UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E COMPUTAÇÃO CIENTÍFICA
DEPARTAMENTO DE MATEMÁTICA APLICADA

EDUARDO GARCIA DE OLIVEIRA CARIAS

Alocação de crianças em horários de atendimento na Casa da Criança Paralítica de Campinas

Campinas
22/11/2019

EDUARDO GARCIA DE OLIVEIRA CARIAS

Alocação de crianças em horários de atendimento na Casa da Criança Paralítica de Campinas

Monografia apresentada ao Instituto de Matemática, Estatística e Computação Científica da Universidade Estadual de Campinas como parte dos requisitos para obtenção de créditos na disciplina Projeto Supervisionado, sob a orientação do(a) Prof. Paulo J. S. Silva.

RESUMO

O objetivo deste projeto é explorar um problema de alocação dupla com base em dados da ONG "Casa da Criança Paralítica de Campinas". A resolução deste problema foi apresentada como um desafio da empresa UniSoma para estudantes de graduação e pós-graduação com objetivo de facilitar rotina dos funcionários da ONG. Para a solução aqui discutida foram desenvolvidos um modelo de alocação clássico e um baseado em teoria de fluxos em rede, ambos capazes de encontrar a solução global do problema, porém com tempos distintos de execução. A princípio é discutida a construção dos modelos e em seguida são discutidas as vantagens e desvantagens de ambos assim como as soluções encontradas.

Palavras-chave: Casa da Criança Paralítica, UniSoma, Fluxos em Rede, Problema de Alocação, Programação Inteira

ABSTRACT

The goal of this project is to explore a specific double assignment problem based on data from the NPO "Casa da Criança Paralítica de Campinas". The solution for this problem was presented as a challenge to graduate and post-graduate students in order to to facilitate the NPO's employees in their work routine. Two models were created for the solution at hand, one classic assignment model and another based in network flows, both capable of finding the problem's global solution, although with different execution times. At first the construction of the models is discussed and then the advantages and disadvantaged of each model as well as the solutions found.

Key-words: Casa da Criança Paralítica, UniSoma, Network Flow, Assignment Problem, Integer Programming

Conteúdo

1 INTRODUÇÃO

A UniSoma é uma empresa especializada em advanced analytics que comercializa soluções aplicando métodos de otimização para o auxiliar tomadas de decisões em diversos setores da indústria. Desde 2017, a UniSoma desafia estudantes de graduação e pós-graduação a criarem softwares matemáticos com intuito de auxiliar uma ONG escolhida pela empresa. Neste ano de 2019 a ONG escolhida foi a Casa da Criança Paralítica de Campinas

A Casa da Criança Paralítica de Campinas (CCP) é uma instituição de utilidade pública e sem fins lucrativos. Ela oferece tratamento de reabilitação gratuito para crianças com deficiência física e comprometimento neurológico. Atualmente os funcionários da ONG planejam os atendimentos semanais das crianças de forma empírica em planilhas, um processo que pode levar horas e que não garante que a melhor alocação possível foi de fato realizada, ou seja, a ONG tem em mãos um problema clássico de alocação, muito bem conhecido e estudado na área de otimização. Apesar de ser um problema muito antigo, datando em torno de 1781, quando foi estudado por Gaspard Monge na forma de um problema de transporte [4], ainda são poucas as pessoas fora do meio acadêmico que tem conhecimento sobre maneiras de resolvê-lo, o que geralmente leva a uma solução empírica e muitas vezes longe do ideal.

1.1 Definição do Problema

O intuito do projeto é criar uma ferramenta que resolva o problema de alocação de crianças em horários de atendimento pela equipe técnica da ONG, maximizando os atendimentos e minimizando o tempo de espera dos pacientes caso haja mais de um atendimento no dia.

Para isso, será discutido o uso de dois modelos distintos para a solução do problema. A hipótese é que com o uso de uma ferramenta de otimização, seria possível resolver o problema em um tempo muito mais razoável do que atualmente, liberando os funcionários para outras tarefas de maior importância e proporcionando melhorias na qualidade de vida das crianças que dependem dos atendimentos oferecidos.

O primeiro modelo explorado é um modelo de alocação simples de maximização com variáveis inteiras, já o segundo é um modelo que trata o problema como fluxos em redes [5] [2] com o objetivo de maximizar o fluxo de atendimentos de médicos para crianças minimizando o tempo de espera.

