Aplicações de Programação com Restrições

Denise Guarnieri Batista and Arnaldo Vieira Moura
{deniseg,arnaldo}@dcc.unicamp.br

Relatório Técnico IC–99-13

Dezembro de 1999

Aplicações de Programação com Restrições

Denise Guarnieri Batista and Arnaldo Vieira Moura
{deniseg,arnaldo}@dcc.unicamp.br *

Resumo

Este documento descreve um modelo de programação por restrições, aplicado ao problema de escalonamento mensal dos profissionais de enfermagem no Hospital Universitário da Unicamp. Sendo mais de 1500 profissionais, esta tarefa, se executada manualmente, consome muito tempo, é propícia a êrros seguidos e, seguramente, não otimiza a escala, nem em termos do número de pessoas alocadas, nem em termos da carga de trabalho individual. Além do modelo propriamente dito, este trabalho oferece também uma primeira versão para uma interface gráfica, que facilite o manuseio do modelo por pessoal não qualificado em informática. O modelo provou que toda escala, se construída dentro das restrições e exigências impostas pela direção do Hospital, é inviável. Como extensão, é proposta revisão do modelo, de forma que atenda às exigências de qualidade impostas pelo Hospital, às expensas do uso de horas extras. Este relatório é baseado no Trabalho de Graduação da aluna Denise G. Batista, formada Engenheira de Computação, na Unicamp, em 1999.

*Instituto de Computação, Universidade Estadual de Campinas, 13081-970 Campinas, SP.

Sumário

1 Introdução

Programação com restrições (*constraint satisfaction programming*) é um novo paradigma de programação que vem ganhando relevância cada vez maior tanto no meio acadêmico quanto no meio industrial. Trata-se de um mecanismo de programação particularmente adequado ao tratamento de problemas de otimização complexos. Sua aplicabilidade estende-se a várias áreas importantes, tais como escalonamento e roteamento.

O paradigma de programação com restrições, quando aliado a um ambiente de otimização (*solver*) adequado, tem se mostrado uma ferramenta versátil e eficaz para modelagem de problemas combinatórios difíceis. A codificação e a execução desses modelos em ambientes de programação com restrições comercialmente disponíveis tem produzido resultados bastante animadores.

O objetivo deste projeto final de graduação foi proporcionar à aluna conhecimento e experiência prática na área de programação com restrições. Para isso, o projeto constituiu-se de duas vertentes concomitantes. A primeira vertente focalizou a área de fundamentos; a segunda contemplou a solução de um problema real em um ambiente de programação com restrições.

2 Aplicação

Os fundamentos teóricos adquiridos (principalmente de programação com restrições) foram colocados em prática na resolução de um problema real. O problema consiste da geração de escalas mensais de trabalho de diversos profissionais para cada enfermaria e cada turno do Hospital Universitário da Universidade Estadual de Campinas, respeitando-se diversas restrições operacionais. O problema escolhido foi modelado e executado no ambiente Eclipse, já disponível no Instituto de Computação da Unicamp.

2.1 Descrição

O Hospital Universitário é dividido em 48 enfermarias. Cada uma delas possui uma escala de trabalho independente, dividida em três turnos (manhã, tarde e noite). A escala de cada turno é independente. Foram considerados apenas os turnos de manhã e tarde, já que o turno da noite possui algumas restrições particulares.

Os profissionais divididem-se entre enfermeiros, auxiliares de enfermagem, técnicos de enfermagem, atendentes e maqueiros. Cada profissional atua em somente uma dessas áreas; não é possivel, por exemplo, substituir um enfermeiro por um auxiliar de enfermagem (e vice-versa).

A escala de trabalho de um mês para uma enfermaria e um determinado turno atribui, para cada um dos dias do mês, um "dia de trabalho", uma "folga"ou um "impedimento"para cada profissional. A categoria "impedimento"inclui os casos de licença médica/gestante, aposentadoria, dispensa sindical e férias.

O número de folgas durante o ano é calculado de acordo com o número total de finais de semana e feriados naquele ano. Esse total é então distribuído de maneira uniforme pelos 12 meses do ano. O número de folgas atribuído a cada profissional em um dado mês é função do número de folgas regulares naquele mês, e também do número de dias que o profissional está impedido de trabalhar. Portanto, o número de folgas e impedimentos pode diferir de um profissional para outro. Deve-se distribuir essas folgas na escala de trabalho de cada profissional sem desrespeitar as restrições descritas na seção seguinte.

2.2 Restrições

2.3 Retrição Básica

A cada dia do mês e cada profissional deve ser associada uma, e exatamente uma, categoria dentre as opções "dia de trabalho", "folga"e "impedimento".

2.3.1 Número de Folgas e Impedimentos

O número de folgas e impedimentos a que um profissional tem direito durante o mês deve corresponder ao número de dias em que lhe foram atribuídas as categorias "folga"e "impedimento", respectivamente.

2.3.2 Número Mínimo de Profissionais

Cada enfermaria requer um número mínimo de profissionais de cada categoria, de acordo com o turno e dia da semana (dia útil vs. finais de semana e feriados), para funcionar adequadamente. Estes números são dados do problema e estão apresentados nas tabelas abaixo para uma enfermaria típica.

Serviço de Enfermagem Pediátrica:

profissional	dias úteis	finais de semana e feriados
enfermeiros	3	1
auxiliares	9	8
técnicos	0	0
atendentes	2	2
maqueiros	0	0

Figura 1: Manhã

profissional	dias úteis	finais de semana e feriados
enfermeiros	3	1
auxiliares	11	10
técnicos	0	0
atendentes	0	0
maqueiros	0	0

Figura 2: Tarde

2.3.3 Preferência de Dias de Folga

O número de folgas a que cada profissional tem direito a cada mês é fixo e é um dado do problema. Aos profissionais é dada a oportunidade de escolher algumas de suas folgas, desde que a escala "parcial"resultante não inviabilize o respeito à restrição de número mínimo de profissionais por dia, descrita na seção 2.3.2. A escala apresentada como solução deve respeitar essas preferências

de dias de folga — ou deve ser indicado que não há solução viável, isto é, uma solução que respeite as preferências marcadas, além de satisfazer a todas as demais restrições do problema.

2.3.4 Folga a Cada 7 Dias

Nos turnos da manhã e da tarde, um profissional não pode trabalhar por mais de 6 dias consecutivos.

2.3.5 Folga no Domingo

Nos turnos da manhã e da tarde, em pelo menos um domingo por mês o profissional não deve tabalhar.

2.3.6 Dia de Reunião

Nos turnos da manhã e da tarde, em pelo menos um dia por mês nenhum profissional deve estar de folga. Não há problema se nesse dia algum profissional estiver impedido de trabalhar. Esse dia não precisa coincidir para os dois turnos ou para todas as enfermarias.

2.4 Interface

2.4.1 Dados de Entrada

Os dados de entrada devem ser fornecidos através de um arquivo, cujo formato será descrito a seguir. A partir desse arquivo serão geradas as restrições para o programa em Eclipse (vide seção 2.6).

- número de dias no mês

  Se X é o número de dias no mês, a primeira linha do arquivo deve ser:

  ```
  dmes X
  ```

  Pode haver comentários após X, pois apenas as duas primeiras "palavras"da linha serão consideradas.

- dia na semana do primeiro dia do mês

  Deve-se acrescentar ao arquivo a seguinte linha:

  ```
  dsemana X
  ```

  X é o número correspondente ao dia na semana do primeiro dia do mês, de acordo com a seguinte tabela:

dia na semana do primeiro dia do mês	X
domingo	1
segunda-feira	2
terça-feira	3
quarta-feira	4
quinta-feira	5
sexta-feira	6
sábado	7

Pode haver comentários após X, pois apenas as duas primeiras "palavras"da linha serão consideradas.

- feriados no mês

  Se os dias X, Y e Z do mês são feriados, deve-se acrescentar a seguinte linha ao arquivo:

  ```
  feriados X Y Z 0
  ```

  Entre "feriados"e "0"deve haver tantos números quantos forem os feriados no mês. A linha é obrigatória; portanto, se o mês não tiver nenhum feriado, acrescenta-se apenas:

  ```
  feriados 0
  ```

  Pode haver comentários após o "0", pois serão desconsiderados.

- número de profissionais da enfermaria, por categoria

  Esses números devem ser fornecidos na seguinte ordem: enfermeiros, auxiliares de enfermagem, técnicos de enfermagem, atendentes e maqueiros, uma linha para cada um. Portanto acrescenta-se as seguintes linhas:

  ```
  nenf V
  naux W
  ntec X
  nate Y
  nmaq Z
  ```

  V, W, X, Y, Z são os números de enfermeiros, auxiliares, técnicos, atendentes e maqueiros, respectivamente. A ordem das linhas deve ser exatamente essa. Se não há um determinado tipo de profissional na enfermaria (por exemplo, maqueiros), basta substituir a letra correspondente (no caso, Z) por "0". Pode haver comentários após os números, pois apenas as duas primeiras "palavras"de cada linha serão consideradas.

- número mínimo de profissionais, por dia e por categoria

  As categorias devem ser ordenadas como descrito anteriormente (enfermeiros, auxiliares, técnicos, atendentes e maqueiros). Para cada categoria deve-se apresentar o mínimo de profissionais que deve estar trabalhando em dias úteis e não úteis, nesta ordem, uma linha para cada um. Portanto acrescenta-se as seguintes linhas:

  ```
  minenfU VU
  minenfN VN
  minauxU WU
  minauxN WN
  mintecU XU
  mintecN XN
  minateU YU
  minateN YN
  minmaqU ZU
  minmaqN ZN
  ```

  VU é o mínimo de enfermeiros em dias úteis, VN é o mínimo de enfermeiros em dias não úteis, e assim por diante. A ordem das linhas deve ser exatamente essa. Se não houver

necessidade de um determinado tipo de profissional na enfermaria (por exemplo, maqueiros em dias não úteis), basta substituir a letra correspondente (no caso, ZN) por "0". Pode haver comentários após os números, pois apenas as duas primeiras "palavras"de cada linha serão consideradas.

- dias em que cada profissional está impedido de trabalhar

  Para cada profissional acrescenta-se a seguinte linha:

  ```
  imp X Y Z 0
  ```

  (X, Y, Z são os dias em que esse profissional está impedido de trabalhar.) Entre "imp"e "0"deve haver tantos números quantos forem os dias em que o profissional está impedido de trabalhar. A linha é obrigatória; portanto, se o profissional não está impedido de trabalhar em nenhum dia, acrescenta-se apenas:

  ```
  imp 0
  ```

  Deve haver tantas linhas iniciadas com "imp"quanto o número de profissionais na enfermaria. As categorias devem ser ordenadas como descrito acima, i.e., em primeiro lugar vêm as linhas referentes aos enfermeiros, em seguida as linhas referentes aos auxiliares, e assim por diante. Pode haver comentários após o "0", pois serão desconsiderados.

- número de dias de folga de cada profissional

  Para cada profissional acrescenta-se a seguinte linha:

  ```
  f X
  ```

  X é o número de folgas que deve ser atribuído a este profissional. A ordem dos profissionais deve ser exatamente igual à utilizada com relação aos impedimentos. Pode haver comentários após os números, pois apenas as duas primeiras "palavras"de cada linha serão consideradas.

- número de dias consecutivos em que cada profissional trabalhou no final do mês anterior

  Para cada profissional acrescenta-se a seguinte linha:

  ```
  prev X
  ```

  X é o número de dias consecutivos em que este profissional trabalhou no final do mês anterior. A ordem dos profissionais deve ser exatamente igual à utilizada com relação aos impedimentos. Pode haver comentários após os números, pois apenas as duas primeiras "palavras"de cada linha serão consideradas.

- preferências de folga de cada profissional

  Para cada profissional acrescenta-se a seguinte linha:

  ```
  pref X Y Z 0
  ```

  (X, Y, Z são os dias em que esse profissional prefere ter folga.) A linha é obrigatória; portanto, se o profissional não tem preferência de folga por nenhum dia, acrescenta-se apenas:

  ```
  pref 0
  ```

  A ordem dos profissionais deve ser exatamente igual à utilizada com relação aos impedimentos. Pode haver comentários após o "0", pois serão desconsiderados.

A ordem das informações no arquivo de entrada deve ser exatamente a mesma da apresentada nesta seção para que o arquivo de restrições seja gerado corretamente. A única exceção são as linhas de comentários, iniciadas pelo símbolo de porcentagem (%), que são ignoradas e podem aparecer em qualquer posição do arquivo. O arquivo não deve apresentar nenhuma linha em branco.

O apêndice A.1 apresenta, como exemplo, o arquivo de entrada para um problema reduzido (geração de uma escala de 10 dias de trabalho para uma enfermaria fictícia).

2.4.2 Utilização

Para se utilizar o programa deve-se proceder da forma descrita abaixo.

Em primeiro lugar, cria-se um arquivo de entrada como descrito na seção anterior (seção 2.4.1). Ao longo desta seção supõe-se que o nome desse arquivo de entrada é `arq1` e que ele está gravado no mesmo diretório do arquivo `Filtro.java` (vide seção 2.6). Supõe-se ainda que tanto Java quanto o Eclipse estejam corretamente instalados na máquina.

