Universidade de Aveiro

Departamento de Electrónica, Telecomunicações e Informática

Linguagens Formais e Autómatos

Enunciado do trabalho prático (Ano Lectivo de 2013/14) Abril de 2014

1 Introdução

Pretende-se projetar e desenvolver um conversor que a partir de uma descrição única num dialeto JSON (`http://www.json.org`) produza os ficheiros XML de configuração do ambiente de simulação CiberRato e uma descrição em formato URDF do ambiente de jogo.

O CiberRato é um ambiente de simulação de agentes robóticos, usado em vários concursos e em várias unidades curriculares de robótica e de inteligência artificial. As ferramentas estão disponíveis em código fonte (`http://www.sourceforge.net/projects/cpss`). A configuração de uma simulação é feita através de 3 ficheiros XML, que descrevem o labirinto, a grelha de partida e alguns parâmetros de simulação.

2 Objectivos

Pretende-se que a caraterização da configuração seja feita numa única descrição usando uma linguagem baseada em JSON. Esta descrição deve incorporar obrigatoriamente as 3 componentes anteriores: o labirinto, a grelha de partida e os parâmetros de simulação. Adicionalmente, pode incorporar outros elementos considerados relevantes.

Com base nesta descrição o conversor a implementar deve produzir os 3 ficheiros de configuração aceites pelo ambiente de simulação CiberRato.

Pretende-se que o conversor também produza uma descrição em formato URDF, que possa ser usada para uma visualização 3D do cenário de jogo.

3 Definição da linguagem

O dialeto JSON está parcialmente definido (ver o ficheiro `./example.json` para um exemplo). Deve suportar as entradas `"challenge name"`, `"challenge class"`, `"cycle time"`, `"duration"`, `"scenario description"`, `"robot modelling parameters"`, `"scoring parameters"` e `"debugging parameters"`.

- O valor da entrada `"challenge name"` é uma *string* e indica o nome da simulação.
- A entrada `"challenge class"` indica o tipo de desafio e pode ter os valores `"competitive"` ou `"collaborative"`.
- O valor da entrada `"cycle time"` é um número inteiro e indica o período de simulação em milissegundos.
- O valor da entrada `"duration"` é um número inteiro e indica a duração total da simulação em número de ciclos.
- A entrada `"scenario description"` descreve o cenário de atuação dos robôs e está descrito na secção 3.1.

- A entrada `"robot modelling parameters"` indica os valores de parâmetros usados para caraterizar os sensores e os atuadores dos corpos virtuais dos robôs e para caraterizar o acesso que os agentes têm aos sensores.
- A entrada `"scoring parameters"` indica os valores de parâmetros usados para calcular a pontuação dos robôs.
- A entrada `"debugging parameters"` indica os valores de parâmetros que permitem o acesso a informação extra durante a fase de desenvolvimento.

Os três tipos de parâmetros estão associadas ao ficheiro XML de parâmteros de simulação. Mais informação sobre a lista de parâmetros é dada na secção 3.2.

3.1 Descrição do cenário

Como dito anteriormente, a entrada `scenario description` é um objeto que contém a descrição do cenário de jogo. Corresponde a um espaço retangular, limitado por paredes exteriores e povoado de obstáculos, fárois e áreas-alvo e onde está marcada uma grelha de partida. Considera-se que o canto inferior esquerdo corresponde ao ponto (0,0), que o eixo dos XX cresce da esquerda para a direita e que o dos YY cresce de baixo para cima. O atributo `dimensions` é usado para indicar as dimensões do espaço.

Os obstáculos correspondem a paredes. São estruturas poligonais, definíveis em dois formatos diferentes. Em ambos existe um atributo, designado `height`, que define a altura da parede. O formato é especificado usando o atributo `thickness`. Se este atributo for igual a zero ou estiver omisso, a parede é representada pelos vértices de uma poligonal fechada. Se for diferente de zero, a parede é representada por uma poligonal aberta, tendo a parede a espessura especificada pelo atributo `thickness`.

Admite-se a possibilidade de criação de modelos que possam ser posteriormente instanciados para criar paredes iguais em diferentes posições no espaço. Cada modelo tem o seu próprio sistema de coordenadas. Um modelo pode ter a altura não especificada. Um modelo pode instanciar outros modelos. Na instanciação de um modelo deve ser especificada a translação e a rotação e, eventualmente, a altura.