1.2 Premissas

Serão fornecidas fontes de dados com o cadastro de todas as crianças e médicos, assim como a relação entre os médicos e suas especialidades, a demanda de cada criança para cada especialidade, e a disponibilidade de cada médico e criança durante a semana. O planejamento é feito semanalmente e deve satisfazer a demanda semanal de atendimentos de cada paciente.

O planejamento deve respeitar a disponibilidade de comparecimento na instituição por parte das crianças (dia e horário), assim como a disponibilidade de atendimento por parte do profissional de saúde. Também vale ressaltar que o tempo de atendimento é de 30 minutos para qualquer uma das especialidades.

Para cada paciente, todos os atendimentos do dia devem estar distribuídos em um único período, ou seja, se houver um atendimento no período da manhã, então todos os outros atendimentos no dia também devem ser de manhã. Além disso, deve haver um intervalo de pelo menos um dia entre atendimentos da mesma especialidade.

1.3 Objetivo

A solução do problema apresentado já foi desenvolvida por diversos grupos que participaram do desafio descrito na introdução. O grupo vencedor criou seu projeto em R e Python e além do modelo também desenvolveu uma interface gráfica de fácil uso para os usuários. O software desenvolvido tem custo zero e já foi apresentado aos funcionários da Casa da Criança Paralítica. Espera-se que o mesmo seja utilizado em um futuro próximo. Também vale salientar que nas premissas originais do problema a minimização do tempo de espera entre atendimentos deveria ser tratada de forma secundária através da restrição de que só poderiam ocorrer atendimentos para uma determinada criança em um dos períodos do dia.

Dito isso, o intuito deste projeto não é de fato entregar a solução para a ONG, mas dar uma outra abordagem ao problema com o diferencial de que, além de resolver o problema de alocação, o tempo de espera entre atendimentos também seja minimizado diretamente.

Como objetivo secundário, será feita a comparação entre dois modelos desenvolvidos para analisar as possíveis vantagens e/ou desvantagens de um modelo de fluxos em redes para solução de problemas inteiros como este comparado a um modelo de alocação simples e intuitivo de programação linear.

2 METODOLOGIA

2.1 Estruturação dos dados para o modelo em fluxos

A ideia básica para a modelagem em fluxos é tratar o problema como um problema de transporte com entrepostos. O modelo geral para um problema como este consiste em definir nós de saída, nós intermediários (entrepostos) e nós de entrada de forma a obter um grafo como o representado na Figura 1 e considerando as seguintes restrições básicas[3]:

$$\text{Min} \sum c_{i,j} x_{i,j}$$

sujeito a

$$\sum_{\text{saída}} x_{i,j} = s_i \text{ Para os nós de origem (1)}$$

$$\sum_{\text{saída}} x_{i,j} - \sum_{\text{entrada}} x_{i,j} = 0 \text{ Para os nós de entreposto(2)}$$

$$\sum_{\text{entrada}} x_{i,j} = d_j \text{ Para os nós de destino(3)}$$

Onde

- $x_{i,j}$ = Quantidade transportada do nó i para j
- $c_{i,j}$ = Custo de transporte do nó i para j
- s_i = Disponibilidade no nó de origem i
- d_j = Demanda no nó de destino j

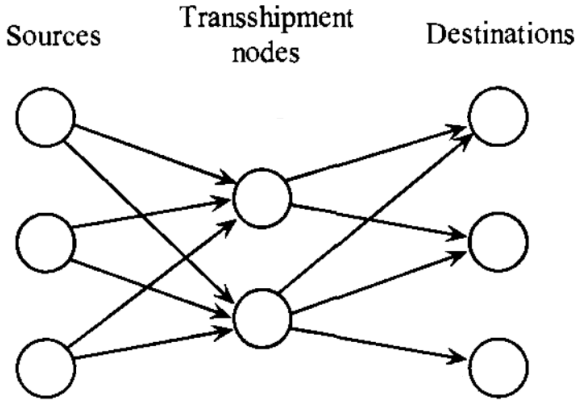

Figura 1: Exemplo de um problema com entrepostos [2]

Inicialmente, para o problema em questão vamos considerar os nós de origem como sendo os médicos, os entrepostos sendo a combinação de dias e horários de atendimento (chamarei a combinação de dia e horário de período) e os nós de destino sendo as crianças que serão atendidas, assim, $x_{i,j}$ é uma variável binária representando o comparecimento do médico i no período j ou o atendimento da criança j no período i .

