# Ademar Marques Lacerda Filho

Aluno de Graduação, Curso de Ciências Moleculares da USP (CCM-USP), último semestre (oitavo).
Iniciação Científica em Visão Computacional, no Instituto de Matemática e Estatística da USP (IME).
Orientador: Roberto Hirata Jr.

### Exercícios

Exercícios do Ademar

### Projeto Final

Não sou biólogo, nem trabalho muito com dados, vou fazer uma proposta de projeto ligada a teoria dos jogos que eu comecei a estudar recentemente. O Plano B parece mais trabalhoso que o Plano A, então eu pretendo implementar o plano A, e se der tempo implemento também o B no prazo dos projetos(not).

#### Plano A - Jogo da Vida

Dada uma matriz binária de células ativas ou inativas, podemos decidir em um conjunto de regras, as quais definirão se uma célula será ativada ou desativada na próxima iteração. A regra mais simples, comum e que produz resultados interessantes é: uma célula ficará desativa se existem *j* ou mais células vizinhas ativas e uma célula se ativa se existem pelo menos *i* células ativas na vizinhaça. Ou seja, ela ficará ativa se existir exatamente um número natural no intervalo [i,j] de células ativas na sua vizinhança direta (suas oito vizinhas).

Penso em fazer uma função em que dada a matriz inicial, *i* e *j*, faz uma animação da evolução temporal do sistema. Eu montaria uma grade retangular e representaria as células ativas por quadrados coloridos e as inativas por quadrados em branco.

Versão Final do Projeto: adgol.tar.gz

Para visualização rápida do código aqui está o código fonte: adgol.r

E aqui o arquivo de help da função principal (as outras estão no pacote): help_adgol.txt

#### Plano B - Jogos Iterados

Pensei também em implemetar um simulador de jogos clássicos de dois jogadores onde cada um deve tomar uma decisão binária e o resultado tem payoffs diferentes para cada combinação de decisões, como por exemplo Dilema do Prisioneiro. O usuário pode definir o payoff dos jogos, mas por padrão seriam pesos do Prisioner's Dilema, e também quantas iterações o jogo deve seguir (digamos que por padrão seria 10). A saída do jogo seria a pontuação de cada um dos jogadores em forma de

tabela. O usuário deve jogar contra o “computador” que estaria seguindo uma estratégia programada. Este jogo poderia ser no próprio terminal, ou o usuário teria a oportunidade de programar uma estratégia para si e executá-la. Outro feature seria o usuário poder escolher a estratégia de ambos jogadores para poder simular uma situação qualquer. Uma estratégia seria basicamente uma função que recebe todos os dados das i(n)terações anteriores e toma a decisão da jogada atual.

Se as decisões podem ser A e B, exemplos de estratégia são:

- tome a decisão ao acaso

- sempre escolha A

- alterne entre A e B

- inicie o jogo com A e repita sempre a escolha anterior do adversário (tit for tat)

- escolha A até que o adversário escolha B, depois escolha sempre B

- escolha A até o jogo acabar ou você acumular 50 pontos, depois escolha sempre B

- se sua pontuação está menor que 100, escolha A, senão escolha B

- se a pontuação do seu adversário é menor que a sua, escolha A, senão escolha B

Há diversos tipos de estratégia, eu implementaria algumas e deixaria disponível no pacote, mas o usuário teria a chance de implemetar as suas próprias estratégias e passar essas funções como parâmetros.

### Comentários sobre o Projeto

Salve, AD. Comentarios: O plano B eh muito mais legal que o plano A - mas tambem eh muito mais trabalhoso. Voce pode implementar cada estrategia como uma funcao que recebe a lista dos ultimos movimentos e decide qual a proxima jogada, e uma funcao “mestra” que recebe como parametros o numero de iteracoes, a tabela de pontos, e as duas estrategias (portanto, passando as funcoes como parametro e nao uma string que as represente, como a gente faz com a apply ou aggregate). Isso vai ficar bom demais, e voce pode usar isso pra fazer uns graficos bonitinhos com “Bonzinho vs. Bonzinho”, “Malvado vs. Tit for Tat”, “Tit for Tat vs. Greedy”. — *André Chalom 2012/03/27 11:36*

#### Ale

Olá Ademar! Concordo com as observações pertinentes do Chalom. Ele se entusiasmou pelo projeto, qq coisa pegue no pé dele para te ajudar! Suas propostas e seu perfil demonstram que vc. tem conhecimento de programação e acredito que o desafio 2 seja mais sedutor e envolvente... entretanto, como vc. mesmo coloca é um desafio mais trabalhoso. Deixo para vc. decidir. Caso opte pela 1 sugiro que ao invés de um limiar de número pré-determinado de células ativas ao redor, use uma distribuição de probabilidades, ou seja que a ativação dependa não só do número de células ativas, mas também de um componente estocástico associado, acho que as simulações podem

produzir resultados ainda mais interessantes. De qq forma, como disse, plano B é bem sedutor... e apesar de não ser biólogo a teoria tem tudo a ver com ecologia...

*Você pode contribuir para esta página editando este tópico!!*

---