Parque de Estacionamento

Agentes e Inteligência Artificial Distribuída

4º ano - 1º semestre

Mestrado Integrado em Engenharia Informática e Computação

Bernardo Belchior - up201405381@fe.up.pt
Maria João Mira Paulo - up201403820@fe.up.pt
Nuno Miguel Mendes Ramos - up201405498@fe.up.pt

10 de Dezembro de 2017

Conteúdo

1 Introdução

1.1 Descrição do cenário

Atualmente deparamos-nos nos com uma grande dificuldade de estacionamento em áreas metropolitanas. O elevado fluxo de trânsito e a lotação dos parques de estacionamento geram uma procura exaustiva de estacionamento no centro da cidade. Ao mesmo tempo, parques na periferia encontram-se na sua maioria desocupados.

Com este desequilíbrio surge a ideia de provocar uma maior procura aos parques não concentrados no centro da cidade e, assim, regular a alocação em parques de estacionamento.

Uma solução encontrada é a aplicação de ***Automated Dynamic Pricing (ADP)***, onde o preço de estacionamento durante um certo período de tempo varia conforme o dia e hora da semana, mas que poderá ter efeitos nocivos na sociedade. O lucro gerado pelo parque e o bem-estar social são algumas das métricas que serão avaliadas através deste projeto.

1.2 Objetivos do trabalho

Um Sistema Multi-Agente *(SMA)* é um sistema computacional composto por múltiplos agentes. Estes agentes exibem um comportamento autónomo mas ao mesmo tempo interagem com os outros agentes presentes no sistema, trabalhando juntos de forma a desempenhar determinadas tarefas ou satisfazer um conjunto de objetivos.

O objetivo do trabalho baseia-se na implementação de uma simulação multi-agente com o propósito de estudar diferentes formas de gestão dos preços praticados pelos parques de estacionamento.

Desta forma, pretende-se estudar o efeito da utilização de uma tarifa dinâmica ao invés do uso de uma tarifa estática no lucro gerado pelos parques de estacionamento, na competitividade e no bem-estar social.

1.3 Resultados esperados e forma de avaliação

Os resultados esperados dependem muito do tipo de esquema utilizado pelos parques de estacionamento. Num esquema com tarifa fixa, é esperado que o bem estar da sociedade seja privilegiado, uma vez que, à partida, os preços serão mais baixos e, portanto, haverá mais condutores com lugar de estacionamento mais perto do seu destino. Por outro lado, espera-se que uma tarifa dinâmica aumente a receita dos parques e que cause um maior descontentamento da população dado que poderão pagar mais pelo mesmo lugar quando comparado com a tarifa estática.

De modo a avaliar os vários cenários serão feitas várias simulações com valores diferentes para o tipo de tarifa, número de condutores omniscientes e número de condutores exploradores.

Desta forma será possível identificar qual o conjunto de fatores que favorecem certos cenários e determinados objetivos.

2 Plataformas e Ferramentas a usar

Um dos propósitos da unidade curricular Agentes e Inteligência Artificial Distribuída é a utilização de plataformas capazes de criar agentes de uma forma

distribuída e de executar simulações sobre esse agentes.

Desta forma e, com o propósito de atingir o objetivo final do trabalho recorreremos ao uso de três plataformas distintas: *JADE*, *REPAST* e *SAJaS*.

2.1 Descrição das características principais

De seguida apresentam-se as principais características das tecnologias escolhidas para a execução deste trabalho.

JADE

JADE é uma *framework open source* implementada em Java que simplifica a implementação de sistemas multi-agente distribuídos de acordo com as especificações da FIPA (*Foundation of Intelligent Physical Agents*).

Uma aplicação baseada em JADE é composta por agentes, associados a um nome único, que executam tarefas de acordo com um comportamento definido e interagem através de mensagens.

A plataforma dispõe de *containers*, onde residem os agentes. Estes agentes podem não estar, necessariamente, na mesma máquina. Cada máquina tem uma JVM (*Java Virtual Machine*) e cada agente tem uma *thread*, atingindo-se, assim, uma plataforma distribuída.

As configurações do JADE podem ser controladas através de uma interface. A partir desta interface é possível criar novos agentes, alterar a configuração dos mesmos dinamicamente ou alterar as máquinas onde estão contidos.

REPAST

A *framework REPAST* é uma plataforma *open source* que simplifica a implementação de simulações baseadas em agentes ao disponibilizar ao utilizador o uso de uma biblioteca com uma grande variedade de códigos e exemplos, para criar, executar, apresentar e recolher dados dos modelos desenvolvidos.