Com isso podemos definir que o custo de "transporte"passa a ser uma constante unitária, ou seja, o custo de atendimento ou não de cada criança ou comparecimento de cada médico no período é igual para todas as combinações possíveis. Além disso, o fluxo tem sentido único, ou seja, não existe entrada para os nós de origem e não existe saída para os nós de destino.

Agora temos uma estrutura básica para o modelo, porém ainda existem dois problemas relacionados aos dados de entrada em mãos e a estrutura do modelo em si:

1. Os dados de input são passados com base em horário e dia separadamente, ou seja, é necessário converter cada combinação de horário e dia para um índice unificado, chamado de período.

2. A demanda de cada criança é dada por especialidade. Apesar de ser possível garantir que a demanda total será atendida com a equação 3, não podemos garantir o atendimento individual de cada criança para a especialidade, visto que não há conexão direta entre os nós de origem e destino, ou seja, a informação de qual médico está atendendo qual criança está sendo perdida nos nós intermediários.

A solução para o primeiro problema é simples e é representada pela Tabela 1. Já para a solução do segundo problema apresentado, a estratégia utilizada foi replicar os nós intermediários do problema inicial para cada especialidade possível. Assim, ao invés de termos cada período da Tabela 1 descrito por um nó intermediário, teremos em cada nó uma tupla (E,P) representando uma combinação da especialidade e do período.

Com isso, definimos que só podem existir arestas ligando os nós de médicos às especialidades que os representam, e agora, ao criar a aresta ligando uma criança a um período, ainda teremos a informação de atendimento da criança pela especialidade demandada. O interessante aqui é que o modelo acomodaria com facilidade a possibilidade de existir mais do que um médico por especialidade e, caso ocorresse, seria necessário um número reduzido de variáveis comparado a um modelo de alocação comum, visto que a variável que liga uma criança a tupla (E,P) é independente do médico. A Figura 2 mostra um exemplo de como serão modelados os atendimentos.

Tabela 1: Tabela de mapeamento dos indices para horários e dias

Horário	Segunda	Terça	Quarta	Quinta	Sexta
7:30:00	1	20	39	58	77
8:00:00	2	21	40	59	78
8:30:00	3	22	41	60	79
9:00:00	4	23	42	61	80
9:30:00	5	24	43	62	81
10:00:00	6	25	44	63	82
10:30:00	7	26	45	64	83
11:00:00	8	27	46	65	84
11:30:00	9	28	47	66	85
12:00:00	10	29	48	67	86
12:30:00	11	30	49	68	87
13:00:00	12	31	50	69	88
13:30:00	13	32	51	70	89
14:00:00	14	33	52	71	90
14:30:00	15	34	53	72	91
15:00:00	16	35	54	73	92
15:30:00	17	36	55	74	93
16:00:00	18	37	56	75	94
16:30:00	19	38	57	76	95

Com a tabela podemos concluir algumas relações que são utilizadas no código:

$P_D \in ((D-1)19+1, D19)$onde P_n são os índices representando os horários do dia D

$P_{Dt} \in ((D-1)19+10, D19)$onde P_{Dt} são os períodos da tarde do dia D.

$H = P - 19(D-1)$é o horário dado um período P e um dia D

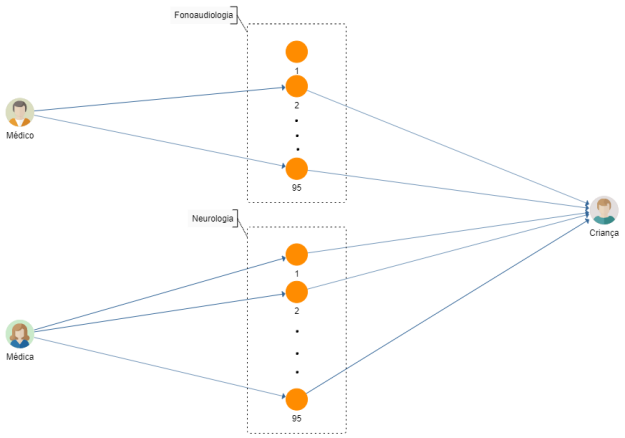

Figura 2: Exemplo de fluxo de atendimento [1]