Deve-se gerar o arquivo `Filtro.class` a partir de `Filtro.java`. Para isso executa-se o comando:

```
javac Filtro.java
```

Em seguida deve-se gerar o arquivo com restrições, que servirá de entrada para o Eclipse. Supõe-se que a este arquivo deseja-se atribuir o nome `arq2`. Então digita-se, de dentro do diretório que contém `arq1`, na linha de comando:

```
java Filtro arq1 > arq2
```

Após a criação do arquivo com restrições deve-se chamar o Eclipse. Supondo-se que o eclipse esteja instalado no diretório `/usr/local/bin/`, digita-se na linha de comando:

```
/usr/local/bin/eclipse
```

Para compilar o arquivo com restrições digita-se (no Eclipse):

```
[arq2].
```

Para executar o programa digita-se em seguida:

```
programa.
```

A seguinte informação indica que não há solução viável:

```
no (more) solution.
```

Se houver solução, a escala encontrada pelo programa será apresentada na tela. O apêndice A.4 apresenta a saída do programa para o problema reduzido citado anteriormente. Na escala apresentada, a ordenação de profissionais dentro de cada categoria corresponde à ordenação utilizada no arquivo de entrada (`arq1`).

Se o programa indicar que não há solução viável recomenda-se repetir o processo a partir de um arquivo de entrada com menos restrições (por exemplo, desconsiderar as preferências de folga de alguns profissionais). Pode-se ainda "simular"a alocação de horas extras através do acréscimo de profissionais não existentes. Por exemplo: acrescenta-se um enfermeiro extra que não está impedido de trabalhar em nenhum dia, cujo número de folgas é igual ao número de dias no mês, menos dois (ou seja, apenas dois dias trabalhando), que não possui preferências de folga, e cujo número de dias consecutivos trabalhados no final do mês anterior é zero. Se a escala tornar-se viável com a presença desse enfermeiro, então é necessário alocar horas extras (no caso, de enfermeiros) em pelo menos um dos dias em que este profissional foi alocado para trabalhar (o outro dia é o dia de reunião, e para este as horas extras talvez não sejam necessárias).

2.5 Modelo

Cada execução do programa definirá a escala mensal (em um determinado mês) de um dos turnos de uma enfermaria.

A escala será representada por variáveis $ESCixj$, onde $1 \leq i \leq np$ (np é o número total de profissionais) e $1 \leq j \leq nd$ (nd é o número de dias no mês). As varíaveis $ESCixj$ têm domínio $\{0, 1, 2\}$ representando as categorias "dia de trabalho"($ESCixj = 0$), "folga"($ESCixj = 1$) e "impedimento"($ESCixj = 2$). O conjunto de variáveis $ESCixj$ será chamado de matriz ESC.

São necessárias ainda duas outras matrizes binárias: $ESCBIN$ e $ESCUTIL$. $ESCBIN$ possui as mesmas dimensões que ESC mas as variáveis $ESCBINixj$ têm domínio binário ($\{0, 1\}$). Neste caso $ESCBINixj = 0$ significa "dia de trabalho"e $ESCBINixj = 1$ significa "folga"ou "impedimento". Já $ESCUTIL$ contém apenas as escalas dos dias úteis, portanto $ESCUTILixj$ existe apenas se j é um dia útil. O domínio das variáveis $ESCUTILixj$ é o mesmo de $ESCBIN$ ($\{0, 1\}$), mas neste caso $ESCUTILixj = 0$ significa "dia de trabalho"ou "impedimento"e $ESCUTILixj = 1$ significa "folga".

Em todas as matrizes, será mantida a seguinte ordem: em primeiro lugar, as fileiras relativas a enfermeiros. Em seguida, auxiliares de enfermagem, técnicos de enfermagem, atendentes e maqueiros, nesta ordem.

As matrizes ESC, $ESCBIN$ e $ESCUTIL$ são ligadas com restrições que garantem as seguintes relações:

trabalho	$ESCixj = 0$	$ESCBINixj = 0$	(se j é dia útil) $ESCUTILixj = 0$
folga	$ESCixj = 1$	$ESCBINixj = 1$	(se j é dia útil) $ESCUTILixj = 1$
impedimento	$ESCixj = 2$	$ESCBINixj = 1$	(se j é dia útil) $ESCUTILixj = 0$

As restrições foram modeladas da seguinte forma:

- Restrição Básica (vide seção 2.3)

  Atendida automaticamente, já que a cada variável deve ser atribuído exatamente um valor do domínio.

- Número de Folgas e Impedimentos (vide seção 2.3.1)

  Se o profissional i está impedido de trabalhar no dia j, então à variável $ESCBINixj$ é atribuído valor 1, forçando-o a não trabalhar neste dia. Por outro lado, para cada profissional i acrescenta-se a restrição

  $$\sum_{j=1}^{nd} ESCBINixj = F + I$$

  (F é o número de folgas e I é o número de dias em que o profissional i está impedido de trabalhar nesse mês.) Dessa forma respeita-se o número de folgas e impedimentos.

- Número Mínimo de Profissionais (vide seção 2.3.2)

  Seja $MinProfDia$ o número mínimo de profissionais do tipo $Prof$ necessários na enfermaria — $Prof$ deve ser substituído por Enf (enfermeiro), Aux (auxiliar de enfermagem), Tec (técnico de enfermagem), Ate (atendente) ou Maq (maqueiro); e Dia deve ser substituído por um número entre 1 e nd. $MinProfDia$ leva em conta se Dia é um dia útil ou não.

Para cada dia do mês j e tipo de profissional $Prof$ acrescenta-se a restrição

$$nProf - \sum_{i=S}^{S+nProf-1} ESCBINixj \geq MinProfj$$

S indica a primeira fileira, na matriz, que representa a escala de um profissional do tipo $Prof$ (se $Prof = Enf$, $S = 1$; se $Prof = Aux$, $S = 1 + nEnf$ ($nEnf$ é o número de enfermeiros); e assim por diante). Dessa forma respeita-se o número mínimo de profissionais em cada dia.

- Preferência de Dias de Folga (vide seção 2.3.3)

  Se o profissional i tem preferência por uma folga no dia j, então à variável $ESCBINixj$ é atribuído valor 1, forçando-o a ter folga neste dia.

- Folga a Cada 7 Dias (vide seção 2.3.4)

  Para cada profissional i estabelecem-se dois tipos de restrição. Em primeiro lugar,

  $$\sum_{j=1}^{7-prev} ESCBINixj > 0$$

  ($prev$ é o número de dias consecutivos em que o profissional i trabalhou no final do mês anterior.) Ou seja, o profissional não deve trabalhar em pelo menos um dia dentre os X primeiros dias do mês (onde X é dado por $7 - prev$).

  Além disso, para todo $7 \leq j \leq nd$ estabelece-se que

  $$\sum_{j-6}^{j} ESCBINixj > 0$$

  Dessa forma garante-se que cada profissional não trabalhará por mais de 6 dias seguidos, já que qualquer intervalo de 7 dias consecutivos durante o mês contempla pelo menos uma "folga" ou "impedimento" para qualquer profissional.

- Folga no Domingo (vide seção 2.3.5)

  Para cada profissional i acrescenta-se a restrição

  $$\sum_{j \in D} ESCBINixj > 0$$

  forçando-o a não trabalhar em pelo menos um domingo ($j \in D$ se e somente se j é um domingo).

- Dia de Reunião (vide seção 2.3.6)

  O atendimento a esse item requer a criação de duas variáveis, $FolgasN$ e $ExisteFolgaN$, para cada dia útil N. $FolgasN$ reflete o número de funcionários que estão de folga no dia N.

Se N é um dia útil,

$$FolgasN = \sum_{i=1}^{np} ESCUTILixN$$

As variáveis binárias $ExisteFolgaN$ refletem o valor de $FolgasN$: $ExisteFolgaN = 0$ se e somente se $FolgasN = 0$.

Acrescenta-se então a restrição de que a soma de $ExisteFolgaN$, para todo N, deve ser estritamente menor que o número de dias úteis no mês. Dessa forma garante-se que em pelo menos um dia útil todos os profissionais da enfermaria estão ou trabalhando ou impedidos de trabalhar.

2.6 Implementação

As restrições lidas pelo Eclipse são produzidas por um programa filtro a partir de um arquivo produzido pelo usuário (vide seção 2.4.1). Esse programa filtro foi programado em Java e possui dois métodos principais. O primeiro trata da leitura do arquivo do usuário e da inicialização das variáveis; o segundo produz o arquivo de restrições de acordo com os dados obtidos. Não é feito tratamento de erro dos dados de entrada; portanto, se o arquivo do usuário contiver erros, o arquivo de restrições não será gerado corretamente (e não será gerada nenhuma mensagem de erro alertando o usuário desse problema).

O apêndice A.2 apresenta o arquivo fonte `Filtro.java`.

O arquivo de restrições é gerado de acordo com o modelo descrito na seção anterior. A ordenação das restrições procura favorecer a busca — as que mais restringem os domínios das variáveis são listadas no início do arquivo. Optou-se por evitar o uso de predicados definidos pelo usuário (que tendem a tornar a execução mais lenta devido a *backtrackings*) e por utilizar predicados pré-definidos sempre que possível, já que estes são programados em baixo nível e portanto bastante eficientes.

O apêndice A.3 apresenta o arquivo de restrições gerado para um problema reduzido (descrito pelo arquivo de entrada presente no apêndice A.1)

3 Resultados

O projeto de fim de curso descrito neste documento contém duas vertentes: fundamentos e aplicação.

Na parte de fundamentos, o estudo de lógica proposicional e de primeira ordem, além de suprir uma lacuna na formação básica de graduação, proporcionou à aluna o entendimento em maior profundidade dos conceitos relacionados à programação em lógica. Esse conhecimento foi de grande valia quando do estudo da teoria de programação com restrições, utilizada posteriormente para modelagem de um problema real.

Na parte de aplicações, o programa foi testado com dois tipos de dados. A princípio criou-se um problema reduzido – geração de uma escala de trabalho para uma enfermaria fictícia com 8 profissionais, para um período de 10 dias. A solução obtida para esse caso é apresentada na seção A.4.

Foram feitos alguns testes adicionais com variações nas restrições impostas, e observou-se que o programa funcionou corretamente para os mesmos. Nos casos em que a solução era inviável isso

foi indicado pelo programa; além disso, em caso de resposta afirmativa (solução viável), as escalas apresentadas respeitavam as restrições impostas a elas.

Foi realizado ainda um teste utilizando dados reais (geração da escala do mês de maio para o serviço de enfermagem pediátrica). Como esperado, o programa indicou que a solução era inviável, já que o número de atendentes era igual ao mínimo necessário por dia, impossibilitando a atribuição de folgas.

Pode-se concluir que ambas as vertentes do projeto foram bem sucedidas. Por um lado, a aluna adquiriu uma considerável bagagem de conhecimentos adicionais, tanto na parte de fundamentos matemáticos, quanto em relação ao novo paradigma de programação com restrições. Por outro lado, o exercício prático produziu bons resultados, quando aplicado a uma situação real, encontrada em uma das enfermarias do Hospital Universitário da Unicamp.

4 Possíveis Extensões

O programa tal como apresentado neste documento (vide seção 2) pode ser estendido de diversas maneiras. Esta seção procura identificar algumas das extensões mais significativas.

A maioria das enfermarias dispõe de menos profissionais do que o ideal e por isso precisa constantemente reavaliar suas restrições e/ou recorrer a horas extras. Devido a essa característica, quando utilizado com dados reais o programa tende a indicar constantemente que não há solução viável. Dessa forma, seria bastante interessante incluir algum tipo de retorno nas respostas negativas, indicando ao usuário quais restrições tornaram a solução inviável, ou se existe uma alocação de horas extras que resolve o problema. A implementação dessa funcionalidade requereria a realização de algumas reuniões de avaliação com a Diretoria de Enfermagem do Hospital para a definição da estratégia a ser utilizada (por exemplo, se é mais adequado alocar horas extras ou desconsiderar preferências de folga).

Além disso, o modelo poderia ser estendido de forma a contemplar as restrições do turno noturno, que não está sendo tratado.

A usabilidade do programa seria favorecida pela criação de uma interface gráfica de entrada e saída. Essa interface seria responsável pelo envio de dados para o Eclipse e pela apresentação da solução encontrada, se alguma. A interface poderia, ainda, proporcionar as seguintes funcionalidades:

- verificação de consistência dos dados de entrada;
- facilidade de submissão de nova consulta semelhante à anterior (refinamento de consulta);
- impressão do resultado.

Referências

[BC93] F. Benhamou and A. Colmerauer. *Constraint Logic Programming.* MIT Press, 1993.

[BCW] J.H.M. Lee B.M.W. Cheng, K.M.F. Choi and J.C.K. Wu. Increasing constraint propagation by redundant modelling: an experience report. Technical report, Department of Computer Science and Engineering, The Chinese University of Hong Kong.

[CM94] W.F. Clocksin and C.S. Mellish. *Programming in Prolog.* Springer-Verlag, 1994.

[Col87] A. Colmerauer. aug 1987.

[HET84] J. Flum H.D. Ebbinghaus and W. Thomas. *Mathematical Logic.* Springer-Verlag, 1984.

[MS98] Kim Marriott and Peter J. Stuckey. *Programming with Constraints: an Introduction.* MIT Press, 1998.

