Estratégia Empresarial

Relatório Final

Resolução de Problema de Desição usando Programação em Lógica com Restrições

Programação em Lógica

3^o **ano**
Mestrado Integrado em Engenharia Informática e Computação

Resumo: Este projeto teve como objetivo desenvolver um programa de Programação em lógica com restrições que permitisse maximizar a situação empresarial consoante as medidas disponíveis. Utilizou-se o sistema de desenvolvimento SICStus Prolog, que inclui um módulo de resolução de restrições sobre domínios finitos. Deste modo, foi possível chegar a uma solução ótima num curto espaço de tempo. Com a elaboração deste trabalho, foi reforçada a aprendizagem dos paradigmas de programação em lógica com restrições.

Turma 1 - Grupo Estratégia Empresarial_3

Luís Cruz
up201303248@fe.up.pt

22 de Dezembro de 2017

1 Introdução

O objetivo deste trabalho era implementar uma resolução de um problema de optimização em Prolog com restrições. Decidi resolver um problema de optimização de estratégia empresarial. O problema consiste num conjunto de medidas contituídas de critérios. Este artigo tem como objetivo a análise da implementação e estudo da eficiência e dos resultados obtidos, nele será descrito a abordagem ao problema, isto é, as variáveis de decisão, restrições e estratégia de pesquisa. Posteriormente é abordada a Visualização da solução e análise de resultados.

2 Descrição do Problema

Este problema de optimização consiste num conjunto de N medidas, sendo estas constituídas por M critérios. Cada medida tem-lhe associada um custo. Cada critério tem uma prioridade associada. Dado um determinado orçamento, deve-se fazer a escolha das melhores medidas, cuja soma dos seus custos não ultrapasse o orçamento fornecido.

3 Abordagem

Na implementação deste problema foi decidido representar as medidas disponíveis como uma lista de listas, em que a dimensão de cada lista representa o seu número de critérios. As prioridades dos respetivos critérios são representados por outra lista, sendo a soma dos seus valores igual a 1. Os custos de cada medida são também representados por uma lista, com o tamanho igual ao número de medidas a considerar.

```
medidas_5x5([[4,-3,1,-9,-4],
             [-10,-1,1,9,9],
             [-8,-1,9,1,-2],
             [1,6,3,8,-3],
             [-6,-3,0,-2,-10]
            ]).
```

Figura 1: Lista de medidas com cinco critérios

```
%emp(Opção)
emp(1):-
      medidas_5x5(Medidas),
      custos_x5(Custos),
      %main(Medidas,Prioridades,Custos,Orcamento)
      main(Medidas,[2,3,1,2,1],Custos,2000).
```

3.1 Variaveis de Decisão

Com esta representação em mente, a solução pretendida pode tomar a forma de uma lista simples, com tamanho variável, representando as melhorias

escolhidas para a melhoria da situação empresarial. Deste modo temos, uma lista **Vars** de tamanho N em que N é o número de medidas a considerar. O **domínio** das variáveis vai de **0** a **1**. Para calcular as melhorias foi implementado o predicado *calcMelhorias()* que recebe as melhorias a considerar e as prioridades dos critérios. De seguida, utilizou-se o predicado *scalar_product* para calcular melhoria final e o seu custo.

```
main(Medidas,Prioridades,Custos,Orcamento):-
        length(Medidas,L),
        length(Vars,L),
        domain(Vars,0,1),

        calcMelhorias(Medidas,Prioridades,Melhorias),

        scalar_product(Melhorias,Vars,#=,Escolhas),
        scalar_product(Custos,Vars,#=,Total),
```

3.2 Restrições

A resolução deste problema considera as seguintes restrições:

1. O produto escalar dos critérios e respetivas prioridasdes tem de ser positivo;
2. A soma dos custos das medidas escolhidas não deve ultrapassar o orçamento definido.

Restrição 1: O producto escalar dos critérios e prioridasdes tem de ser positivo

Uma melhoria é considerada como o produto escalar entre os critérios e as prioridades. Estes valores podem ser negativos ou positivos, visto que os critérios podem tomar valores negativos. Como o objetivo do problema é a maximização da melhoria, esta deve ter um valor positivo.

Restrição 2: A soma dos custos das medidas escolhidas não deve ultrapassar o orçamento definido
A soma dos custos das medidas escolhidas não deve ser superior ao orçamento.

3.3 Estratégia de Pesquisa

Ao nível da estratégia de pesquisa de soluções, foram consideradas as condições de *bisect* e *enum*. Entre estas opções foi escolhida a *enum*, visto ser aquela que resulta num tempo mais pequenos, em relação às restantes e ao *default*. Para além destas opções é também é utilizada a opção de *maximize(Escolhas)* para maximizar a melhoria obtida.