Os faróis são elementos emissores de radiação sentida por um sensor acoplado aos robôs. As áreas-alvo são círculos no chão que os robôs têm de visitar. Normalmente, os fárois estão associados a áreas-alvo, mas tabém podem ser independentes. O posicionamento de um farol na área de jogo é definido por um ponto na área de jogo e por uma altura. As áreas-alvo são caraterizadas por um ponto e um raio.

A grelha de partida indica as posições e orientações dos robôs no início de uma simulação. Deve ser representada por um array de triplos, cada um contendo uma posição e uma orientação.

3.2 Parâmetros de simulação

A lista de parâmetros válidos é fornecida através de um ficheiro JSON (ver o ficheiro `./param-list.json` para um exemplo). O nome de cada entrada neste ficheiro representa o nome de um parâmetro. O valor de cada entrada carateriza o parâmetro em termos de classe, tipo de valor, valor por defeito e nome do atributo XML correspondente.

A classe pode ter um dos valores `"robot modelling parameters"`, `"scoring parameters"` ou `"debugging parameters"`. O valor de um parâmetro pode ser dos tipos inteiro positivo (`"uint"`), real (`"double"`), booleano (`"boolean"`) ou interruptor (`"switch"`). Uma variável do tipo `"switch"` pode ter os valores `"on"` ou `"off"`. O valor por defeito indica o valor a assumir

pelo parâmetro caso não esteja presente no ficheiro de configuração. Se não for indicado valor por defeito para um parâmetro é porque ele tem de estar presente no ficheiro de configuração.

4 Tabela de símbolos

A gestão de parâmetros deve ser feita através de uma tabela de símbolos. Esta tabela deve ser preenchida inicialmente com a lista de parâmetros válidos. Depois, durante o *parsing* do ficheiro de entrada (JSON) esta tabela deve ser usada para verificar se os nomes dos parâmetros são válidos. Se o forem, o valor por defeito deve ser substituído pelo valor especificado, caso este esteja dentro da gama válida.

A gestão de modelos também deve ser feita numa tabela de símbolos. Fica ao critério do grupo usar a mesma ou outra tabela.

5 Plano de trabalho

Prevê-se o seguinte plano de trabalho:

1. Familiarização com o ambiente de simulação CiberRato, com particular incidência nos dialetos XML usados para descrever o labirinto, a grelha de partida e os parâmetros de simulação.
2. Familiarização com a linguagem JSON. Pode usar um editor *online* de JSON (por exemplo, `http://www.jsoneditoronline.org/`) para apoio nessa familiarização.
3. Definição da gramática do dialeto JSON desenvolvido pelo grupo. A concretização deste ponto obriga à elaboração e entrega de um programa em `bison+flex` que avalie sintaticamente textos escritos de acordo com o dialeto desenvolvido. A entrega deve ser feita na forma de uma pasta *zipada* com o material necessário para gerar o programa, incluindo a existência de uma `Makefile`. Obriga ainda à escrita e entrega de um ficheiro que use o dialeto desenvolvido para descrever o labirinto, a grelha de partida e os parâmetros de simulação usados na final do CiberRato de 2005.
4. Projeto e implementação da tabela de símbolos. Este ponto pressupõe o *parsing* do ficheiro JSON que indica a lista de parâmetros de simulação válidos. A concretização deste ponto obriga à elaboração do módulo que implemente a tabela de símbolos.
5. Definição da gramática de atributos que implemente o conversor pretendido. A concretização deste ponto obriga à elaboração e entrega de um programa em `bison+flex+C/C++` que implemente o gerador dos ficheiros XML de configuração do CiberRato.
6. Incorporação do gerador de URDF. **Falar com o docente antes de abordar este ponto.**

6 Metodologia de avaliação

A definir oportunamente.

7 Execução do trabalho e prazo de entrega

As últimas 4 semanas de aulas práticas serão usadas para apoio na execução do trabalho. Para cada semana será definido um objetivo e a intenção é que tanto quanto possível cada grupo

o atinja. As aulas não são para executar o trabalho, mas para apontar direções e esclarecer dúvidas.

Em cada semana, um elemento diferente do grupo funcionará como porta-voz na interação com o docente na apresentação do trabalho realizado até aí.

O trabalho deverá ser entregue até às 24 horas do dia 8 de Junho de 2014.

8 Grupos

O trabalho deve ser realizado por grupos de 4 ou 5 elementos.