[NS93] A. Nerode and R. Shore. *Logic for Applications.* Springer, 1993.

[Pou95] D. Pountain. Constraint logic programming. feb 1995.

A Aplicação

A.1 Arquivo de Entrada

```
% arquivo de entrada -- exemplo
% problema reduzido (8 profissionais vs.  10 dias no mes)
dmes 10 // dias do mes
dsemana 7 // dia da semana do primeiro dia do mes:  sabado
feriados 0 // feriados
% numero de profissionais
nenf 2 // numero de enfermeiros
naux 4 // numero de auxiliares
ntec 0 // numero de tecnicos
nate 2 // numero de atendentes
nmaq 0 // numero de maqueiros
% numero minimo de profissionais
minenfU 1 // minimo de enfermeiros em dias uteis
minenfN 1 // minimo de enfermeiros em dias nao uteis
minauxU 2 // minimo de auxiliares em dias uteis
minauxN 2 // minimo de auxiliares em dias nao uteis
mintecU 0 // minimo de tecnicos em dias uteis
mintecN 0 // minimo de tecnicos em dias nao uteis
minateU 1 // minimo de atendentes em dias uteis
minateN 1 // minimo de atendentes em dias nao uteis
minmaqU 0 // minimo de maqueiros em dias uteis
minmaqN 0 // minimo de maqueiros em dias nao uteis
% impedimentos
imp 1 2 3 4 0 // impedimentos profissional 1
imp 0 // impedimentos profissional 2
imp 5 6 7 8 9 0 // impedimentos profissional 3
imp 0 // impedimentos profissional 4
imp 0 // impedimentos profissional 5
imp 0 // impedimentos profissional 6
imp 0 // impedimentos profissional 7
imp 0 // impedimentos profissional 8
% folgas
f 2 // numero de folgas profissional 1
f 2 // numero de folgas profissional 2
f 2 // numero de folgas profissional 3
f 2 // numero de folgas profissional 4
f 2 // numero de folgas profissional 5
f 2 // numero de folgas profissional 6
f 2 // numero de folgas profissional 7
f 2 // numero de folgas profissional 8
% dias de trabalho consecutivos
prev 1 // dias de trabalho consecutivos no mes anterior - profissional 1
prev 2 // dias de trabalho consecutivos no mes anterior - profissional 2
prev 2 // dias de trabalho consecutivos no mes anterior - profissional 3
prev 3 // dias de trabalho consecutivos no mes anterior - profissional 4
prev 4 // dias de trabalho consecutivos no mes anterior - profissional 5
prev 5 // dias de trabalho consecutivos no mes anterior - profissional 6
```

```
prev 4 // dias de trabalho consecutivos no mes anterior - profissional 7
prev 4 // dias de trabalho consecutivos no mes anterior - profissional 8
pref 7 0 // preferencias profissional 1
pref 5 0 // preferencias profissional 2
pref 1 0 // preferencias profissional 3
pref 3 0 // preferencias profissional 4
pref 2 0 // preferencias profissional 5
pref 2 5 0 // preferencias profissional 6
pref 2 7 0 // preferencias profissional 7
pref 3 9 0 // preferencias profissional 8
```