Desta forma, ao facilitar a recolha de dados da simulação e a visualização destes através de gráficos ou registos, o *REPAST* possibilita a análise da interação entre os agentes.

SAJaS

A *framework SAJaS* é uma API que faz a ponte entre uma plataforma de desenvolvimento de sistemas multi-agente, JADE, e uma plataforma de simulações multi-agente, REPAST.

A ferramenta *SAJaS* melhora *frameworks* de simulação de sistemas multi-agente, com determinadas características do JADE. Além disso, esta tecnologia promove, para programadores de JADE, um desenvolvimento mais rápido de modelos de simulação.

2.2 Realce das funcionalidades relevantes para o trabalho

De seguida apresentam-se de que forma é que as tecnologias mencionadas e suas funcionalidades se revelam relevantes na execução do projeto em questão.

JADE

O *JADE* permite definir *behaviours* e trocar mensagens entre agentes. O primeiro característica explicita o comportamento que cada agente deve ter, enquanto a segunda funcionalidade possibilita a existência de transferência de informação entre os elementos do sistema.

Extrapolando para o nosso problema, alguns exemplos de *behaviours* a implementar seriam:

- Dirigir-se ao parque mais próximo, para os condutores exploradores;
- Dirigir-se ao parque com maior utilidade, para os condutores omniscientes;
- Atualizar a tarifa conforme o esquema de preço, para os parques de estacionamento.

O *JADE* também permite aos agentes trocarem mensagens usando o formato ACL definido pela FIPA. Este sistema será útil no nosso problema, para que os condutores comuniquem com os parques de estacionamento com o intuito de verificar a sua lotação e/ou tarifa atual.

Esta plataforma, oferece um sistema de "páginas amarelas"onde é possível a um agente encontrar outro agente que fornece um serviço necessário para o primeiro acabar uma determinada tarefa. Este sistema será útil, para que, os diferentes agentes percebam quais as entidades em funcionamento.

Em suma, o *JADE* será a ferramenta responsável pela lógica de simulação.

REPAST

O *REPAST* permite visualizar uma determinada simulação, sendo possível alterar os parâmetros do modelo, como, por exemplo, adicionar qualquer tipo de agentes. Além disso permite a criação de múltiplas simulações, de forma a ser possível testar diferentes e variados cenários.

O *REPAST* permite, também, a extração de informação para futuramente ser analisada. É possível selecionar as informações a serem guardadas, e escolher onde guardar essa informação, se na consola ou num ficheiro. Podemos usar esta funcionalidade para, por exemplo, guardar o número de carros dentro de um determinado parque. Por fim, é também possível construir gráficos, no final de uma simulação. Uma ferramenta muito útil para uma análise mais precisa e objetiva.

Em conclusão, o *REPAST* será a ferramenta usada para testar a nossa simulação.

SAJaS

Plataforma responsável por iniciar a simulação, responsável por instanciar o *REPAST* e o *JADE*. Será esta tecnologia que fará o transporte de informação entre as duas plataformas de maneira abstrata e invisível para o programador.

3 Especificação

3.1 Identificação e caracterização dos agentes

Consideramos dois tipos de agentes, *Parking Facilities* e *Drivers*.

3.1.1 *Drivers*

Arquitetura e Comportamento

Os agentes do tipo *Driver* podem ser *Guided* ou *Explorer* e são ambos agentes **baseados em utilidade**, isto é, preocupam-se em tentar maximizar as suas expectativas e, através de uma função de utilidade estabelecer preferências entre sequências de estados que permitam atingir os mesmos objetivos. Atingir a maior utilidade significa atingir a maior medida de satisfação.

1. ***Guided Drivers*** são caracterizados pelo seu total conhecimento acerca de preços e localização dos parques de estacionamento. São agentes que sabem quais as condições mais satisfatórias e que estão seguros quanto às suas possibilidades. Este agente, exatamente por ter total conhecimento dos preços em vigor, é o que mais rapidamente reage à subida de preços.

   Quando entram no sistema sabem qual o parque que maximiza a sua utilidade e dirigem-se a esse. No caso do parque não ter lugares disponíveis repetem o processo com os restantes. Se a única solução possível resultar numa utilidade negativa para o condutor, o agente sai do sistema.

   - **Medida de desempenho:** Encontrar um determinado parque de estacionamento com disponibilidade e com o preço habitual.
   - **Ambiente:** Mapa da Cidade
   - **Atuadores:** Direcionar-se ao próximo parque que conhece ou estacionar.
   - **Sensores:** Disponibilidade do parque e preço.

2. ***Explorer Drivers*** são caracterizados pela sua ignorância, isto é, por não terem consciência dos preços realizados pelos parques. Consequentemente, procuram apenas um parque de estacionamento livre perto do seu destino e, consequentemente, não reagem tão rapidamente à mudança de tarifa.