4 Visualização de Solução

Para a visualização da solução ao problema após o *labeling* é impresso na consola a lista **Vars** das medidas escolhidas, de seguida as melhorias disponiveis e os custos destas. São de seguida impressas as medidas escolhidas, para se ter uma referência, a melhoria da situação empresarial e o respetivo custo dessa melhoria. A visualização das medidas escolhidas é feita pelo predicado **printResultado()** recebe a lista de **Vars** e a das **Medidas**, imprimindo na consola todas a medidas escolhidas, ou seja, aquelas cujo valor em **Vars** seja **1**.

```
Vars = [0,1,0,1,0]
Melhorias = [-22,5,-10,36,-35]
Custos = [100,300,200,500,720]
Escolhas:
Medida 1: [-10,-1,1,9,9]
Medida 2: [1,6,3,8,-3]
Melhoria Final = 41
Total = 800
yes
```

5 Resultados

Para a visualização dos resultados, foram utilizadas diferentes estratégias de pesquisa, para comparação de tempos e consequentemente melhorar a eficiência. Deste modo, seguem-se três tabelas com resultados de desempenho com estratégias de pesquisa diferentes no *labeling* de **Escolhas**.

Dimensão	Tempo	Retomadas	Envolvimentos	Podas	Retrocessos	Restrições Criadas
5x5	0.001	13	6	33	3	2
5x10	0.017	23	12	55	1	2
10x5	0.000	12	6	46	6	2
20x5	0.018	57	11	265	17	4
20x10	0.000	119	29	409	23	2
50x5	0.006	1234	128	6221	363	2

Tabela 1: Tabela de Tempos com bisect

Dimensão	Tempo	Retomadas	Envolvimentos	Podas	Retrocessos	Restrições Criadas
5x5	0.017	13	6	29	3	2
5x10	0.000	23	12	50	1	2
10x5	0.001	12	6	39	6	2
20x5	0.001	55	11	238	17	2
20x10	0.019	119	29	387	23	2
50x5	0.010	1234	128	5848	363	2

Tabela 2: Tabela de Tempos com enum

Dimensão	Tempo	Retomadas	Envolvimentos	Podas	Retrocessos	Restrições Criadas
5x5	0.001	13	6	29	3	2
5x10	0.000	23	12	50	1	2
10x5	0.000	12	6	39	6	2
20x5	0.000	55	11	243	17	2
20x10	0.001	119	29	409	23	2
50x5	0.006	1234	128	6221	363	2

Tabela 3: Tabela de Tempos com opções em default

6 Conclusões

A utilização de Programação em Lógica com Restrições aumenta significativamente a eficiência da resolução de problemas como os de otimização. Como é possível verificar, com diferentes dimensões do problema é possível chegar a uma solução num tempo significativamente baixo. A execução do trabalho em si, não mostrou demasiada complexidade. Grande parte do tempo dedicado a este trabalho foi gasto na planificação e idealização da sua execução.

Referências

[1] Random Labeling, `https://stackoverflow.com/questions/8693788/prolog-random-labeling`, 12 12 2017.

[2] SICStus Prolog, `https://sicstus.sics.se/`, 20 10 2016.

[3] SWI-Prolog, `http://www.swi-prolog.org/`, 22 12 2017.

A Código fonte

A.1 emp.pl

```
1  :-use_module(library(system)).
2  :-use_module(library(clpfd)).
3  :-use_module(library(random)).
4  :-use_module(library(lists)).
5  :-use_module(library(aggregate)).
6  :-include('empresaTest.pl').
7
8  % variavel com tamanho igual ao numero de medidas, com dominio
       entre 0 e 1. 1 quer dizer que toma essa medida,
9  % a soma dos custos deve ser menor ou igual ao orcamento.
10 main(Medidas,Prioridades,Custos,Orcamento):-
11         length(Medidas,L),
12         length(Vars,L),
13         domain(Vars,0,1),
14
15         calcMelhorias(Medidas,Prioridades,Melhorias),
16
17         scalar_product(Melhorias,Vars,#=,Escolhas),
18         scalar_product(Custos,Vars,#=,Total),
19
20         Escolhas #> 0,
21         Total #=< Orcamento,
22         statistics(walltime,_),
23         labeling(maximize(Escolhas),Vars),
24         statistics(walltime, [_, ElapsedTime | _]),
25         format('An answer has been found!~nElapsed time: ~3d
               seconds', ElapsedTime), nl,
26         fd_statistics,nl,
27         resultado(Vars,Medidas,Resultado),
28         write('Vars = '),write(Vars),nl,
29         write('Melhorias = '),write(Melhorias),nl,
30         write('Custos = '),write(Custos),nl,
31         write('Escolhas: '),nl,printResultado(Resultado,1),
32         write('Melhoria Final = '),write(Escolhas),nl,
33         write('Total = '),write(Total),nl.
34
35
36 printResultado([Medida | Resto],N):-
37         write('Medida '),write(N),write(': '),write(Medida),nl,
38         N1 is N+1,
39         printResultado(Resto,N1).
40 printResultado([],_).
41
42 resultado([],[],[]).
43 resultado([Var | Vars],[Medida | Medidas],Resultado):-
44         resultado(Vars,Medidas,NovoResultado),
45         Var == 1,
46         Resultado = [Medida | NovoResultado].
47
48 resultado([_ | Vars],[_ | Medidas],NovoResultado):-
49         resultado(Vars,Medidas,NovoResultado).
50
51 calcMelhorias([],_,[]).
52 calcMelhorias([Medida | Resto],Prioridades,Melhorias):-
53         calcMelhorias(Resto,Prioridades,NovaMelhoria),
54         scalar_product(Medida,Prioridades,#=,Resultado),
55         Melhorias = [Resultado | NovaMelhoria].
```