A.2 Programa Filtro

```
import java.io.*;
import java.util.*;
import java.lang.*;

class Filtro {

    static String descricao = ""; //string impressa no inicio do arquivo
    static int dmes; //numero de dias no mes
    static int dsemana; //dia na semana do primeiro dia do mes (1 = domingo, 2 = segunda, etc)
    static boolean[] diasuteis; //se diasuteis[i] = true entao i eh um dia util
    static int nuteis; //numero de diasuteis no mes
    static boolean[] domingos; //se domingos[i] = true entao i eh um domingo
    //numero de profissionais de cada tipo
    static int nenf; //numero de enfermeiros
    static int naux; //numero de auxiliares
    static int ntec; //numero de tecnicos
    static int nate; //numero de atendentes
    static int nmaq; //numero de maqueiros
    static int np; //total de profissionais
    //numero minimo profissionais de cada tipo
    static int minenfU; //numero minimo de enfermeiros em dias uteis
    static int minenfN; //numero minimo de enfermeiros em dias nao uteis
    static int minauxU; //numero minimo de auxiliares em dias uteis
    static int minauxN; //numero minimo de auxiliares em dias nao uteis
    static int mintecU; //numero minimo de tecnicos em dias uteis
    static int mintecN; //numero minimo de tecnicos em dias nao uteis
    static int minateU; //numero minimo de atendentes em dias uteis
    static int minateN; //numero minimo de atendentes em dias nao uteis
    static int minmaqU; //numero minimo de maqueiros em dias uteis
    static int minmaqN; //numero minimo de maqueiros em dias nao uteis
    //numero minimo profissionais de cada tipo -- por dia
    //se minimos[1][j] = k entao eh preciso no minimo k enfermeiras(tipo 1) no dia j
    static int[][] minimos;
    //impedimentos
    //se imp[i][j] = true entao o profissional i esta impedido de trabalhar no dia j
    static boolean[][] imp;
    //numero de impedimentos
    //nimp[i] eh o numero de dias em que o profissional i esta impedido de trabalhar
    static int[] nimp;
```

```
    //numero de folgas
    //folgas[i] eh o numero de folgas do profissional i
    static int[] folgas;
    //numero de dias ja trabalhando
    //prev[i] eh o numero de dias consecutivos em que o profissional i
    //trabalhou no final do mes anterior
    static int[] prev;
    //preferencias
    //se pref[i][j] = true entao o profissional i tem preferencia por folga no dia j
    static boolean[][] pref;

public static void main(String args[]) throws IOException
{
    if (args[0] != null)
    {
        String InputFileName = args [0];
        initializeVariables(InputFileName);
        constructConstraints();
    }
    else System.out.println("Use: java Filtro entrada > saida");
}

public static void initializeVariables(String InputFileName) throws IOException
{
    int prof = 0;   //"contador" de profissionais --> 1 a np
    FileInputStream fis = new FileInputStream(InputFileName);
    DataInputStream dis = new DataInputStream (fis);
    StringTokenizer st;
    String token1;
    String token2;
    String token;
    //le uma linha do arquivo de entrada
    String s = dis.readLine();
    while (s!=null)
    {
        st = new StringTokenizer(s);
        token1 = st.nextToken();
        //dias do mes
        if (token1.equals("dmes"))
        {
            token2 = st.nextToken();
            //conversao de "string" para "inteiro"
            dmes = (new Integer(token2)).intValue();
            nuteis = dmes;
            //criacao de "diasuteis"
            diasuteis = new boolean[dmes+1];
            for (int i = 1; i <= dmes; i++) {diasuteis[i] = true;}
            //criacao de "domingos"
            domingos = new boolean[dmes+1];
        }
        //dia na semana do primeiro dia do mes
        if (token1.equals("dsemana"))
        {
            token2 = st.nextToken();
            //conversao de "string" para "inteiro"
```

```
            dsemana = (new Integer(token2)).intValue();
            //atualizacao de "diasuteis"
            for (int i = 1; i <= dmes; i++)
            {
                if (((i + 12 - dsemana) % 7) == 0) //domingo
                {
                    domingos[i] = true;
                    diasuteis[i] = false;
                    nuteis--;
                }
                if (((i + 6 - dsemana) % 7) == 0) //sabado
                {
                    diasuteis[i] = false;
                    nuteis--;
                }
            }
        }
        //feriados
        if (token1.equals("feriados"))
        {
            token = st.nextToken();
            while (!token.equals("0"))
            {
                diasuteis[(new Integer(token)).intValue()] = false;
                nuteis--;
                token = st.nextToken();
            }
        }
        //numero de enfermeiros
        if (token1.equals("nenf"))
        {
            token2 = st.nextToken();
            //conversao de "string" para "inteiro"
            nenf = (new Integer(token2)).intValue();
        }
        //numero de auxiliares
        if (token1.equals("naux"))
        {
            token2 = st.nextToken();
            //conversao de "string" para "inteiro"
            naux = (new Integer(token2)).intValue();
        }
        //numero de tecnicos
        if (token1.equals("ntec"))
        {
            token2 = st.nextToken();
            //conversao de "string" para "inteiro"
            ntec = (new Integer(token2)).intValue();
        }
        //numero de atendentes
        if (token1.equals("nate"))
        {
            token2 = st.nextToken();
            //conversao de "string" para "inteiro"
            nate = (new Integer(token2)).intValue();
```

```
}
//numero de maqueiros
if (token1.equals("nmaq"))
{
    token2 = st.nextToken();
    //conversao de "string" para "inteiro"
    nmaq = (new Integer(token2)).intValue();
    np = nenf + naux + ntec + nate + nmaq;
    //matriz tipos de profissional vs. dias (minimos)
    minimos = new int[6][dmes+1];
    folgas = new int[np+1];
    imp = new boolean[np+1][dmes+1];
    prev = new int[np+1];
    pref = new boolean[np+1][dmes+1];
    nimp = new int[np+1];
    for (int i = 1; i <= np; i++) {nimp[i] = 0;}    //inicializado com 0
}
//numero minimo de enfermeiros em dias uteis
if (token1.equals("minenfU"))
{
    token2 = st.nextToken();
    //conversao de "string" para "inteiro"
    minenfU = (new Integer(token2)).intValue();
}
//numero minimo de enfermeiros em dias nao uteis
if (token1.equals("minenfN"))
{
    token2 = st.nextToken();
    //conversao de "string" para "inteiro"
    minenfN = (new Integer(token2)).intValue();
}
//numero minimo de auxiliares em dias uteis
if (token1.equals("minauxU"))
{
    token2 = st.nextToken();
    //conversao de "string" para "inteiro"
    minauxU = (new Integer(token2)).intValue();
}
//numero minimo de auxiliares em dias nao uteis
if (token1.equals("minauxN"))
{
    token2 = st.nextToken();
    //conversao de "string" para "inteiro"
    minauxN = (new Integer(token2)).intValue();
}
//numero minimo de tecnicos em dias uteis
if (token1.equals("mintecU"))
{
    token2 = st.nextToken();
    //conversao de "string" para "inteiro"
    mintecU = (new Integer(token2)).intValue();
}
//numero minimo de tecnicos em dias nao uteis
if (token1.equals("mintecN"))
{
```

```
    token2 = st.nextToken();
    //conversao de "string" para "inteiro"
    mintecN = (new Integer(token2)).intValue();
}
//numero minimo de atendentes em dias uteis
if (token1.equals("minateU"))
{
    token2 = st.nextToken();
    //conversao de "string" para "inteiro"
    minateU = (new Integer(token2)).intValue();
}
//numero minimo de atendentes em dias nao uteis
if (token1.equals("minateN"))
{
    token2 = st.nextToken();
    //conversao de "string" para "inteiro"
    minateN = (new Integer(token2)).intValue();
}
//numero minimo de maqueiros em dias uteis
if (token1.equals("minmaqU"))
{
    token2 = st.nextToken();
    //conversao de "string" para "inteiro"
    minmaqU = (new Integer(token2)).intValue();
}
//numero minimo de maqueiros em dias nao uteis
if (token1.equals("minmaqN"))
{
    token2 = st.nextToken();
    //conversao de "string" para "inteiro"
    minmaqN = (new Integer(token2)).intValue();
    //inicializacao da matriz "minimos"
    for (int j = 0; j <= dmes; j++)
    {
        if (diasuteis[j])
        {
            minimos[1][j] = minenfU;
            minimos[2][j] = minauxU;
            minimos[3][j] = mintecU;
            minimos[4][j] = minateU;
            minimos[5][j] = minmaqU;
        }
        else
        {
            minimos[1][j] = minenfN;
            minimos[2][j] = minauxN;
            minimos[3][j] = mintecN;
            minimos[4][j] = minateN;
            minimos[5][j] = minmaqN;
        }
    }
}
//impedimentos
if (token1.equals("imp"))
{
```

```
            prof++;
            //prof eh o numero do profissional com que se esta lidando
            token = st.nextToken();
            while (!token.equals("0"))
            {
                imp[prof][(new Integer(token)).intValue()] = true;
                token = st.nextToken();
                nimp[prof]++;
            }
            if (prof == np) prof = 0;
            //se acabaram os "imp", zera o contador prof
        }
        //numero de folgas
        if (token1.equals("f"))
        {
            prof++;
            //prof eh o numero do profissional com que se esta lidando
            token2 = st.nextToken ();
            folgas[prof] = (new Integer(token2)).intValue();;
            //se acabaram os "f", zera o contador prof
            if (prof == np) prof = 0;
        }
        //numero de dias consecutivos em que o profissional trabalhou no mes anterior
        if (token1.equals("prev"))
        {
            prof++;
            //prof eh o numero do profissional com que se esta lidando
            token2 = st.nextToken ();
            prev[prof] = (new Integer(token2)).intValue();
            if (prof == np) prof = 0;
            //se acabaram os "prev", zera o contador prof
        }
        //preferencias por folgas
        if (token1.equals("pref"))
        {
            prof++;
            //prof eh o numero do profissional com que se esta lidando
            token = st.nextToken();
            while (!token.equals("0"))
            {
                pref[prof][(new Integer(token)).intValue()] = true;
                token = st.nextToken();
            }
            if (prof == np) prof = 0;
            //se acabaram os "pref", zera o contador prof
        }
        //le a proxima linha do arquivo de entrada
        s = dis.readLine ();
        }
}

public static void constructConstraints()
{
    int i;
    int j;
```

```
String temp = "";
System.out.println("% Arquivo gerado automaticamente");
System.out.println("% " + descricao);
System.out.println("");
System.out.println("% utilizacao da biblioteca de dominios finitos");
System.out.println("");
System.out.println(":- use_module(library(fd)).");
System.out.println("");
System.out.println("% definicao de labelff");
System.out.println("");
System.out.println("labelff([]) :- !.");
System.out.println("labelff(L) :- deleteff(Var,L,Resto),");
System.out.println("\tindomain(Var),");
System.out.println("\tlabelff(Resto).");
System.out.println("");
System.out.println("% dominio das variaveis");
System.out.println("");
System.out.println("% ESC");
System.out.println("% 0 = trabalho");
System.out.println("% 1 = folga");
System.out.println("% 2 = impedimento");
System.out.println("");
System.out.println("programa :- ");
for (i = 1; i <= np; i++) //para todos os profissionais
{
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        System.out.println("ESC" + i + "x" + j + "::0..2,");
    }
}
System.out.println("");
System.out.println("% ESCBIN");
System.out.println("% 0 = trabalho");
System.out.println("% 1 = folga ou impedimento");
System.out.println("");
for (i = 1; i <= np; i++) //para todos os profissionais
{
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        System.out.println("ESCBIN" + i + "x" + j + "::0..1,");
    }
}
System.out.println("");
System.out.println("% ESCUTIL");
System.out.println("% 0 = trabalho ou impedimento");
System.out.println("% 1 = folga");
System.out.println("");
for (i = 1; i <= np; i++) //para todos os profissionais
{
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        if (diasuteis[j]) System.out.println("ESCUTIL" + i + "x" + j + "::0..1,");
    }
}
System.out.println("");
```

```
System.out.println("% restricao basica: atendida automaticamente");
System.out.println("");
System.out.println("% preferencia de dias de folga");
System.out.println("");
for (i = 1; i <= np; i++) //para todos os profissionais
{
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        if (pref[i][j]) System.out.println("ESCBIN" + i + "x" + j + " #= 1,");
    }
}
System.out.println("");
System.out.println("% impedimentos");
System.out.println("");
for (i = 1; i <= np; i++) //para todos os profissionais
{
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        if (imp[i][j]) System.out.println("ESC" + i + "x" + j + " #= 2,");
    }
}
System.out.println("");
System.out.println("% equivalencia entre as matrizes ESC e ESCBIN");
System.out.println("");
for (i = 1; i <= np; i++) //para todos os profissionais
{
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        System.out.println("ESC" + i + "x" + j + "inc #= ESC" + i + "x" + j +
        " + 1, element(ESC" + i + "x" + j + "inc,[0,1,1],ESCBIN" + i + "x" + j + "),");
    }
}
System.out.println("");
System.out.println("% equivalencia entre as matrizes ESC e ESCUTIL");
System.out.println("");
for (i = 1; i <= np; i++) //para todos os profissionais
{
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        if (diasuteis[j]) System.out.println("element(ESC" + i + "x" + j +
        "inc,[0,1,0],ESCUTIL" + i + "x" + j + "),");

    }
}
System.out.println("");
