Organização de Jantar

Autores: Tiago José Grosso Pacheco (up201402722) e Vasco Ferreira Ribeiro (up201402723)

Afiliações: FEUP-PLOG, Turma 3MIEIC04, Grupo Organização de Jantares\_5

Resumo

Neste trabalho teríamos, assim, de resolver o problema de Organizar um Jantar, utilizando a linguagem PROLOG e as diversas técnicas de restrição e otimização que nela existe. O problema de Organizar um Jantar consiste em conseguir dispor os vários convidados do jantar em mesas, de forma que, as pessoas se encontrem dispostas não aleatoriamente, mas sim pelo grau de afinidade entre as mesmas (quer grau de parentesco quer hobbies e interesses). Para resolver o problema, usamos a biblioteca clpfd do SICStus, pois contém várias funções uteis para a resolução.

1. Introdução:

O objetivo deste programa era implementar a solução de otimização em Prolog com restrições.

O problema Organização de Jantar tem como objetivo organizar convidados em mesas, de forma que haja compatibilidade entre os convidados.

Segue, no decorrer deste documento, a abordagem usada para a resolução do mesmo e a sua análise.

A solução implementada permite obter uma solução dando uma lista de listas já com as afinidades entre os convidados.

1. Descrição do Problema:

O problema principal deste trabalho é então Organizar um Jantar, dispondo os convidados em mesas. A organização tem como base um número máximo de mesas, um número máximo de convidados por mesa e um número de convidados. A principal restrição é as pessoas se sentir confortáveis, ou seja, terem afinidade com as pessoas que partilham a mesa. Pretende-se, assim, saber quantas mesas utilizar, e como ficarão distribuídas as pessoas.

1. Abordagem:
   1. Variáveis de decisão:

As variáveis de decisão são as mesas que vão ser utilizadas e os convidados que vão estar presentes na festa.

O domínio das mesas, Tables, é então um número de 1 a N (número de mesas da festa), onde cada número representa uma mesa e o domínio dos convidados, Z, é um número de 1 a Tmp(número de convidados), sendo assim representados cada um por um número:

domain(Tables,1,N),

domain(Z,0,Tmp)

* 1. Restrições:

As restrições, tal como explicado acima no ponto 2, servem para posicionar as pessoas num dos lugares da mesa.

Assim, através de restrições fizemos com que em primeiro lugar, pessoas da família de um lado, e pessoas da família do outro ficassem separadas. Depois, tentamos com restrições que as pessoas presentes na mesa de cada convidado tivessem uma ligação forte com o mesmo (verificando a lista de listas com afinidades).

Para além das restrições, relacionadas com a companhia de cada convidado, existe a restrição para o tamanho da mesa que é entre um máximo e um mínimo atribuído.

Assim, o programa garante que as condições de disposição dos convidados sejam feitas ao agrado de todos, utilizando os recursos disponíveis.

(FALTA FOTO COM CODIGO PARA AQUI)

* 1. Função de Avaliação:

Não achamos necessária a implementação de uma função de avaliação visto que as relações são dadas previamente numa lista de lista com as afinidades, logo o erro é menor. Para além disso, seria relativo considerar se a pessoa A tem mais afinidades com a C ou a B em alguns aspetos.

* 1. Estratégia de Pesquisa:

A estratégia de pesquisa utilizada é percorrer uma lista de listas que tem a relação de um convidado com todos os outros e aplicar as restrições para obter a mesa que vai ficar.

1. Visualização da Solução:

Para visualização da solução, e das mesas preenchidas com os convidados, usamos um ciclo for que percorre o número de mesas e vai dando print dos convidados presentes em cada, que são guardados previamente numa lista de listas, onde o primeiro elemento é uma lista com os convidados da primeira mesa.

(POR IMAGEM DA SOLUÇÃO)

1. Resultados:

Reproduzimos o problema para a existência de apenas um número diferentes de mesas, e obtivemos diferentes resultados no que toca ao tempo que o programa demora a encontrar solução:

(Ver tempo para cada N mesas e criar tabela)

1. Conclusões:

Para concluir este projeto, podemos afirmar que a utilização de Prolog para certos problemas complexos, facilita o trabalho do programador, apesar de este não ser um problema de fácil resolução. No entanto, no que toca a outras tarefas mais simples, como mostrar a representação das mesas de uma forma mais interativa, se tornar bastante mais complicado comparado com outras linguagens.