Renan L. S. Del Bel

Mestrando em Ecologia, meu interesse é na estrutura de comunidades vegetais, principalmente se existe estruturação na distribuição de caracteres funcionais em uma floresta.

Meus exercícios resolvidos Meus Exercícios

Trabalho Final

Plano A

Justificativa

Pong foi um jogo que marcou a história, ele foi o primeiro comercializado para o público geral, em larga escala, com sucesso. Sua criação foi o início da era da indústria de jogos eletrônicos. Antes disso, jogos eram em geral trechos de código que programadores compartilhavam entre si por diversão ou demonstrações de poder computacional. E que jogo melhor para testar os limites no R? Sendo uma linguagem orientada a objetos, o R é, a primeira vista, incompatível com um jogo do tipo arcade, que depende de eventos. Sendo um jogo simples, sua construção vai depender menos da capacidade de escrever grandes quantidades de códigos com várias condicionais e mais da capacidade de abstração, pesquisa de soluções e resiliência a frustrações. À semelhança dos jogos pré-pong, o produto final é só uma curiosidade para quem tiver acesso ao código, mas o verdadeiro objetivo está no processo: Ser um desafio para um programador e talvez uma fonte de inspiração disponível na comunidade, por apresentar a resolução para problemas incomuns e explorar o alcance do que é viável.

Pseudo Código

1. Abrir um dispositivo gráfico para trabalhar
2. Estabelecer qual será o tamanho do quadro
3. Criar uma barra interativa em janela separada, para interagir com o usuário
4. Aguardar clique na tela para começar a partida
5. Marca o tempo do sistema
6. Coeficiente Y de velocidade da bolinha é igual a 1
7. Coeficiente X de velocidade da bolinha é igual a 0
8. A posição em x do paddle é atualizada automaticamente para corresponder com a barra (começando o jogo no centro)
9. A posição da bolinha é atualizada somando os coeficientes X e Y nas respectivas coordenadas
10. Testa se a bolinha ultrapassou os limites do cenário
    1. Se ultrapassou uma parede vertical, inverter o coeficiente horizontal X, posicionar a bolinha onde deveria tocar a parede
    2. Se ultrapassou uma parede horizontal, inverter o coeficiente vertical Y, posicionar a bolinha onde deveria tocar a parede

    3. Se ultrapassou os limites do gol, testar se tocou o paddle
        1. Se não, o jogo acaba, marca o novo tempo do sistema
        2. Se sim, o coeficiente X é modificado de acordo com a posição do paddle que a bolinha tocou, com modificador 0 no centro e 1/-1 nas pontas.
        3. X não poderá ser 3 vezes maior que Y.
11. Plota o gráfico contendo a bolinha e o paddle
12. Aumenta Y em uma pequena taxa, para a dificuldade aumentar
13. Repete a partir do passo 7 até o jogo acabar.
14. Subtrair o tempo final do tempo inicial para saber quanto durou o jogo
15. Mostrar tela de game over
16. Retorna a duração do jogo

Plano B

Justificativa

Durante essa disciplina e Planeco os alunos são apresentados a uma forma de selecionar modelos usando comparações por anova par-a-par. Essa seleção é curiosa por ser um algorítimo repetitivo que é executado apenas parcialmente pela máquina (ajuste dos modelos, tabela anova e sumários) com outra parte sendo executada pelo humano (criação de modelos mais simples, decisão sobre o P-valor e chamar as funções que a máquina irá executar). Parece fazer sentido deixar a tomada de decisões a cargo do humano, mas então recordamos que se trata de um aluno seguindo alguns poucos critérios bastante objetivos e definidos a priori.

A tarefa repetitiva executada pelo aluno pode ser realizada pela máquina em muito menos tempo e trazendo resultados mais consistentes. Com isso os alunos podem checar suas respostas e comparar o efeito de partir de diferentes modelos cheios (incluindo os que seriam grandes demais para valer a pena testar manualmente) e comparar os resultados com os de outros critérios para selecionar modelos.

Essa função provavelmente não seria indicada para encontrar modelos inferenciais, por criar uma nova camada de caixa preta no processo e por ser cego à interpretação dos modelos que está gerando. Mas é útil para qualquer um que esteja praticando e pode até ser útil para criar modelos que sejam puramente preditivos.