System.out.println("% folga no domingo");
System.out.println("");
for (i = 1; i <= np; i++) //para todos os profissionais
{
    temp = "";
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        if (domingos[j]) temp = temp.concat("ESCBIN" + i + "x" + j + " + ");
    }
    temp = temp.concat("0 #> 0,");
```

```
    System.out.println(temp);
}
System.out.println("");
System.out.println("% folga a cada 7 dias");
System.out.println("");
int jdec1;
int jdec2;
int jdec3;
int jdec4;
int jdec5;
int jdec6;
for (i = 1; i <= np; i++) //para todos os profissionais
{
    //do primeiro dia ate 7 - Prev deve ter pelo menos uma folga
    int maximo = 7 - prev[i];
    temp = "atmost(" + maximo + ",[";
    for (int k = 1; k <= maximo; k++)
    {
        temp = temp.concat("ESCBIN" + i + "x" + k + ",");
    }
    System.out.println(temp + "0],0),");
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        if (j >= 7)
        {
            //do atual ate 6 anteriores a ele deve ter pelo menos uma folga
            jdec1 = j - 1;
            jdec2 = j - 2;
            jdec3 = j - 3;
            jdec4 = j - 4;
            jdec5 = j - 5;
            jdec6 = j - 6;
            System.out.println("atmost(6,[ESCBIN" + i + "x" + jdec6 + ",ESCBIN"
            + i + "x" + jdec5 + ",ESCBIN" + i + "x" + jdec4 + ",ESCBIN" + i +
            "x" + jdec3 + ",ESCBIN" + i + "x" + jdec2 + ",ESCBIN" + i + "x" +
            jdec1 + ",ESCBIN" + i + "x" + j + "],0),");
        }
    }
}
System.out.println("");
System.out.println("% no. de folgas e impedimentos");
System.out.println("");
int prof = 0;   //"contador" de profissionais
for (i = 1; i <= nenf; i++) //para todos os enfermeiros
{
    prof++; //"numero" do profissional
    System.out.println("%Enf" + prof + "F" + "::" + folgas[i] + ".." + folgas[i] + ",");
    System.out.println("%Enf" + prof + "I" + "::" + nimp[i] + ".." + nimp[i] + ",");
    temp = "ESC" + i + "x1";
    for (j = 2; j <= dmes; j++) //para todos os dias do mes a partir do segundo
    {
        temp = temp.concat(" + ESC" + i + "x" + j);
    }
    temp = temp.concat(" #= " + folgas[i] + " + 2*" + nimp[i] + ",");
    System.out.println(temp);
```

```
    temp = "ESCBIN" + i + "x1";
    for (j = 2; j <= dmes; j++) //para todos os dias do mes a partir do segundo
    {
        temp = temp.concat(" + ESCBIN" + i + "x" + j);
    }
    temp = temp.concat(" #= " + folgas[i] + " + " + nimp[i] + ",");
    System.out.println(temp);
}
prof = 0;
for (i = nenf + 1; i <= nenf + naux; i++) //para todos os auxiliares
{
    prof++; //"numero" do profissional
    System.out.println("%Aux" + prof + "F" + "::" + folgas[i] + ".." + folgas[i] + ",");
    System.out.println("%Aux" + prof + "I" + "::" + nimp[i] + ".." + nimp[i] + ",");
    temp = "ESC" + i + "x1";
    for (j = 2; j <= dmes; j++) //para todos os dias do mes a partir do segundo
    {
        temp = temp.concat(" + ESC" + i + "x" + j);
    }
    temp = temp.concat(" #= " + folgas[i] + " + 2*" + nimp[i] + ",");
    System.out.println(temp);
    temp = "ESCBIN" + i + "x1";
    for (j = 2; j <= dmes; j++) //para todos os dias do mes a partir do segundo
    {
        temp = temp.concat(" + ESCBIN" + i + "x" + j);
    }
    temp = temp.concat(" #= " + folgas[i] + " + " + nimp[i] + ",");
    System.out.println(temp + "");
}
prof = 0;
for (i = nenf + naux + 1; i <= nenf + naux + ntec; i++) //para todos os tecnicos
{
    prof++; //"numero" do profissional
    System.out.println("%Tec" + prof + "F" + "::" + folgas[i] + ".." + folgas[i] + ",");
    System.out.println("%Tec" + prof + "I" + "::" + nimp[i] + ".." + nimp[i] + ",");
    temp = "ESC" + i + "x1";
    for (j = 2; j <= dmes; j++) //para todos os dias do mes a partir do segundo
    {
        temp = temp.concat(" + ESC" + i + "x" + j);
    }
    temp = temp.concat(" #= " + folgas[i] + " + 2*" + nimp[i] + ",");
    System.out.println(temp);
    temp = "ESCBIN" + i + "x1";
    for (j = 2; j <= dmes; j++) //para todos os dias do mes a partir do segundo
    {
        temp = temp.concat(" + ESCBIN" + i + "x" + j);
    }
    temp = temp.concat(" #= " + folgas[i] + " + " + nimp[i] + ",");
    System.out.println(temp + "");
}
prof = 0;
for (i = nenf + naux + ntec + 1; i <= nenf + naux + ntec + nate; i++)
//para todos os atendentes
{
    prof++; //"numero" do profissional
```

```
    System.out.println("%Ate" + prof + "F" + "::" + folgas[i] + ".." + folgas[i] + ",");
    System.out.println("%Ate" + prof + "I" + "::" + nimp[i] + ".." + nimp[i] + ",");
    temp = "ESC" + i + "x1";
    for (j = 2; j <= dmes; j++) //para todos os dias do mes a partir do segundo
    {
        temp = temp.concat(" + ESC" + i + "x" + j);
    }
    temp = temp.concat(" #= " + folgas[i] + " + 2*" + nimp[i] + ",");
    System.out.println(temp);
    temp = "ESCBIN" + i + "x1";
    for (j = 2; j <= dmes; j++) //para todos os dias do mes a partir do segundo
    {
        temp = temp.concat(" + ESCBIN" + i + "x" + j);
    }
    temp = temp.concat(" #= " + folgas[i] + " + " + nimp[i] + ",");
    System.out.println(temp + "");
}
prof = 0;
for (i = nenf + naux + ntec + nate + 1; i <= nenf + naux + ntec + nate + nmaq; i++)
//para todos os maqueiros
{
    prof++; //"numero" do profissional
    System.out.println("%Maq" + prof + "F" + "::" + folgas[i] + ".." + folgas[i] + ",");
    System.out.println("%Maq" + prof + "I" + "::" + nimp[i] + ".." + nimp[i] + ",");
    temp = "ESC" + i + "x1";
    for (j = 2; j <= dmes; j++) //para todos os dias do mes a partir do segundo
    {
        temp = temp.concat(" + ESC" + i + "x" + j);
    }
    temp = temp.concat(" #= " + folgas[i] + " + 2*" + nimp[i] + ",");
    System.out.println(temp);
    temp = "ESCBIN" + i + "x1";
    for (j = 2; j <= dmes; j++) //para todos os dias do mes a partir do segundo
    {
        temp = temp.concat(" + ESCBIN" + i + "x" + j);
    }
    temp = temp.concat(" #= " + folgas[i] + " + " + nimp[i] + ",");
    System.out.println(temp + "");
}
prof = 0;
System.out.println("");
System.out.println("% no. minimo de profissionais");
System.out.println("");
int minenf;
for (j = 1; j <= dmes; j++) //para todos os dias do mes
{
    if (diasuteis[j])
    {
        minenf = minenfU;
        System.out.println("%MinEnf" + j + "::" + minenfU + ".." + minenfU + ", ");
    }
    else
    {
        minenf = minenfN;
        System.out.println("%MinEnf" + j + "::" + minenfN + ".." + minenfN + ", ");
```

```
        }
        temp = "" + nenf;
        for (i = 1; i <= nenf; i++) //para todos os enfermeiros
        {
            temp = temp.concat(" - ESCBIN" + i + "x" + j);
        }
        temp = temp.concat(" #>= " + minenf + ",");
        System.out.println(temp);
    }
    System.out.println("");
    int minaux;
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        if (diasuteis[j])
        {
            minaux = minauxU;
            System.out.println("%MinAux" + j + "::" + minauxU + ".." + minauxU + ", ");
        }
        else
        {
            minaux = minauxN;
            System.out.println("%MinAux" + j + "::" + minauxN + ".." + minauxN + ", ");
        }
        temp = "" + naux;
        for (i = nenf + 1; i <= nenf + naux; i++) //para todos os auxiliares
        {
            temp = temp.concat(" - ESCBIN" + i + "x" + j);
        }
        temp = temp.concat(" #>= " + minaux + ",");
        System.out.println(temp);
    }
    System.out.println("");
    int mintec;
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        if (diasuteis[j])
        {
            mintec = mintecU;
            System.out.println("%MinTec" + j + "::" + mintecU + ".." + mintecU + ", ");
        }
        else
        {
            mintec = mintecN;
            System.out.println("%MinTec" + j + "::" + mintecN + ".." + mintecN + ", ");
        }
        temp = "" + ntec;
        for (i = nenf + naux + 1; i <= nenf + naux + ntec; i++)
        //para todos os tecnicos
        {
            temp = temp.concat(" - ESCBIN" + i + "x" + j);
        }
        temp = temp.concat(" #>= " + mintec + ",");
        System.out.println(temp);
    }
    System.out.println("");
```

```
int minate;
for (j = 1; j <= dmes; j++) //para todos os dias do mes
{
    if (diasuteis[j])
    {
        minate = minateU;
        System.out.println("%MinAte" + j + "::" + minateU + ".." + minateU + ", ");
    }
    else
    {
        minate = minateN;
        System.out.println("%MinAte" + j + "::" + minateN + ".." + minateN + ", ");
    }
    temp = "" + nate;
    for (i = nenf + naux + ntec + 1; i <= nenf + naux + ntec + nate; i++)
    //para todos os atendentes
    {
        temp = temp.concat(" - ESCBIN" + i + "x" + j);
    }
    temp = temp.concat(" #>= " + minate + ",");
    System.out.println(temp);
}
System.out.println("");
int minmaq;
for (j = 1; j <= dmes; j++) //para todos os dias do mes
{
    if (diasuteis[j])
    {
        minmaq = minmaqU;
        System.out.println("%MinMaq" + j + "::" + minmaqU + ".." + minmaqU + ", ");
    }
    else
    {
        minmaq = minmaqN;
        System.out.println("%MinMaq" + j + "::" + minmaqN + ".." + minmaqN + ", ");
    }
    temp = "" + minmaq;
    for (i = nenf + naux + ntec + nate + 1; i <= nenf + naux + ntec + nate + nmaq; i++)
    //para todos os maqueiros
    {
        temp = temp.concat(" - ESCBIN" + i + "x" + j);
    }
    temp = temp.concat(" #>= " + minmaq + ",");
    System.out.println(temp);
}
System.out.println("");
System.out.println("% reuniao");
System.out.println("");
System.out.println("% FolgasN conta o numero de pessoas em folga no dia N");
System.out.println("% dominio: 0 ate numero de profissionais;
for (j = 1; j <= dmes; j++) //para todos os dias do mes
{
    if (diasuteis[j]) System.out.println("Folgas" + j + "::0.." + np + ",");
}
System.out.println("");
```

```
for (j = 1; j <= dmes; j++) //para todos os dias do mes
{
    if (diasuteis[j])
    {
        temp = "";
        for (i = 1; i <= np; i++) //para todos os profissionais
        {
            temp = temp.concat("ESCUTIL" + i + "x" + j + " + ");
        }
        temp = temp.concat("0 #= Folgas" + j + ",");
        System.out.println(temp);
    }
}
System.out.println("");
System.out.println("% ExisteFolgaN indica se algum profissional esta de folga no dia N
(0 = ninguem de folga; 1 = pelo menos um de folga)");
for (j = 1; j <= dmes; j++) //para todos os dias do mes
{
    if (diasuteis[j]) System.out.println("ExisteFolga" + j + "::0..1,");
}
System.out.println("");
for (j = 1; j <= dmes; j++) //para todos os dias do mes
{
    if (diasuteis[j])
    {
        temp = "Folgas" + j + "inc #= Folgas" + j + " + 1, element(Folgas" + j + "inc,[0,";
        for (i = 1; i < np; i++) //para todos os profissionais, menos 1
        {
            temp = temp.concat("1,");
        }
        temp = temp.concat("1],ExisteFolga" + j + "),");
        System.out.println(temp);
    }
}
System.out.println("");
System.out.println("% soma de todos ExisteFolga deve ser menor que o numero de dias uteis
(pelo menos um dia sem folga nenhuma)");
temp = "";
for (j = 1; j <= dmes; j++) //para todos os dias do mes
{
    if (diasuteis[j]) temp = temp.concat("ExisteFolga" + j + " + ");
}
temp = temp.concat("0 #< " + nuteis + ",");
System.out.println(temp);
System.out.println("");
System.out.println("% obtencao do resultado");
System.out.println("");
System.out.print("labelff([");
for (i = 1; i <= np; i++) //para todos os profissionais
{
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        System.out.print("ESC" + i + "x" + j + ", ");
    }
}
```

```
System.out.println("ESCBIN1x1]),");
System.out.println("");
prof = 0;
for (i = 1; i <= nenf; i++) //para todos os enfermeiros
{
    prof++;
    temp = "write(\"Enf" + prof + ": \"),";
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        temp = temp.concat("write(ESC" + i + "x" + j + "), ");
    }
    temp = temp.concat("write(\"\\n\"),");
    System.out.println(temp);
}
System.out.println("");
prof = 0;
for (i = nenf + 1; i <= nenf + naux; i++) //para todos os auxiliares
{
    prof++;
    temp = "write(\"Aux" + prof + ": \"),";
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        temp = temp.concat("write(ESC" + i + "x" + j + "), ");
    }
    temp = temp.concat("write(\"\\n\"),");
    System.out.println(temp);
}
System.out.println("");
prof = 0;
for (i = nenf + naux + 1; i <= nenf + naux + ntec; i++) //para todos os tecnicos
{
    prof++;
    temp = "write(\"Tec" + prof + ": \"),";
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        temp = temp.concat("write(ESC" + i + "x" + j + "), ");
    }
    temp = temp.concat("write(\"\\n\"),");
    System.out.println(temp);
}
System.out.println("");
prof = 0;
for (i = nenf + naux + ntec + 1; i <= nenf + naux + ntec + nate; i++)
//para todos os atendentes
{
    prof++;
    temp = "write(\"Ate" + prof + ": \"),";
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        temp = temp.concat("write(ESC" + i + "x" + j + "), ");
    }
    temp = temp.concat("write(\"\\n\"),");
    System.out.println(temp);
}
System.out.println("");
```

```
    prof = 0;
    for (i = nenf + naux + ntec + nate + 1; i <= nenf + naux + ntec + nate + nmaq; i++)
    //para todos os maqueiros
    {
        prof++;
        temp = "write(\"Maq" + prof + ": \"),";
        for (j = 1; j <= dmes; j++) //para todos os dias do mes
        {
            temp = temp.concat("write(ESC" + i + "x" + j + "), ");
        }
        temp = temp.concat("write(\"\\n\"),");
        System.out.println(temp);
    }
    System.out.println("");
    System.out.println("write(\"\\n\"),");
    System.out.println("");
    for (j = 1; j <= dmes; j++) //para todos os dias do mes
    {
        if (diasuteis[j])
        {
            System.out.println("write(\"ExisteFolga" + j + ": \"),
            write(ExisteFolga" + j + "), write(\"\\n\")," );
        }
    }
    System.out.println("write(\"\\n\")." );
}

}
```