A.2 empresaTest.pl

```
1  %emp(Opcao)
2  emp(1):-
3        medidas_5x5(Medidas),
4        custos_x5(Custos),
5        %main(Medidas,Prioridades,Custos,Orcamento)
6        main(Medidas,[2,3,1,2,1],Custos,2000).
7
8  emp(2):-
9        medidas_5x10(Medidas),
10       custos_x5(Custos),
11       main(Medidas,[1,1,1,1,1,1,1,1,1,1],Custos,2000).
12
13 emp(3):-
14       medidas_10x5(Medidas),
15       custos_x10(Custos),
16       main(Medidas,[2,3,1,2,1],Custos,2000).
17
18 emp(4):-
19       medidas_20x5(Medidas),
20       custos_x20(Custos),
21       main(Medidas,[2,3,1,2,1],Custos,2000).
22
23 emp(5):-
24       medidas_20x10(Medidas),
25       custos_x20(Custos),
26       main(Medidas,[1,1,1,1,1,1,1,1,1,1],Custos,2000).
27
28 emp(6):-
29       medidas_50x5(Medidas),
30       custos_x50(Custos),
31       main(Medidas,[2,3,1,2,1],Custos,2000).
32
33
34
35 medidas_5x5([[4,-3,1,-9,-4],
36               [-10,-1,1,9,9],
37               [-8,-1,9,1,-2],
38               [1,6,3,8,-3],
39               [-6,-3,0,-2,-10]
40             ]).
41
42 medidas_10x5([[4,-3,1,-9,-4],
43               [-10,3,1,-9,9],
44               [-8,-1,9,2,-3],
45               [1,-6,3,8,-3],
46               [-6,-3,0,-2,-10],
47               [2,-3,5,-9,-4],
48               [-10,3,1,-2,9],
49               [5,-1,9,1,-3],
50               [1,-2,6,8,-3],
51               [-6,-3,2,-2,10]
52             ]).
53
54 medidas_20x5([[0,6,-6,-5,9],
55                [5,-9,-1,1,6],
56                [0,6,-6,-5,9],
57                [-6,3,6,2,-9],
58                [5,-9,-1,1,6],
59                [0,6,-6,-5,9],
60                [10,-10,3,2,-8],
```

```
                [-6,3,6,2,-9],
                [5,-9,-1,1,6],
                [0,6,-6,-5,9],
                [-5,4,1,10,5],
                [10,-10,3,2,-8],
                [-6,3,6,2,-9],
                [5,-9,-1,1,6],
                [0,6,-6,-5,9],
                [-3,1,3,-4,-7],
                [-5,4,1,10,5],
                [10,-10,3,2,-8],
                [-6,3,6,2,-9],
                [5,-9,-1,1,6]
               ]).

medidas_50x5([[10,3,9,0,2],
                [-2,-5,-6,1,3],
                [10,3,9,0,2],
                [8,0,-7,2,6],
                [-2,-5,-6,1,3],
                [10,3,9,0,2],
                [2,-6,3,-1,5],
                [8,0,-7,2,6],
                [-2,-5,-6,1,3],
                [10,3,9,0,2],
                [-4,10,5,7,4],
                [2,-6,3,-1,5],
                [8,0,-7,2,6],
                [-2,-5,-6,1,3],
                [10,3,9,0,2],
                [-9,7,-4,9,-2],
                [-4,10,5,7,4],
                [2,-6,3,-1,5],
                [8,0,-7,2,6],
                [-2,-5,-6,1,3],
                [10,3,9,0,2],
                [-1,-5,-3,-8,4],
                [-9,7,-4,9,-2],
                [-4,10,5,7,4],
                [2,-6,3,-1,5],
                [8,0,-7,2,6],
                [-2,-5,-6,1,3],
                [10,3,9,0,2],
                [8,-4,-8,9,-6],
                [-1,-5,-3,-8,4],
                [-9,7,-4,9,-2],
                [-4,10,5,7,4],
                [2,-6,3,-1,5],
                [8,0,-7,2,6],
                [-2,-5,-6,1,3],
                [10,3,9,0,2],
