num intervalo fixo. Têm apenas o papel de condicionar os agentes do tipo veículo.

2.1.3 Rádio

A rádio é um agente que tem o papel de sinalizar aos agentes do tipo veículo sobre alguns casos de troços congestionados. A rádio faz isto de forma aleatória, tanto o troço como a frequência com que o faz. Este comportamento não vai

necessariamente reduzir o tempo de viagem dos agentes devido à sua natureza aleatória.

2.2 Protocolos de interacção

Existem duas formas dos agentes interagirem com a informação no mundo. A mais frequente é utilizando o espaço físico. Isto é feito diretamente na lógica de cada agente, acedendo diretamente ao mundo. A outra é utilizando mensagens enviadas de um agente para outro.

Esta primeira forma não dispõe de qualquer protocolo de interação. A segundo forma necessita protocolo para entender o contexto do conteúdo das mensagens. Contudo, neste projeto, não houve a necessidade de desenvolver protocolos complexos devido à falta de complexidade da informação a transmitir.

Apenas houve a necessidade de transmitir informação sobre troços dados como congestionados. Ou seja, os agentes do tipo veículo sabiam que sempre que recebiam uma mensagem ia ser para aquele fim, qual seria o formato e que não haveria necessidade de apresentar qualquer resposta.

Resumidamente, não existem protocolos que valham a pena discutir. Agentes mandam as mensagens para outros agentes como discutido anteriormente e não existem respostas.

Desenvolvimento

3.1 Plataforma e Ferramentas utilizadas

Foi utilizada a biblioteca SAJaS com o objetivo de usar simultaneamente as bibliotecas JADE e Repast. Para este efeito também foi usado o IDE do Repast Simphony e daí associadas as bibliotecas JADE e SAJaS.

A biblioteca JADE tem a função de disponibilizar uma plataforma multi-agente e com ela enviar mensagens para a comunicação entre os vários agentes.

A biblioteca Repast também é uma plataforma de agentes mas tem um ênfase em simulação de agentes num espaço físico.

Estas duas plataformas têm as suas qualidades e falhas. A biblioteca SAJaS permite facilitar o uso conjunto destas duas primeiras bibliotecas de forma a aproveitar as qualidades de ambas e diminuir o efeito das falhas das mesmas.

3.2 Estrutura da aplicação

A aplicação está contida numa package de Java chamada trafegoNumaCidade. Dentro desta package encontram-se outras duas packages, graph e streetlight que contêm código para classes de suporte para criar uma representação de grafos e para classes que representam o comportamento dos semáforos, respetivamente. Foi criada a package para os semáforos devido ao número razoável de classes necessárias.

Na package principal encontra-se tudo o resto. Os outros dois agentes (CarAgent - agente do tipo veículo; Radio - agente rádio).

TrafegoCidadeBuilder é a classe inicializadora da aplicação.

MyPoint é uma classe de suporte que define o comportamento de pontos e vetores geométricos.

TNCSpace encapsula o espaço físico da simulação Repast, fornecendo funções úteis e guardando informação sobre a posições dos agentes veículo.

Road e RoadWayIndicator são classes que servem para ajudar na representação visual do grafo no ambiente do Repast. A classe graph.Node também é utilizada para esse fim.

StaticComponent é uma classe abstrata de qual se estende para todos os elementos que não são agentes mas que queremos que sejam representados no espaço do Repast.

Map é uma classe que serve para inicializar os elementos a representar no Repast a partir de um objeto da classe Graph.

Na package streetlight encontram-se:

Streetlight é a classe que implementa o agente semáforo.

GoStreetlight e StopStreetlight que são outras classes a usar para a representação gráfica de elementos, neste caso de semáforos, o estado de avançar e de parar, respetivamente. Estas classes estendem a classe StreetlightStatic para ser mais simples declarar as estruturas de dados para guardar informação sobre estas classes.

StreetlightCluster serve para agrupar todos os semáforos presentes num nó (graph.Node) para acesso mais simples.

O diagrama de classes da aplicação está presente juntamente com este relatório. Caso contrário ocuparia muito espaço e também torna-se mais legível desta forma.

3.3 Detalhes relevantes da implementação

A aplicação é principalmente controlada pelos *ticks* presentes na simulação do Repast, em termos de tempo. *Behaviours* associados ao JADE são controlados por mecanismos do JADE, nomeadamente o que permite aos agentes do tipo veículo receber mensagens.

Criamos uma pequena linguagem para permitir uma criação simples do grafo a usar. Isto permite ser mais simples criar o grafo e faz com que não seja necessária a recompilação da aplicação.