A.3 Arquivo de Restrições

```
% Arquivo gerado automaticamente
%

% utilizacao da biblioteca de dominios finitos

:- use_module(library(fd)).

% definicao de labelff

labelff([]) :- !.
labelff(L) :- deleteff(Var,L,Resto),
indomain(Var),
labelff(Resto).

% dominio das variaveis

% ESC
% 0 = trabalho
% 1 = folga
% 2 = impedimento

programa :-
ESC1x1::0..2,
```

```
ESC1x2::0..2,
ESC1x3::0..2,
ESC1x4::0..2,
ESC1x5::0..2,
ESC1x6::0..2,
ESC1x7::0..2,
ESC1x8::0..2,
ESC1x9::0..2,
ESC1x10::0..2,
ESC2x1::0..2,
ESC2x2::0..2,
ESC2x3::0..2,
ESC2x4::0..2,
ESC2x5::0..2,
ESC2x6::0..2,
ESC2x7::0..2,
ESC2x8::0..2,
ESC2x9::0..2,
ESC2x10::0..2,
ESC3x1::0..2,
ESC3x2::0..2,
ESC3x3::0..2,
ESC3x4::0..2,
ESC3x5::0..2,
ESC3x6::0..2,
ESC3x7::0..2,
ESC3x8::0..2,
ESC3x9::0..2,
ESC3x10::0..2,
ESC4x1::0..2,
ESC4x2::0..2,
ESC4x3::0..2,
ESC4x4::0..2,
ESC4x5::0..2,
ESC4x6::0..2,
ESC4x7::0..2,
ESC4x8::0..2,
ESC4x9::0..2,
ESC4x10::0..2,
ESC5x1::0..2,
ESC5x2::0..2,
ESC5x3::0..2,
ESC5x4::0..2,
ESC5x5::0..2,
ESC5x6::0..2,
ESC5x7::0..2,
ESC5x8::0..2,
ESC5x9::0..2,
ESC5x10::0..2,
ESC6x1::0..2,
ESC6x2::0..2,
ESC6x3::0..2,
ESC6x4::0..2,
ESC6x5::0..2,
ESC6x6::0..2,
```

```
ESC6x7::0..2,
ESC6x8::0..2,
ESC6x9::0..2,
ESC6x10::0..2,
ESC7x1::0..2,
ESC7x2::0..2,
ESC7x3::0..2,
ESC7x4::0..2,
ESC7x5::0..2,
ESC7x6::0..2,
ESC7x7::0..2,
ESC7x8::0..2,
ESC7x9::0..2,
ESC7x10::0..2,
ESC8x1::0..2,
ESC8x2::0..2,
ESC8x3::0..2,
ESC8x4::0..2,
ESC8x5::0..2,
ESC8x6::0..2,
ESC8x7::0..2,
ESC8x8::0..2,
ESC8x9::0..2,
ESC8x10::0..2,

% ESCBIN
% 0 = trabalho
% 1 = folga ou impedimento

ESCBIN1x1::0..1,
ESCBIN1x2::0..1,
ESCBIN1x3::0..1,
ESCBIN1x4::0..1,
ESCBIN1x5::0..1,
ESCBIN1x6::0..1,
ESCBIN1x7::0..1,
ESCBIN1x8::0..1,
ESCBIN1x9::0..1,
ESCBIN1x10::0..1,
ESCBIN2x1::0..1,
ESCBIN2x2::0..1,
ESCBIN2x3::0..1,
ESCBIN2x4::0..1,
ESCBIN2x5::0..1,
ESCBIN2x6::0..1,
ESCBIN2x7::0..1,
ESCBIN2x8::0..1,
ESCBIN2x9::0..1,
ESCBIN2x10::0..1,
ESCBIN3x1::0..1,
ESCBIN3x2::0..1,
ESCBIN3x3::0..1,
ESCBIN3x4::0..1,
ESCBIN3x5::0..1,
ESCBIN3x6::0..1,
```

```
ESCBIN3x7::0..1,
ESCBIN3x8::0..1,
ESCBIN3x9::0..1,
ESCBIN3x10::0..1,
ESCBIN4x1::0..1,
ESCBIN4x2::0..1,
ESCBIN4x3::0..1,
ESCBIN4x4::0..1,
ESCBIN4x5::0..1,
ESCBIN4x6::0..1,
ESCBIN4x7::0..1,
ESCBIN4x8::0..1,
ESCBIN4x9::0..1,
ESCBIN4x10::0..1,
ESCBIN5x1::0..1,
ESCBIN5x2::0..1,
ESCBIN5x3::0..1,
ESCBIN5x4::0..1,
ESCBIN5x5::0..1,
ESCBIN5x6::0..1,
ESCBIN5x7::0..1,
ESCBIN5x8::0..1,
ESCBIN5x9::0..1,
ESCBIN5x10::0..1,
ESCBIN6x1::0..1,
ESCBIN6x2::0..1,
ESCBIN6x3::0..1,
ESCBIN6x4::0..1,
ESCBIN6x5::0..1,
ESCBIN6x6::0..1,
ESCBIN6x7::0..1,
ESCBIN6x8::0..1,
ESCBIN6x9::0..1,
ESCBIN6x10::0..1,
ESCBIN7x1::0..1,
ESCBIN7x2::0..1,
ESCBIN7x3::0..1,
ESCBIN7x4::0..1,
ESCBIN7x5::0..1,
ESCBIN7x6::0..1,
ESCBIN7x7::0..1,
ESCBIN7x8::0..1,
ESCBIN7x9::0..1,
ESCBIN7x10::0..1,
ESCBIN8x1::0..1,
ESCBIN8x2::0..1,
ESCBIN8x3::0..1,
ESCBIN8x4::0..1,
ESCBIN8x5::0..1,
ESCBIN8x6::0..1,
ESCBIN8x7::0..1,
ESCBIN8x8::0..1,
ESCBIN8x9::0..1,
ESCBIN8x10::0..1,
```

```
% ESCUTIL
% 0 = trabalho ou impedimento
% 1 = folga

ESCUTIL1x3::0..1,
ESCUTIL1x4::0..1,
ESCUTIL1x5::0..1,
ESCUTIL1x6::0..1,
ESCUTIL1x7::0..1,
ESCUTIL1x10::0..1,
ESCUTIL2x3::0..1,
ESCUTIL2x4::0..1,
ESCUTIL2x5::0..1,
ESCUTIL2x6::0..1,
ESCUTIL2x7::0..1,
ESCUTIL2x10::0..1,
ESCUTIL3x3::0..1,
ESCUTIL3x4::0..1,
ESCUTIL3x5::0..1,
ESCUTIL3x6::0..1,
ESCUTIL3x7::0..1,
ESCUTIL3x10::0..1,
ESCUTIL4x3::0..1,
ESCUTIL4x4::0..1,
ESCUTIL4x5::0..1,
ESCUTIL4x6::0..1,
ESCUTIL4x7::0..1,
ESCUTIL4x10::0..1,
ESCUTIL5x3::0..1,
ESCUTIL5x4::0..1,
ESCUTIL5x5::0..1,
ESCUTIL5x6::0..1,
ESCUTIL5x7::0..1,
ESCUTIL5x10::0..1,
ESCUTIL6x3::0..1,
ESCUTIL6x4::0..1,
ESCUTIL6x5::0..1,
ESCUTIL6x6::0..1,
ESCUTIL6x7::0..1,
ESCUTIL6x10::0..1,
ESCUTIL7x3::0..1,
ESCUTIL7x4::0..1,
ESCUTIL7x5::0..1,
ESCUTIL7x6::0..1,
ESCUTIL7x7::0..1,
ESCUTIL7x10::0..1,
ESCUTIL8x3::0..1,
ESCUTIL8x4::0..1,
ESCUTIL8x5::0..1,
ESCUTIL8x6::0..1,
ESCUTIL8x7::0..1,
ESCUTIL8x10::0..1,

% restricao basica: atendida automaticamente
```

```
% preferencia de dias de folga

ESCBIN1x7 #= 1,
ESCBIN2x5 #= 1,
ESCBIN3x1 #= 1,
ESCBIN4x3 #= 1,
ESCBIN5x2 #= 1,
ESCBIN6x2 #= 1,
ESCBIN6x5 #= 1,
ESCBIN7x2 #= 1,
ESCBIN7x7 #= 1,
ESCBIN8x3 #= 1,
ESCBIN8x9 #= 1,

% impedimentos

ESC1x1 #= 2,
ESC1x2 #= 2,
ESC1x3 #= 2,
ESC1x4 #= 2,
ESC3x5 #= 2,
ESC3x6 #= 2,
ESC3x7 #= 2,
ESC3x8 #= 2,
ESC3x9 #= 2,

% equivalencia entre as matrizes ESC e ESCBIN

ESC1x1inc #= ESC1x1 + 1, element(ESC1x1inc,[0,1,1],ESCBIN1x1),
ESC1x2inc #= ESC1x2 + 1, element(ESC1x2inc,[0,1,1],ESCBIN1x2),
ESC1x3inc #= ESC1x3 + 1, element(ESC1x3inc,[0,1,1],ESCBIN1x3),
ESC1x4inc #= ESC1x4 + 1, element(ESC1x4inc,[0,1,1],ESCBIN1x4),
ESC1x5inc #= ESC1x5 + 1, element(ESC1x5inc,[0,1,1],ESCBIN1x5),
ESC1x6inc #= ESC1x6 + 1, element(ESC1x6inc,[0,1,1],ESCBIN1x6),
ESC1x7inc #= ESC1x7 + 1, element(ESC1x7inc,[0,1,1],ESCBIN1x7),
ESC1x8inc #= ESC1x8 + 1, element(ESC1x8inc,[0,1,1],ESCBIN1x8),
ESC1x9inc #= ESC1x9 + 1, element(ESC1x9inc,[0,1,1],ESCBIN1x9),
ESC1x10inc #= ESC1x10 + 1, element(ESC1x10inc,[0,1,1],ESCBIN1x10),
ESC2x1inc #= ESC2x1 + 1, element(ESC2x1inc,[0,1,1],ESCBIN2x1),
ESC2x2inc #= ESC2x2 + 1, element(ESC2x2inc,[0,1,1],ESCBIN2x2),
ESC2x3inc #= ESC2x3 + 1, element(ESC2x3inc,[0,1,1],ESCBIN2x3),
ESC2x4inc #= ESC2x4 + 1, element(ESC2x4inc,[0,1,1],ESCBIN2x4),
ESC2x5inc #= ESC2x5 + 1, element(ESC2x5inc,[0,1,1],ESCBIN2x5),
ESC2x6inc #= ESC2x6 + 1, element(ESC2x6inc,[0,1,1],ESCBIN2x6),
ESC2x7inc #= ESC2x7 + 1, element(ESC2x7inc,[0,1,1],ESCBIN2x7),
ESC2x8inc #= ESC2x8 + 1, element(ESC2x8inc,[0,1,1],ESCBIN2x8),
ESC2x9inc #= ESC2x9 + 1, element(ESC2x9inc,[0,1,1],ESCBIN2x9),
ESC2x10inc #= ESC2x10 + 1, element(ESC2x10inc,[0,1,1],ESCBIN2x10),
ESC3x1inc #= ESC3x1 + 1, element(ESC3x1inc,[0,1,1],ESCBIN3x1),
ESC3x2inc #= ESC3x2 + 1, element(ESC3x2inc,[0,1,1],ESCBIN3x2),
ESC3x3inc #= ESC3x3 + 1, element(ESC3x3inc,[0,1,1],ESCBIN3x3),
ESC3x4inc #= ESC3x4 + 1, element(ESC3x4inc,[0,1,1],ESCBIN3x4),
ESC3x5inc #= ESC3x5 + 1, element(ESC3x5inc,[0,1,1],ESCBIN3x5),
ESC3x6inc #= ESC3x6 + 1, element(ESC3x6inc,[0,1,1],ESCBIN3x6),
ESC3x7inc #= ESC3x7 + 1, element(ESC3x7inc,[0,1,1],ESCBIN3x7),
```

```
ESC3x8inc #= ESC3x8 + 1, element(ESC3x8inc,[0,1,1],ESCBIN3x8),
ESC3x9inc #= ESC3x9 + 1, element(ESC3x9inc,[0,1,1],ESCBIN3x9),
ESC3x10inc #= ESC3x10 + 1, element(ESC3x10inc,[0,1,1],ESCBIN3x10),
ESC4x1inc #= ESC4x1 + 1, element(ESC4x1inc,[0,1,1],ESCBIN4x1),
ESC4x2inc #= ESC4x2 + 1, element(ESC4x2inc,[0,1,1],ESCBIN4x2),
ESC4x3inc #= ESC4x3 + 1, element(ESC4x3inc,[0,1,1],ESCBIN4x3),
ESC4x4inc #= ESC4x4 + 1, element(ESC4x4inc,[0,1,1],ESCBIN4x4),
ESC4x5inc #= ESC4x5 + 1, element(ESC4x5inc,[0,1,1],ESCBIN4x5),
ESC4x6inc #= ESC4x6 + 1, element(ESC4x6inc,[0,1,1],ESCBIN4x6),
ESC4x7inc #= ESC4x7 + 1, element(ESC4x7inc,[0,1,1],ESCBIN4x7),
ESC4x8inc #= ESC4x8 + 1, element(ESC4x8inc,[0,1,1],ESCBIN4x8),
ESC4x9inc #= ESC4x9 + 1, element(ESC4x9inc,[0,1,1],ESCBIN4x9),
ESC4x10inc #= ESC4x10 + 1, element(ESC4x10inc,[0,1,1],ESCBIN4x10),
ESC5x1inc #= ESC5x1 + 1, element(ESC5x1inc,[0,1,1],ESCBIN5x1),
ESC5x2inc #= ESC5x2 + 1, element(ESC5x2inc,[0,1,1],ESCBIN5x2),
ESC5x3inc #= ESC5x3 + 1, element(ESC5x3inc,[0,1,1],ESCBIN5x3),
ESC5x4inc #= ESC5x4 + 1, element(ESC5x4inc,[0,1,1],ESCBIN5x4),
ESC5x5inc #= ESC5x5 + 1, element(ESC5x5inc,[0,1,1],ESCBIN5x5),
ESC5x6inc #= ESC5x6 + 1, element(ESC5x6inc,[0,1,1],ESCBIN5x6),
ESC5x7inc #= ESC5x7 + 1, element(ESC5x7inc,[0,1,1],ESCBIN5x7),
ESC5x8inc #= ESC5x8 + 1, element(ESC5x8inc,[0,1,1],ESCBIN5x8),
ESC5x9inc #= ESC5x9 + 1, element(ESC5x9inc,[0,1,1],ESCBIN5x9),
ESC5x10inc #= ESC5x10 + 1, element(ESC5x10inc,[0,1,1],ESCBIN5x10),
ESC6x1inc #= ESC6x1 + 1, element(ESC6x1inc,[0,1,1],ESCBIN6x1),
ESC6x2inc #= ESC6x2 + 1, element(ESC6x2inc,[0,1,1],ESCBIN6x2),
ESC6x3inc #= ESC6x3 + 1, element(ESC6x3inc,[0,1,1],ESCBIN6x3),
ESC6x4inc #= ESC6x4 + 1, element(ESC6x4inc,[0,1,1],ESCBIN6x4),
ESC6x5inc #= ESC6x5 + 1, element(ESC6x5inc,[0,1,1],ESCBIN6x5),
ESC6x6inc #= ESC6x6 + 1, element(ESC6x6inc,[0,1,1],ESCBIN6x6),
ESC6x7inc #= ESC6x7 + 1, element(ESC6x7inc,[0,1,1],ESCBIN6x7),
ESC6x8inc #= ESC6x8 + 1, element(ESC6x8inc,[0,1,1],ESCBIN6x8),