Pseudo Código

1. Cria uma função de criar modelos onde:
    1. Salva o modelo recebido numa lista
    2. Estabelece x=1
    3. Para cada fator no modelo além do intercepto, partindo do mais complexo:
        1. Se não for significativo, e seu nome não estiver totalmente contido no nome de outro fator, criar um novo modelo sem esse fator
            1. Chamar a própria função usando o modelo novo, salvar resultado na lista
    4. Retorna a lista

1. Cria uma função de seleção onde

    1. A é a primeira posição da lista recebida
    2. Coloca A numa lista nova
    3. Partindo de X = 2 até chegar ao comprimento da lista recebida
        1. B é a primeira posição da lista de posição X na lista recebida
        2. Compara A e B pela função anova
        3. Se não der significativo, chama a própria função para a posição X da lista recebida e coloca o resultado na lista nova
    4. Retorna a lista nova

1. Cria uma função de limpeza, onde:
    1. Salva a primeira posição em uma nova lista
    2. Partindo da posição 2 até a última posição da lista
        1. Se o modelo da posição atual tiver mesma quantidade de fatores que os da nova lista, salvar na nova lista
        2. Se o modelo da posição atual tiver menos fatores que os da nova lista, apagar a nova lista e salvar este na primeira posição
    3. Retorna a nova lista

1. Usa as funções anteriores para criar e limpar os modelos a partir de um modelo cheio e alpha(padrão 0.05) fornecidos pelo usuário
2. Usa a função limpeza para retirar os modelos mais complexos, a menos que o usuário peça para que não seja feito
3. Retorna o resultado

Comentário Alê

> — *Alexandre Adalardo de Oliveira 2018/05/11 17:11* Renan,
>
> Gosto mais da proposta 1, não sou muito fã de procedimentos de seleção automática de modelos. Acho importante o ritual de simplificação para aumentar o entendimento do processo envolvido. Além dissoa iteração faz com que as pessoas fiquem mais intimas do modelo e do significado das variáveis. Mas isso é só minha visão. Ambas as propostas são factíveis, mas acho a 1 mais desafiantes para o exercício de programação. Bora no pong!

Trabalho Final

Extras: Além do acordado, a função trás um oponente virtual, um placar em tempo real, 3 modos de jogo e a opção de jogar novamente sem ter que chamar a função outra vez além de outras adições menores.

As seguintes considerações estão também comentadas no código:

1. Colocar o controle e o jogo em telas diferentes (como dito na proposta) tornava a experiência pior, agora eles estão na mesma janela

2. O grau de precisão da posição da bolinha é maior do que eu pretendia originalmente, então os cálculos para corrigir a posição após colisões foi retirado por ser quase imperceptível o efeito

3. Na proposta original seria retornado o tempo de jogo como forma de score, mas o cálculo de score usado aqui é mais interessante

4. O aumento de velocidade em y da bolinha também foi retirado por estar ativamente tornando o jogo menos divertido, além disso, para derrotar o oponente o jogador terá que acelerar a bolinha por conta própria

5. Dependendo do programa que você usar para trabalhar no R, as janelas criadas pelo tcltk podem aparecer quando você criar a função (mesmo sem dar o comando para rodar) e podem estar desconfiguradas. Nesse caso, surgirá a janela de fim de jogo com a mensagem padrão “Aperte Ok”. Apertar ok deve fazer as janelas fecharem. A função deve rodar normalmente independente disso acontecer.

6. Os pacotes tcltk permitem que as funções do R sejam chamadas de acordo com eventos e rodem em paralelo, por exemplo o slider pode ter uma função chamada sempre que seu valor é mudado. Provavelmente tem como fazer esse jogo rodar ainda melhor usando essas opções orientadas a eventos, mas considerando a justificativa da proposta achei melhor evita-las o máximo possível. A janela de novo jogo atribui uma função aos botões que é ativada no evento do botão ser apertado, mas como tk.wait mantém o R em pausa até um botão ser apertado e essa função apenas atribui um novo valor a uma variável, a lógica não é diferente de pedir input do usuário na linha de comando, usei a janela pela estética.

Linques

Função

Help

From:
http://ecor.ib.usp.br/ - **ecoR**

Permanent link:
http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2018:alunos:trabalho_final:renan.bel:start

Last update: **2020/08/12 06:04**