Experiências

Fizemos duas experiências, uma para validar a funcionalidade do comportamento entre agentes do tipo veículo (avisarem-se entre si sobre troços congestionados) e que este comportamento produz valores melhores; Outra experiência para observar o serviço da rádio a funcionar e verificar que efetivamente funciona, sem ver o impacto no desempenho, visto que não esperamos ver melhoramento no desempenho.

Para a primeira experiência criamos um grafo que é composto por um nó inicial e um nó final estáticos. No caminho do nó inicial para o final, existe uma bifurcação, um dos caminhos sendo mais curto e outro mais comprido. Contudo, o mais curto dispõe de semáforos, fazendo com que os veículos sintam que aquele caminho demora demasiado tempo ("congestionado") e avisam outros sobre o mesmo, de forma a tomarem o caminho mais comprido e sem semáforo. Nesta experiência, desativando o comportamento de aviso eram necessárias aproximadamente 500 unidades de tempo (ciclos de movimento, não havendo uma conversão direta para tempo real; A quantidade de movimento é fixa para cada ciclo) e com o comportamento de aviso eram necessárias à volta de 300 unidades de tempo. Para cada um dos cenários acumularam-se 100 resultados e gerou-se a mediana.

Para a segunda experiência usou-se um contexto parecido, a única diferença sendo que o caminho mais curto não teria semáforo. Desta forma esse caminho não é tomado como "congestionado" pelo veículos e apenas pelo agente rádio. Neste caso consegue-se observar que alguns veículos são ocasionalmente redirecionados para o caminho mais longo.

Conclusões

5.1 Análise dos resultados das experiências

Na experiência 1 conseguimos concluir que o sistema funciona para o fim descrito no enunciado, ou seja, veículos informarem outros sobre vias congestionadas. Para além de concluir que o sistema funciona, também conseguimos observar que é eficaz devido à redução significativa do tempo necessário para os agentes chegarem ao seu destino pretendido.

Na experiência 2 dá para perceber que o sistema de rádio afeta o comportamento normalmente determinista dos veículos (com o A* seria determinista devido a querer usar sempre o caminho mais curto) optando às vezes pelo caminho mais comprido.

5.2 Desenvolvimento do trabalho

As bibliotecas usadas têm muito para compreender ao ínicio. Contudo, após um período inicial, são ferramentas fáceis de usar e que permitem uma simulação potencialmente muito versátil de sistemas multi-agente.

Pensamos que sistemas multi-agente têm alta aplicabilidade neste contexto, especialmente na área do planeamento das vias rodoviárias. Saber que efeitos a adição de uma via numa zona, remoção noutra ou conversão para sentido único podem ter no fluxo do tráfego num contexto de uma cidade, por exemplo.

Melhoramentos

- Permitir a manipulação de mais parâmetros da simulação, nomeadamente o ritmo a que os agentes são criados ou de outros eventos.
- Permitir manipulação do grafo da simulação em tempo real.
- Criar agentes a um ritmo variável.
- Haver um maior grau de interoperabilidade dos agentes de forma a criar situações mais interessantes.

Recursos

7.1 Bibliografia

https://paginas.fe.up.pt/~eol/AIAD/aulas/jade_en.pdf
http://jade.tilab.com/
https://repast.github.io/
https://web.fe.up.pt/~hlc/doku.php?id=sajas

7.2 Software

JADE - http://jade.tilab.com/
Repast Simphony - https://repast.github.io/index.html
SAJaS - https://web.fe.up.pt/~hlc/doku.php?id=sajas
Eclipse - http://www.eclipse.org/

7.3 Elementos do grupo

Andreia Rodrigues	33%
Eduardo Leite	33%
Francisco Queirós	33%

Apêndice

A única funcionalidade que o utilizador tem controlo na versão atual do trabalho é a morfologia do grafo que o programa usa.

O programa procura no diretório ativo um ficheiro chamado "data" de onde vai extrair a informação sobre o grafo.

Esta informação é uma sequência de linhas em um dos cinco formatos a seguir:

1. Declaração de um nó: N <identificador (string) do nó> <posição x do nó> <posição y do nó>
2. Declaração de uma aresta: E <identificador do nó de partida> <identificador do nó de chegada>
3. Declarar que um nó tem as suas saídas controladas por semáforos: S <identificador do nó>
4. Indicar que os agentes veículo devem aparecer sempre de um nó específico (*force spawn*): FS <identificador do nó>
5. Indicar que os agentes veículo devem sempre ter um nó específico como objetivo (*force goal*): FG <identificador do nó>

O programa não é sensível à ordem das linhas mas se forem usados identificadores nos formatos 2-5 que não forem declarados no formato 1, o programa vai funcionar de forma inesperada.