ESC6x9inc #= ESC6x9 + 1, element(ESC6x9inc,[0,1,1],ESCBIN6x9),
ESC6x10inc #= ESC6x10 + 1, element(ESC6x10inc,[0,1,1],ESCBIN6x10),
ESC7x1inc #= ESC7x1 + 1, element(ESC7x1inc,[0,1,1],ESCBIN7x1),
ESC7x2inc #= ESC7x2 + 1, element(ESC7x2inc,[0,1,1],ESCBIN7x2),
ESC7x3inc #= ESC7x3 + 1, element(ESC7x3inc,[0,1,1],ESCBIN7x3),
ESC7x4inc #= ESC7x4 + 1, element(ESC7x4inc,[0,1,1],ESCBIN7x4),
ESC7x5inc #= ESC7x5 + 1, element(ESC7x5inc,[0,1,1],ESCBIN7x5),
ESC7x6inc #= ESC7x6 + 1, element(ESC7x6inc,[0,1,1],ESCBIN7x6),
ESC7x7inc #= ESC7x7 + 1, element(ESC7x7inc,[0,1,1],ESCBIN7x7),
ESC7x8inc #= ESC7x8 + 1, element(ESC7x8inc,[0,1,1],ESCBIN7x8),
ESC7x9inc #= ESC7x9 + 1, element(ESC7x9inc,[0,1,1],ESCBIN7x9),
ESC7x10inc #= ESC7x10 + 1, element(ESC7x10inc,[0,1,1],ESCBIN7x10),
ESC8x1inc #= ESC8x1 + 1, element(ESC8x1inc,[0,1,1],ESCBIN8x1),
ESC8x2inc #= ESC8x2 + 1, element(ESC8x2inc,[0,1,1],ESCBIN8x2),
ESC8x3inc #= ESC8x3 + 1, element(ESC8x3inc,[0,1,1],ESCBIN8x3),
ESC8x4inc #= ESC8x4 + 1, element(ESC8x4inc,[0,1,1],ESCBIN8x4),
ESC8x5inc #= ESC8x5 + 1, element(ESC8x5inc,[0,1,1],ESCBIN8x5),
ESC8x6inc #= ESC8x6 + 1, element(ESC8x6inc,[0,1,1],ESCBIN8x6),
ESC8x7inc #= ESC8x7 + 1, element(ESC8x7inc,[0,1,1],ESCBIN8x7),
ESC8x8inc #= ESC8x8 + 1, element(ESC8x8inc,[0,1,1],ESCBIN8x8),
ESC8x9inc #= ESC8x9 + 1, element(ESC8x9inc,[0,1,1],ESCBIN8x9),
ESC8x10inc #= ESC8x10 + 1, element(ESC8x10inc,[0,1,1],ESCBIN8x10),

% equivalencia entre as matrizes ESC e ESCUTIL
```

```
element(ESC1x3inc,[0,1,0],ESCUTIL1x3),
element(ESC1x4inc,[0,1,0],ESCUTIL1x4),
element(ESC1x5inc,[0,1,0],ESCUTIL1x5),
element(ESC1x6inc,[0,1,0],ESCUTIL1x6),
element(ESC1x7inc,[0,1,0],ESCUTIL1x7),
element(ESC1x10inc,[0,1,0],ESCUTIL1x10),
element(ESC2x3inc,[0,1,0],ESCUTIL2x3),
element(ESC2x4inc,[0,1,0],ESCUTIL2x4),
element(ESC2x5inc,[0,1,0],ESCUTIL2x5),
element(ESC2x6inc,[0,1,0],ESCUTIL2x6),
element(ESC2x7inc,[0,1,0],ESCUTIL2x7),
element(ESC2x10inc,[0,1,0],ESCUTIL2x10),
element(ESC3x3inc,[0,1,0],ESCUTIL3x3),
element(ESC3x4inc,[0,1,0],ESCUTIL3x4),
element(ESC3x5inc,[0,1,0],ESCUTIL3x5),
element(ESC3x6inc,[0,1,0],ESCUTIL3x6),
element(ESC3x7inc,[0,1,0],ESCUTIL3x7),
element(ESC3x10inc,[0,1,0],ESCUTIL3x10),
element(ESC4x3inc,[0,1,0],ESCUTIL4x3),
element(ESC4x4inc,[0,1,0],ESCUTIL4x4),
element(ESC4x5inc,[0,1,0],ESCUTIL4x5),
element(ESC4x6inc,[0,1,0],ESCUTIL4x6),
element(ESC4x7inc,[0,1,0],ESCUTIL4x7),
element(ESC4x10inc,[0,1,0],ESCUTIL4x10),
element(ESC5x3inc,[0,1,0],ESCUTIL5x3),
element(ESC5x4inc,[0,1,0],ESCUTIL5x4),
element(ESC5x5inc,[0,1,0],ESCUTIL5x5),
element(ESC5x6inc,[0,1,0],ESCUTIL5x6),
element(ESC5x7inc,[0,1,0],ESCUTIL5x7),
element(ESC5x10inc,[0,1,0],ESCUTIL5x10),
element(ESC6x3inc,[0,1,0],ESCUTIL6x3),
element(ESC6x4inc,[0,1,0],ESCUTIL6x4),
element(ESC6x5inc,[0,1,0],ESCUTIL6x5),
element(ESC6x6inc,[0,1,0],ESCUTIL6x6),
element(ESC6x7inc,[0,1,0],ESCUTIL6x7),
element(ESC6x10inc,[0,1,0],ESCUTIL6x10),
element(ESC7x3inc,[0,1,0],ESCUTIL7x3),
element(ESC7x4inc,[0,1,0],ESCUTIL7x4),
element(ESC7x5inc,[0,1,0],ESCUTIL7x5),
element(ESC7x6inc,[0,1,0],ESCUTIL7x6),
element(ESC7x7inc,[0,1,0],ESCUTIL7x7),
element(ESC7x10inc,[0,1,0],ESCUTIL7x10),
element(ESC8x3inc,[0,1,0],ESCUTIL8x3),
element(ESC8x4inc,[0,1,0],ESCUTIL8x4),
element(ESC8x5inc,[0,1,0],ESCUTIL8x5),
element(ESC8x6inc,[0,1,0],ESCUTIL8x6),
element(ESC8x7inc,[0,1,0],ESCUTIL8x7),
element(ESC8x10inc,[0,1,0],ESCUTIL8x10),

% folga no domingo

ESCBIN1x2 + ESCBIN1x9 + 0 #> 0,
ESCBIN2x2 + ESCBIN2x9 + 0 #> 0,
ESCBIN3x2 + ESCBIN3x9 + 0 #> 0,
```

```
ESCBIN4x2 + ESCBIN4x9 + 0 #> 0,
ESCBIN5x2 + ESCBIN5x9 + 0 #> 0,
ESCBIN6x2 + ESCBIN6x9 + 0 #> 0,
ESCBIN7x2 + ESCBIN7x9 + 0 #> 0,
ESCBIN8x2 + ESCBIN8x9 + 0 #> 0,

% folga a cada 7 dias

atmost(6,[ESCBIN1x1,ESCBIN1x2,ESCBIN1x3,ESCBIN1x4,ESCBIN1x5,ESCBIN1x6,0],0),
atmost(6,[ESCBIN1x1,ESCBIN1x2,ESCBIN1x3,ESCBIN1x4,ESCBIN1x5,ESCBIN1x6,ESCBIN1x7],0),
atmost(6,[ESCBIN1x2,ESCBIN1x3,ESCBIN1x4,ESCBIN1x5,ESCBIN1x6,ESCBIN1x7,ESCBIN1x8],0),
atmost(6,[ESCBIN1x3,ESCBIN1x4,ESCBIN1x5,ESCBIN1x6,ESCBIN1x7,ESCBIN1x8,ESCBIN1x9],0),
atmost(6,[ESCBIN1x4,ESCBIN1x5,ESCBIN1x6,ESCBIN1x7,ESCBIN1x8,ESCBIN1x9,ESCBIN1x10],0),
atmost(5,[ESCBIN2x1,ESCBIN2x2,ESCBIN2x3,ESCBIN2x4,ESCBIN2x5,0],0),
atmost(6,[ESCBIN2x1,ESCBIN2x2,ESCBIN2x3,ESCBIN2x4,ESCBIN2x5,ESCBIN2x6,ESCBIN2x7],0),
atmost(6,[ESCBIN2x2,ESCBIN2x3,ESCBIN2x4,ESCBIN2x5,ESCBIN2x6,ESCBIN2x7,ESCBIN2x8],0),
atmost(6,[ESCBIN2x3,ESCBIN2x4,ESCBIN2x5,ESCBIN2x6,ESCBIN2x7,ESCBIN2x8,ESCBIN2x9],0),
atmost(6,[ESCBIN2x4,ESCBIN2x5,ESCBIN2x6,ESCBIN2x7,ESCBIN2x8,ESCBIN2x9,ESCBIN2x10],0),
atmost(5,[ESCBIN3x1,ESCBIN3x2,ESCBIN3x3,ESCBIN3x4,ESCBIN3x5,0],0),
atmost(6,[ESCBIN3x1,ESCBIN3x2,ESCBIN3x3,ESCBIN3x4,ESCBIN3x5,ESCBIN3x6,ESCBIN3x7],0),
atmost(6,[ESCBIN3x2,ESCBIN3x3,ESCBIN3x4,ESCBIN3x5,ESCBIN3x6,ESCBIN3x7,ESCBIN3x8],0),
atmost(6,[ESCBIN3x3,ESCBIN3x4,ESCBIN3x5,ESCBIN3x6,ESCBIN3x7,ESCBIN3x8,ESCBIN3x9],0),
atmost(6,[ESCBIN3x4,ESCBIN3x5,ESCBIN3x6,ESCBIN3x7,ESCBIN3x8,ESCBIN3x9,ESCBIN3x10],0),
atmost(4,[ESCBIN4x1,ESCBIN4x2,ESCBIN4x3,ESCBIN4x4,0],0),
atmost(6,[ESCBIN4x1,ESCBIN4x2,ESCBIN4x3,ESCBIN4x4,ESCBIN4x5,ESCBIN4x6,ESCBIN4x7],0),
atmost(6,[ESCBIN4x2,ESCBIN4x3,ESCBIN4x4,ESCBIN4x5,ESCBIN4x6,ESCBIN4x7,ESCBIN4x8],0),
atmost(6,[ESCBIN4x3,ESCBIN4x4,ESCBIN4x5,ESCBIN4x6,ESCBIN4x7,ESCBIN4x8,ESCBIN4x9],0),
atmost(6,[ESCBIN4x4,ESCBIN4x5,ESCBIN4x6,ESCBIN4x7,ESCBIN4x8,ESCBIN4x9,ESCBIN4x10],0),
atmost(3,[ESCBIN5x1,ESCBIN5x2,ESCBIN5x3,0],0),
atmost(6,[ESCBIN5x1,ESCBIN5x2,ESCBIN5x3,ESCBIN5x4,ESCBIN5x5,ESCBIN5x6,ESCBIN5x7],0),
atmost(6,[ESCBIN5x2,ESCBIN5x3,ESCBIN5x4,ESCBIN5x5,ESCBIN5x6,ESCBIN5x7,ESCBIN5x8],0),
atmost(6,[ESCBIN5x3,ESCBIN5x4,ESCBIN5x5,ESCBIN5x6,ESCBIN5x7,ESCBIN5x8,ESCBIN5x9],0),
atmost(6,[ESCBIN5x4,ESCBIN5x5,ESCBIN5x6,ESCBIN5x7,ESCBIN5x8,ESCBIN5x9,ESCBIN5x10],0),
atmost(2,[ESCBIN6x1,ESCBIN6x2,0],0),
atmost(6,[ESCBIN6x1,ESCBIN6x2,ESCBIN6x3,ESCBIN6x4,ESCBIN6x5,ESCBIN6x6,ESCBIN6x7],0),
atmost(6,[ESCBIN6x2,ESCBIN6x3,ESCBIN6x4,ESCBIN6x5,ESCBIN6x6,ESCBIN6x7,ESCBIN6x8],0),
atmost(6,[ESCBIN6x3,ESCBIN6x4,ESCBIN6x5,ESCBIN6x6,ESCBIN6x7,ESCBIN6x8,ESCBIN6x9],0),
atmost(6,[ESCBIN6x4,ESCBIN6x5,ESCBIN6x6,ESCBIN6x7,ESCBIN6x8,ESCBIN6x9,ESCBIN6x10],0),
atmost(3,[ESCBIN7x1,ESCBIN7x2,ESCBIN7x3,0],0),
atmost(6,[ESCBIN7x1,ESCBIN7x2,ESCBIN7x3,ESCBIN7x4,ESCBIN7x5,ESCBIN7x6,ESCBIN7x7],0),
atmost(6,[ESCBIN7x2,ESCBIN7x3,ESCBIN7x4,ESCBIN7x5,ESCBIN7x6,ESCBIN7x7,ESCBIN7x8],0),
atmost(6,[ESCBIN7x3,ESCBIN7x4,ESCBIN7x5,ESCBIN7x6,ESCBIN7x7,ESCBIN7x8,ESCBIN7x9],0),
atmost(6,[ESCBIN7x4,ESCBIN7x5,ESCBIN7x6,ESCBIN7x7,ESCBIN7x8,ESCBIN7x9,ESCBIN7x10],0),
atmost(3,[ESCBIN8x1,ESCBIN8x2,ESCBIN8x3,0],0),
atmost(6,[ESCBIN8x1,ESCBIN8x2,ESCBIN8x3,ESCBIN8x4,ESCBIN8x5,ESCBIN8x6,ESCBIN8x7],0),
atmost(6,[ESCBIN8x2,ESCBIN8x3,ESCBIN8x4,ESCBIN8x5,ESCBIN8x6,ESCBIN8x7,ESCBIN8x8],0),
atmost(6,[ESCBIN8x3,ESCBIN8x4,ESCBIN8x5,ESCBIN8x6,ESCBIN8x7,ESCBIN8x8,ESCBIN8x9],0),
atmost(6,[ESCBIN8x4,ESCBIN8x5,ESCBIN8x6,ESCBIN8x7,ESCBIN8x8,ESCBIN8x9,ESCBIN8x10],0),

% no. de folgas e impedimentos

%Enf1F::2..2,
%Enf1I::4..4,
ESC1x1 + ESC1x2 + ESC1x3 + ESC1x4 + ESC1x5 + ESC1x6 +
ESC1x7 + ESC1x8 + ESC1x9 + ESC1x10 #= 2 + 2*4,
```

```
ESCBIN1x1 + ESCBIN1x2 + ESCBIN1x3 + ESCBIN1x4 + ESCBIN1x5 + ESCBIN1x6 +
ESCBIN1x7 + ESCBIN1x8 + ESCBIN1x9 + ESCBIN1x10 #= 2 + 4,
%Enf2F::2..2,
%Enf2I::0..0,
ESC2x1 + ESC2x2 + ESC2x3 + ESC2x4 + ESC2x5 + ESC2x6 +
ESC2x7 + ESC2x8 + ESC2x9 + ESC2x10 #= 2 + 2*0,
ESCBIN2x1 + ESCBIN2x2 + ESCBIN2x3 + ESCBIN2x4 + ESCBIN2x5 + ESCBIN2x6 +
ESCBIN2x7 + ESCBIN2x8 + ESCBIN2x9 + ESCBIN2x10 #= 2 + 0,
%Aux1F::2..2,
%Aux1I::5..5,
ESC3x1 + ESC3x2 + ESC3x3 + ESC3x4 + ESC3x5 + ESC3x6 +
ESC3x7 + ESC3x8 + ESC3x9 + ESC3x10 #= 2 + 2*5,
ESCBIN3x1 + ESCBIN3x2 + ESCBIN3x3 + ESCBIN3x4 + ESCBIN3x5 + ESCBIN3x6 +
ESCBIN3x7 + ESCBIN3x8 + ESCBIN3x9 + ESCBIN3x10 #= 2 + 5,
%Aux2F::2..2,
%Aux2I::0..0,
ESC4x1 + ESC4x2 + ESC4x3 + ESC4x4 + ESC4x5 + ESC4x6 +
ESC4x7 + ESC4x8 + ESC4x9 + ESC4x10 #= 2 + 2*0,
ESCBIN4x1 + ESCBIN4x2 + ESCBIN4x3 + ESCBIN4x4 + ESCBIN4x5 + ESCBIN4x6 +
ESCBIN4x7 + ESCBIN4x8 + ESCBIN4x9 + ESCBIN4x10 #= 2 + 0,
%Aux3F::2..2,
%Aux3I::0..0,