   - **Medida de desempenho:** Encontrar um parque de estacionamento disponível, com um preço satisfatório e perto do local de destino de forma a que o condutor não tenha que se deslocar uma grande distância a pé.
   - **Ambiente:** Mapa da Cidade
   - **Atuadores:** Continuar a procurar ou estacionar.
   - **Sensores:** Disponibilidade do parque, proximidade ao local de destino e preço.

Os agentes *Drivers* têm conhecimento do respetivo *ID*, das coordenadas do local de partida, coordenadas do local de destino pretendido, hora de chegada, máximo preço por hora, duração do estacionamento, máxima distância a pé, data e hora de inicio de estacionamento.

Estratégias

A função de utilidade é a mesma para os dois tipos de *Drivers*:

$$w_{i,j} = C_i - pr_i * (P_{i,j})^u - w_i * (W_{i,j})^v$$

$$P_{i,j} = \alpha * price_j * duration\_of\_stay_i$$

$$W_{i,j} = \beta * distance\_to\_destination_{i,j}$$

C_i representa a utilidade, ou seja, a satisfação atingida pelo condutor ao chegar ao seu destino sem ter que se deslocar a pé ou pagar pelo estacionamento num parque. Este valor é atribuído aleatoriamente a cada condutor.

$P_{i,j}$ representa o preço a pagar pelo condutor tendo em conta o tempo no qual teve estacionado no parque, multiplicado por uma constante *alpha*.

$W_{i,j}$ representa o esforço dedicado pelo condutor de forma a chegar do parque de estacionamento ao seu destino, multiplicado por uma constante *beta*.

As constantes α e β permitem obter uma comparação mais justa e realista do impacto dos metros e euros na utilidade do condutor.

pr_i, w_i tratam-se de valores gerados aleatoriamente que simbolizam a personalidade do agente i. Isto é, traduzem a utilidade gerada no agente ao pagar a quantia $P_{i,j}$ e ao andar $W_{i,j}$ metros a pé.

Os expoentes u e v pretendem criar não linearidade no impacto do preço e da distância a percorrer a pé. Por exemplo, o acréscimo de 1 € ao custo do parque a um cliente que terá de pagar 2 € terá, obviamente, mais impacto do que num cliente que deve um valor maior, como 20 €.

3.1.2 *Parking Facilities*

Arquitetura e Comportamento

Os agentes *Parking Facilities* podem ser de dois tipos, dependendo do esquema de preços que querem adotar, *Static* ou *Dynamic*. Existem sete esquemas de preços a aplicar, um para cada dia da semana.

Ambos os agentes podem reagir aos comportamentos dos *Drivers*, mas não devem interagir com os mesmos, isto é, não existe negociação de preços. Tratam-se, por isso, de agentes reativos simples com um estado interno, ou seja, que mantêm um relógio interno com informação atualizada do tempo de estacionamento e, em função disso, calculam o esquema de preço.

Os agentes *Parking Facilities* têm conhecimento da localização, nome, preço, capacidade do parque e da empresa onde pertencem.

1. *Static*
   Adotam um esquema de preço estático, ou seja, que não varia com fatores do ambiente.

2. *Dynamic*
   Adotam um esquema de preço dinâmico, procurando atingir um maior lucro. O preço que exercem varia em função do dia da semana e da hora. A cada final de semana, é feita uma atualização aos esquemas de preço com o objetivo de maximizar o lucro obtido.

- **Medida de desempenho:** Atingir o maior lucro

- **Ambiente:** Mapa da Cidade
- **Atuadores:** Continuar a procurar ou estacionar.
- **Sensores:** Disponibilidade do parque, dia da semana e hora.

Estratégias

O preço final de estacionamento é calculado tendo em conta três parâmetros: o preço por minuto, um valor mínimo que deve ser pago por todos os clientes e um valor máximo que, pode ser pago, por exemplo, no caso do condutor usufruir de um dia inteiro estacionado no parque.

Existe, além disso, outro parâmetro que pode causar tanto inflações como deflações no preço por minuto. O valor por defeito deste parâmetro é 1 e, no caso de aumentar ou diminuir durante a otimização, causará variações, hora a hora, no preço final. Por exemplo, no caso do preço por hora ser 1, e a inflação 1.10, o preço irá variar de acordo com tabela a seguir.

Hora	Preço	Preço Final
1	1	1
2	1.1	2.1
3	1.21	3.31
4	1.331	4.641

Tabela 1: Preço ajustado tendo em conta o parâmetro de oscilação

O preço final pode depender ainda da capacidade do parque. Assim, caso a disponibilidade do parque varie entre os 30% e os 70%, o preço mantêm-se constante. Caso contrário, o valor oscila linearmente de acordo com o parâmetro a ser atualizado.

