Alunos: Eduardo de Moraes e Pedro Henrique Dadalt de Queiroz;

Trabalho I - Haskell

Definição Kojun

Kojun é um jogo que consiste no preenchimento de uma matriz de células, delimitado por grupos de tamanho e formato variado e que respeita certas condições. Cada grupo possui células valoradas de 1 até a quantidade de células presentes nesse grupo. Cada célula deve obedecer a regra de não conter o mesmo valor no mesmo grupo e nem em células verticalmente e horizontalmente ortogonais, ou seja, não pode ser igual aos vizinhos imediatos da esquerda, direita, cima e baixo. Além disso, se duas células são verticalmente adjacentes na mesma área, o valor da célula de cima deve ser maior que o valor da célula de baixo, exemplo presente na imagem a seguir.

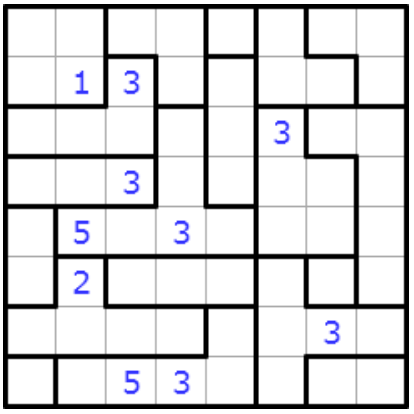

Imagem 1

Para dar início às análises que permitem o preenchimento completo da matriz, o jogo geralmente é inicializado contendo algumas células valoradas.

Modelagem do trabalho

O passo-a-passo para instalação de dependências e execução do projeto estão no *README.md*. Utilizou-se como base a resolução por **CSP** (*Constraint Satisfaction Problem*), utilizando uma biblioteca externa *Control.Monad.CSP*. Utilizando a biblioteca é possível definir o domínio das variáveis e as restrições sobre elas aplicadas. Dessa forma, o intuito é restringir as possibilidades de valores em cada célula da matriz, para chegar em apenas uma possibilidade e a conseguinte solução, que é o que a função *oneCSPSolution* retorna.

O uso de listas é imprescindível para resoluções mais complexas em Haskell e o presente trabalho não fugiu do padrão. Para conseguir popular a lista de restrições de cada célula presente na estrutura *dvs*, precisou-se selecioná-las com o uso de *list comprehension*.

Organização do trabalho

A organização do trabalho consistiu na participação coletiva de todos os membros nas etapas que formam a tarefa: código fonte e relatório. Os integrantes reuniram seus conhecimentos acerca da linguagem Haskell, visando encontrar uma ideia em comum acordo

que tivesse a capacidade de resolver o problema proposto pelo trabalho. Cada integrante buscou apoio em soluções já implementadas que ajudassem a resolver o Kojun.

Dificuldades

Inicialmente, decidiu-se implementar uma resolução para o jogo Vergleichssudoku, mas como a base do código se utiliza de conceitos avançados como *Functors*, *Applicatives* e *Monads*, as principais dificuldades foram justamente sobre isso. Apesar do uso desse paradigma do Haskell não ser novidade para o grupo, os conceitos abordados causaram complicações na procura por uma solução. Cada integrante estudou os assuntos e procurou materiais e referências de código para a resolução do trabalho, porém, colocar em prática os conceitos não foi tão trivial e encontramos problemas do tipo "como fazemos quando uma função está em um contexto, como um Maybe (a -> a -> m b), mas precisamos aplicar valores fora de um contexto, como um [a]?".

Por fim, decidimos implementar uma solução para o Kojun e o desenvolvimento se deu muito menos complexo, sem o uso de *Functors* e *Applicatives*, apenas com o uso abstraído pela biblioteca externa de *Monads*.