ESC5x1 + ESC5x2 + ESC5x3 + ESC5x4 + ESC5x5 + ESC5x6 +
ESC5x7 + ESC5x8 + ESC5x9 + ESC5x10 #= 2 + 2*0,
ESCBIN5x1 + ESCBIN5x2 + ESCBIN5x3 + ESCBIN5x4 + ESCBIN5x5 + ESCBIN5x6 +
ESCBIN5x7 + ESCBIN5x8 + ESCBIN5x9 + ESCBIN5x10 #= 2 + 0,
%Aux4F::2..2,
%Aux4I::0..0,
ESC6x1 + ESC6x2 + ESC6x3 + ESC6x4 + ESC6x5 + ESC6x6 +
ESC6x7 + ESC6x8 + ESC6x9 + ESC6x10 #= 2 + 2*0,
ESCBIN6x1 + ESCBIN6x2 + ESCBIN6x3 + ESCBIN6x4 + ESCBIN6x5 + ESCBIN6x6 +
ESCBIN6x7 + ESCBIN6x8 + ESCBIN6x9 + ESCBIN6x10 #= 2 + 0,
%Ate1F::2..2,
%Ate1I::0..0,
ESC7x1 + ESC7x2 + ESC7x3 + ESC7x4 + ESC7x5 + ESC7x6 +
ESC7x7 + ESC7x8 + ESC7x9 + ESC7x10 #= 2 + 2*0,
ESCBIN7x1 + ESCBIN7x2 + ESCBIN7x3 + ESCBIN7x4 + ESCBIN7x5 + ESCBIN7x6 +
ESCBIN7x7 + ESCBIN7x8 + ESCBIN7x9 + ESCBIN7x10 #= 2 + 0,
%Ate2F::2..2,
%Ate2I::0..0,
ESC8x1 + ESC8x2 + ESC8x3 + ESC8x4 + ESC8x5 + ESC8x6 +
ESC8x7 + ESC8x8 + ESC8x9 + ESC8x10 #= 2 + 2*0,
ESCBIN8x1 + ESCBIN8x2 + ESCBIN8x3 + ESCBIN8x4 + ESCBIN8x5 + ESCBIN8x6 +
ESCBIN8x7 + ESCBIN8x8 + ESCBIN8x9 + ESCBIN8x10 #= 2 + 0,

% no. minimo de profissionais

%MinEnf1::1..1,
2 - ESCBIN1x1 - ESCBIN2x1 #>= 1,
%MinEnf2::1..1,
2 - ESCBIN1x2 - ESCBIN2x2 #>= 1,
%MinEnf3::1..1,
2 - ESCBIN1x3 - ESCBIN2x3 #>= 1,
%MinEnf4::1..1,
2 - ESCBIN1x4 - ESCBIN2x4 #>= 1,
```

```
%MinEnf5::1..1,
2 - ESCBIN1x5 - ESCBIN2x5 #>= 1,
%MinEnf6::1..1,
2 - ESCBIN1x6 - ESCBIN2x6 #>= 1,
%MinEnf7::1..1,
2 - ESCBIN1x7 - ESCBIN2x7 #>= 1,
%MinEnf8::1..1,
2 - ESCBIN1x8 - ESCBIN2x8 #>= 1,
%MinEnf9::1..1,
2 - ESCBIN1x9 - ESCBIN2x9 #>= 1,
%MinEnf10::1..1,
2 - ESCBIN1x10 - ESCBIN2x10 #>= 1,

%MinAux1::2..2,
4 - ESCBIN3x1 - ESCBIN4x1 - ESCBIN5x1 - ESCBIN6x1 #>= 2,
%MinAux2::2..2,
4 - ESCBIN3x2 - ESCBIN4x2 - ESCBIN5x2 - ESCBIN6x2 #>= 2,
%MinAux3::2..2,
4 - ESCBIN3x3 - ESCBIN4x3 - ESCBIN5x3 - ESCBIN6x3 #>= 2,
%MinAux4::2..2,
4 - ESCBIN3x4 - ESCBIN4x4 - ESCBIN5x4 - ESCBIN6x4 #>= 2,
%MinAux5::2..2,
4 - ESCBIN3x5 - ESCBIN4x5 - ESCBIN5x5 - ESCBIN6x5 #>= 2,
%MinAux6::2..2,
4 - ESCBIN3x6 - ESCBIN4x6 - ESCBIN5x6 - ESCBIN6x6 #>= 2,
%MinAux7::2..2,
4 - ESCBIN3x7 - ESCBIN4x7 - ESCBIN5x7 - ESCBIN6x7 #>= 2,
%MinAux8::2..2,
4 - ESCBIN3x8 - ESCBIN4x8 - ESCBIN5x8 - ESCBIN6x8 #>= 2,
%MinAux9::2..2,
4 - ESCBIN3x9 - ESCBIN4x9 - ESCBIN5x9 - ESCBIN6x9 #>= 2,
%MinAux10::2..2,
4 - ESCBIN3x10 - ESCBIN4x10 - ESCBIN5x10 - ESCBIN6x10 #>= 2,

%MinTec1::0..0,
0 #>= 0,
%MinTec2::0..0,
0 #>= 0,
%MinTec3::0..0,
0 #>= 0,
%MinTec4::0..0,
0 #>= 0,
%MinTec5::0..0,
0 #>= 0,
%MinTec6::0..0,
0 #>= 0,
%MinTec7::0..0,
0 #>= 0,
%MinTec8::0..0,
0 #>= 0,
%MinTec9::0..0,
0 #>= 0,
%MinTec10::0..0,
0 #>= 0,
```

```
%MinAte1::1..1,
2 - ESCBIN7x1 - ESCBIN8x1 #>= 1,
%MinAte2::1..1,
2 - ESCBIN7x2 - ESCBIN8x2 #>= 1,
%MinAte3::1..1,
2 - ESCBIN7x3 - ESCBIN8x3 #>= 1,
%MinAte4::1..1,
2 - ESCBIN7x4 - ESCBIN8x4 #>= 1,
%MinAte5::1..1,
2 - ESCBIN7x5 - ESCBIN8x5 #>= 1,
%MinAte6::1..1,
2 - ESCBIN7x6 - ESCBIN8x6 #>= 1,
%MinAte7::1..1,
2 - ESCBIN7x7 - ESCBIN8x7 #>= 1,
%MinAte8::1..1,
2 - ESCBIN7x8 - ESCBIN8x8 #>= 1,
%MinAte9::1..1,
2 - ESCBIN7x9 - ESCBIN8x9 #>= 1,
%MinAte10::1..1,
2 - ESCBIN7x10 - ESCBIN8x10 #>= 1,

%MinMaq1::0..0,
0 #>= 0,
%MinMaq2::0..0,
0 #>= 0,
%MinMaq3::0..0,
0 #>= 0,
%MinMaq4::0..0,
0 #>= 0,
%MinMaq5::0..0,
0 #>= 0,
%MinMaq6::0..0,
0 #>= 0,
%MinMaq7::0..0,
0 #>= 0,
%MinMaq8::0..0,
0 #>= 0,
%MinMaq9::0..0,
0 #>= 0,
%MinMaq10::0..0,
0 #>= 0,

% reuniao

% FolgasN conta o numero de pessoas em folga no dia N
% dominio: 0 ate numero de profissionais
Folgas3::0..8,
Folgas4::0..8,
Folgas5::0..8,
Folgas6::0..8,
Folgas7::0..8,
Folgas10::0..8,

ESCUTIL1x3 + ESCUTIL2x3 + ESCUTIL3x3 + ESCUTIL4x3 + ESCUTIL5x3 + ESCUTIL6x3 +
ESCUTIL7x3 + ESCUTIL8x3 + 0 #= Folgas3,
```

```
ESCUTIL1x4 + ESCUTIL2x4 + ESCUTIL3x4 + ESCUTIL4x4 + ESCUTIL5x4 + ESCUTIL6x4 +
ESCUTIL7x4 + ESCUTIL8x4 + 0 #= Folgas4,
ESCUTIL1x5 + ESCUTIL2x5 + ESCUTIL3x5 + ESCUTIL4x5 + ESCUTIL5x5 + ESCUTIL6x5 +
ESCUTIL7x5 + ESCUTIL8x5 + 0 #= Folgas5,
ESCUTIL1x6 + ESCUTIL2x6 + ESCUTIL3x6 + ESCUTIL4x6 + ESCUTIL5x6 + ESCUTIL6x6 +
ESCUTIL7x6 + ESCUTIL8x6 + 0 #= Folgas6,
ESCUTIL1x7 + ESCUTIL2x7 + ESCUTIL3x7 + ESCUTIL4x7 + ESCUTIL5x7 + ESCUTIL6x7 +
ESCUTIL7x7 + ESCUTIL8x7 + 0 #= Folgas7,
ESCUTIL1x10 + ESCUTIL2x10 + ESCUTIL3x10 + ESCUTIL4x10 + ESCUTIL5x10 + ESCUTIL6x10 +
ESCUTIL7x10 + ESCUTIL8x10 + 0 #= Folgas10,

% ExisteFolgaN indica se algum profissional esta de folga no dia N
% (0 = ninguem de folga; 1 = pelo menos um de folga)
ExisteFolga3::0..1,
ExisteFolga4::0..1,
ExisteFolga5::0..1,
ExisteFolga6::0..1,
ExisteFolga7::0..1,
ExisteFolga10::0..1,

Folgas3inc #= Folgas3 + 1, element(Folgas3inc,[0,1,1,1,1,1,1,1,1],ExisteFolga3),
Folgas4inc #= Folgas4 + 1, element(Folgas4inc,[0,1,1,1,1,1,1,1,1],ExisteFolga4),
Folgas5inc #= Folgas5 + 1, element(Folgas5inc,[0,1,1,1,1,1,1,1,1],ExisteFolga5),
Folgas6inc #= Folgas6 + 1, element(Folgas6inc,[0,1,1,1,1,1,1,1,1],ExisteFolga6),
Folgas7inc #= Folgas7 + 1, element(Folgas7inc,[0,1,1,1,1,1,1,1,1],ExisteFolga7),
Folgas10inc #= Folgas10 + 1, element(Folgas10inc,[0,1,1,1,1,1,1,1,1],ExisteFolga10),

% soma de todos ExisteFolga deve ser menor que o numero de dias uteis
% (pelo menos um dia sem folga nenhuma)
ExisteFolga3 + ExisteFolga4 + ExisteFolga5 + ExisteFolga6 + ExisteFolga7 + ExisteFolga10 + 0 #< 6,

% obtencao do resultado

labelff([ESC1x1, ESC1x2, ESC1x3, ESC1x4, ESC1x5, ESC1x6, ESC1x7, ESC1x8, ESC1x9, ESC1x10,
ESC2x1, ESC2x2, ESC2x3, ESC2x4, ESC2x5, ESC2x6, ESC2x7, ESC2x8, ESC2x9, ESC2x10,
ESC3x1, ESC3x2, ESC3x3, ESC3x4, ESC3x5, ESC3x6, ESC3x7, ESC3x8, ESC3x9, ESC3x10,
ESC4x1, ESC4x2, ESC4x3, ESC4x4, ESC4x5, ESC4x6, ESC4x7, ESC4x8, ESC4x9, ESC4x10,
ESC5x1, ESC5x2, ESC5x3, ESC5x4, ESC5x5, ESC5x6, ESC5x7, ESC5x8, ESC5x9, ESC5x10,
ESC6x1, ESC6x2, ESC6x3, ESC6x4, ESC6x5, ESC6x6, ESC6x7, ESC6x8, ESC6x9, ESC6x10,
ESC7x1, ESC7x2, ESC7x3, ESC7x4, ESC7x5, ESC7x6, ESC7x7, ESC7x8, ESC7x9, ESC7x10,
ESC8x1, ESC8x2, ESC8x3, ESC8x4, ESC8x5, ESC8x6, ESC8x7, ESC8x8, ESC8x9, ESC8x10, ESCBIN1x1]),

write("Enf1: "),write(ESC1x1), write(ESC1x2), write(ESC1x3), write(ESC1x4), write(ESC1x5),
write(ESC1x6), write(ESC1x7), write(ESC1x8), write(ESC1x9), write(ESC1x10), write("\n"),
write("Enf2: "),write(ESC2x1), write(ESC2x2), write(ESC2x3), write(ESC2x4), write(ESC2x5),
write(ESC2x6), write(ESC2x7), write(ESC2x8), write(ESC2x9), write(ESC2x10), write("\n"),

write("Aux1: "),write(ESC3x1), write(ESC3x2), write(ESC3x3), write(ESC3x4), write(ESC3x5),
write(ESC3x6), write(ESC3x7), write(ESC3x8), write(ESC3x9), write(ESC3x10), write("\n"),
write("Aux2: "),write(ESC4x1), write(ESC4x2), write(ESC4x3), write(ESC4x4), write(ESC4x5),
write(ESC4x6), write(ESC4x7), write(ESC4x8), write(ESC4x9), write(ESC4x10), write("\n"),
write("Aux3: "),write(ESC5x1), write(ESC5x2), write(ESC5x3), write(ESC5x4), write(ESC5x5),
write(ESC5x6), write(ESC5x7), write(ESC5x8), write(ESC5x9), write(ESC5x10), write("\n"),
write("Aux4: "),write(ESC6x1), write(ESC6x2), write(ESC6x3), write(ESC6x4), write(ESC6x5),
write(ESC6x6), write(ESC6x7), write(ESC6x8), write(ESC6x9), write(ESC6x10), write("\n"),
```

```
write("Ate1: "),write(ESC7x1), write(ESC7x2), write(ESC7x3), write(ESC7x4), write(ESC7x5),
write(ESC7x6), write(ESC7x7), write(ESC7x8), write(ESC7x9), write(ESC7x10), write("\n"),
write("Ate2: "),write(ESC8x1), write(ESC8x2), write(ESC8x3), write(ESC8x4), write(ESC8x5),
write(ESC8x6), write(ESC8x7), write(ESC8x8), write(ESC8x9), write(ESC8x10), write("\n"),


write("\n"),

write("ExisteFolga3: "), write(ExisteFolga3), write("\n"),
write("ExisteFolga4: "), write(ExisteFolga4), write("\n"),
write("ExisteFolga5: "), write(ExisteFolga5), write("\n"),
write("ExisteFolga6: "), write(ExisteFolga6), write("\n"),
write("ExisteFolga7: "), write(ExisteFolga7), write("\n"),
write("ExisteFolga10: "), write(ExisteFolga10), write("\n"),
write("\n").
```

A.4 Saída do Eclipse

```
[eclipse 2]: programa.
Enf1: 2222001001
Enf2: 0000100010
Aux1: 1000222221
Aux2: 0010000010
Aux3: 0100000100
Aux4: 0100100000
Ate1: 0100001000
Ate2: 0010000010

ExisteFolga3: 1
ExisteFolga4: 0
ExisteFolga5: 1
ExisteFolga6: 0
ExisteFolga7: 1
ExisteFolga10: 1


yes.
```