Cada parque de estacionamento deve atualizar os parâmetros referidos anteriormente exclusivamente uma vez por semana e, além disso, cada parâmetro deve ser atualizado cinco vezes antes de atualizar o próximo. Desta forma garante-se uma maior confiança e certeza no impacto de cada um dos parâmetros.

A expressão usada na atualização de parâmetros, no caso de, anteriormente, ter existido uma inflação no preço é:

$$x_i + 1 = x_i + \delta.x_i.(\gamma - 1)$$

Caso contrário, ou seja, caso se tenha verificado uma deflação no preço, deve ter em conta a expressão:

$$x_i + 1 = x_i - \delta.x_i.(\gamma - 1)$$

Sendo que, δ representa a taxa de aprendizagem do agente e, γ, representa a diferença, em percentagem, dos lucros obtidos na semana anterior. Este tipo de aprendizagem permite que o parque aumente os preços para valores considerados justos tanto pelo dono do parque, como pela sociedade.

3.2 Ambiente

O ambiente escolhido simula o centro de uma cidade, uma vez que se trata do local onde existe uma maior procura de estacionamento e, por isso, uma maior oferta de parques de estacionamento.

Este mapa é representado através de uma matriz que inclui os parques de estacionamento e as ruas que lhes dão acesso.

3.3 Protocolos de Interação

Os agentes *Drivers* podem ter, ou não, conhecimento total do sistema de parques. No caso dos *Explorer Drivers*, não existe necessidade de interação com outros agentes, tirando o parque em que pretende estacionar. Por outro lado, os *Guided Drivers* precisam de saber toda a informação do sistema a qualquer momento. Devido a estas características existem duas alternativas para a passagem de informação sobre o estado dos parques aos *Guided Drivers*

- Os parques de estacionamento enviam informação quando esta muda, isto é, se entra ou sai um veículo, o parque de estacionamento envia uma mensagem a todos os *Guided Drivers* informando a nova capacidade;
- Os *Guided Drivers* fazem perguntas constantes sobre a capacidade do parque;

Ambas abordagens são possíveis e cada uma tem as suas desvantagens. A primeira pode ser mais vantajosa num sistema com muitos agentes, independentemente do tipo, enquanto a segunda será melhor numa simulação com poucos *Guided Drivers*, porque serão necessárias menos mensagens para a mesma informação. A primeira parece ser mais vantajosa na maioria dos casos, pelo que será essa a utilizada.

Além deste caso de comunicação, os *drivers* também terão que comunicar com o parque para poderem entrar ou sair, pelo que haverá mais duas mensagens que permitem entrar ou sair do parque.

3.4 Faseamento do projeto

Fase 1

Criar agentes, definir comportamentos e implementar as estratégias para cada um deles.

Fase 2

Fazer um testes isolados dos comportamentos para cada um dos agentes. Isto é, testar comportamento de:

1. Um agente *Guided Driver* e um agente *Static Parking Facility*;
2. Um agente *Explorer Driver* e um agente *Static Parking Facility*;
3. Um agente *Guided Driver* e um agente *Dynamic Parking Facility*;
4. Um agente *Explorer Driver* e um agente *Dynamic Parking Facility.*

Fase 3

Fazer testes com múltiplos agentes

Fase 4

Retirar conclusões acerca de qual o método que permite regular a alocação em parques de estacionamento, e de que forma é que este método influencia o bem-estar social e a competitividade.

4 Recursos

4.1 Bibliografia

1. Thomas Vrancken Daniel Tenbrock Sebastian Reick Dejan Bozhinovski Gerhard Weiss Gerasimos Spanakis (2017): Multi-Agent Parking Place Simulation, in Advances in Practical Applications of Cyber-Physical Multi-Agent Systems: The PAAMS Collection: 15th International Conference, PAAMS 2017, Porto, Portugal, June 21-23, 2017, Proceedings, Y. Demazeau, et al., Editors. 2017, Springer International Publishing: Cham. p. 272-283.

2. Jade Documentation - http://jade.tilab.com/doc/tutorials/JADEProgramming-Tutorial-for-beginners.pdf

3. Repast Simphony Documentation - https://repast.github.io/docs/RepastJavaGettingStarted.pdf

4. SAJaS Documentation - https://web.fe.up.pt/ hlc/doku.php?id=sajas

4.2 Software

O projeto será desenvolvido através do IDE *Eclipse*, com acesso às *frameworks JADE*, *REPAST* e *SAJaS*.