                [7,2,-1,-6,4],
                [8,-4,-8,9,-6],
                [-1,-5,-3,-8,4],
                [-9,7,-4,9,-2],
                [-4,10,5,7,4],
                [2,-6,3,-1,5],
                [8,0,-7,2,6],
                [-2,-5,-6,1,3],
                [10,3,9,0,2],
                [6,-7,9,3,2],
                [7,2,-1,-6,4],
```

```
123                [8,-4,-8,9,-6],
124                [-1,-5,-3,-8,4],
125                [-9,7,-4,9,-2]
126               ]).
127
128  medidas_20x10([[6,1,-6,3,-2,5,9,-1,-8,2],
129                 [3,-9,8,-4,9,-2,-5,10,5,6],
130                 [6,1,-6,3,-2,5,9,-1,-8,2],
131                 [4,8,-4,-9,9,-6,-1,-3,-2,-8],
132                 [3,-9,8,-4,9,-2,-5,10,5,6],
133                 [6,1,-6,3,-2,5,9,-1,-8,2],
134                 [-3,6,-7,9,3,2,7,0,-2,-8],
135                 [4,8,-4,-9,9,-6,-1,-3,-2,-8],
136                 [3,-9,8,-4,9,-2,-5,10,5,6],
137                 [6,1,-6,3,-2,5,9,-1,-8,2],
138                 [-1,2,-9,-6,0,4,-2,-7,6,-10],
139                 [-3,6,-7,9,3,2,7,0,-2,-8],
140                 [4,8,-4,-9,9,-6,-1,-3,-2,-8],
141                 [3,-9,8,-4,9,-2,-5,10,5,6],
142                 [6,1,-6,3,-2,5,9,-1,-8,2],
143                 [-1,-9,-3,-7,1,8,6,5,9,-8],
144                 [-1,2,-9,-6,0,4,-2,-7,6,-10],
145                 [-3,6,-7,9,3,2,7,0,-2,-8],
146                 [4,8,-4,-9,9,-6,-1,-3,-2,-8],
147                 [3,-9,8,-4,9,-2,-5,10,5,6]
148                ]).
149
150  medidas_5x10([[4,-3,1,-9,-4,3,2,-3,5,7],
151                [-10,-1,1,9,8,4,2,-3,4,-1],
152                [-8,-1,9,1,-2,8,-4,5,1,-3],
153                [1,6,3,8,-3,3,-3,1,7,4],
154                [-6,-3,0,-2,-10,1,-3,5,-6,3]
155               ]).
156
157  custos_x5([100,300,200,500,720]).
158  custos_x10([100,1300,200,500,720,300,800,1200,450,2000]).
159  custos_x20
         ([433,1269,1525,798,479,661,1695,1372,542,1660,531,1957,549,1368,63,424,1320,523,867,1
         .
160  custos_x50
         ([1892,999,522,1848,1645,616,829,1099,1632,471,1677,886,773,999,1557,1447,196,1255,956
         .
161
162  gerarMedidas(0,_,[]).
163  gerarMedidas(NumMedidas,NumCriterios,Medidas):-
164          length(Criterios,NumCriterios),
165          all_distinct(Criterios),
166          domain(Criterios, -10, 10),
167          labeling([value(enume)],Criterios),
168          Medidas = [Criterios | NovoMedidas],
169          Num1 is NumMedidas - 1,
170          gerarMedidas(Num1,NumCriterios,NovoMedidas),
171          write(Medidas).
172
173  gerarCustos(NumCustos,Custos):-
174           length(Custos,NumCustos),
175           domain(Custos,1,2000),
176           labeling(value(enume),Custos),
177           write(Custos).
178
179  enume(Var,_,BB0,BB):-
180          fd_set(Var,Set),
```

```
181         select_best_value(Set,Value),
182         (
183            first_bound(BB0,BB), Var #= Value;
184            later_bound(BB0,BB), Var #\= Value
185         ).
186
187  select_best_value(Set,BestValue):-
188           fdset_to_list(Set,Lista),
189           length(Lista,Len),
190           random(0,Len,RandomIndex),
191           nth0(RandomIndex,Lista,BestValue).
```