2.2 Conjuntos e Símbolos Utilizados nos Modelos

Constam a seguir todos os conjuntos e símbolos utilizados para descrever os modelos:

Utilizados em ambos os modelos

- $C = \{c_1, c_2, ..., c_{70}\}$ onde c_n é a n-ésima criança
- $M = \{m_1, m_2, ..., m_6\}$ onde m_n é o n-ésimo profissional
- $E = \{e_1, e_2, ..., e_6\}$ onde e_n é a n-ésima especialidade
- $H = \{h_1, h_2, ..., h_{19}\}$ onde h_n é o n-ésimo horário do dia
- $D = \{d_1, d_2, ..., d_5\}$ onde d_n é o n-ésimo dia da semana
- $ME_e = \{m_1, m_2, ..., m_n : m_n$é da especialidade e$\}$
- $PM = \{h_1, h_2, ..., h_n : h_n$é um horário da manhã$\}$
- $BigM = 500$ constante
- $A =$ peso arbitrário atribuído ao atendimento.

Utilizados exclusivamente no modelo de fluxos

- $P = \{p_1, p_2, ..., p_{95}\}$ onde p_n representa um horário e dia
- $EP = \{(p_1, e_1), ..., (p_{95}, e_6)\}$
  onde (p_n, e_m) representa um $p_n \in P$ relacionado a especialidade $e_m \in E$
- $HR = \{h_n : h_n = p_n - 19(d_n - 1)\}$ é o horário tal que $p_n \in P$ e $d_n \in D$

2.3 Modelo de Fluxos

Seguem as variáveis e restrições utilizadas para construção do modelo baseado em teoria de fluxos em rede. A construção deste se baseia na estrutura discutida na seção anterior para os índices e variáveis utilizadas.

Variáveis

- $atendimento_{c,ep}$ é uma variável binária que representa o atendimento ou não da criança c pela especialidade e período (e,p). Para o modelo abaixo vamos assumir que a variável só pode assumir valor diferente de zero caso haja demanda da criança c pela especialidade representada por (e,p).
- $comparecimento_{m,ep}$ é uma variável binária que representa o comparecimento ou não do médico m da especialidade e no período p. Analogamente a variável anterior, o comparecimento só pode assumir valor caso determinado médico esteja disponível no período p representado por (e,p).
- $atendeHoje_{c,d,e}$ é a variável binária representando o atendimento ou não da criança c no dia d pela especialidade e. Se houver atendimento a variável deve assumir valor um e quando se não houver atendimento deve assumir valor zero.
- $atendeTarde_{c,d}$ é a variável binária que representa o atendimento da criança c no período da tarde do dia d. Análogo a variável anterior.
- $hMin_{c,d}$ é o menor horário de atendimento da criança c no dia d.
- $hMax_{c,d}$ é o maior horário de atendimento da criança c no dia d
- $tempo_{c,d}$ é o tempo de espera entre atendimentos.

Restrições

$$\text{Max} \sum_{c \in C, ep \in EP} A * atendimento_{c,ep} - \sum_{c \in C, d \in D} tempo_{c,d}$$

sujeito a

Garante que o fluxo total nos nós intermediários é zero.

$$\sum_{c \in C} atendimento_{c,ep} = \sum_{m \in M} comparecimento_{m,ep}, \forall ep \in EP \tag{1}$$

Limita a quantidade de atendimentos na especialidade a demanda da criança pela mesma

$$\sum_{ep \in EP | e = esp} atendimento_{c,ep} \leq demanda_{c,esp} \, \forall c \in C, esp \in E \tag{2}$$

Apenas uma criança pode ser atendida de cada vez para cada especialidade.

$$\sum_{ep \in EP | p = peri} atendimento_{c,ep} \leq 1 \;\; \forall c \in C, peri \in P \tag{3}$$

Um médico pode atender de cada vez para uma especialidade.

$$\sum_{ep \in EP | p = peri} comparecimento_{m,ep} \leq 1 \;\; \forall m \in M, peri \in P \tag{4}$$

Garante que atendeTarde é igual a um casa ocorra algum atendimento no período da manhã.

$$\sum_{ep \in EP | p \in PM} atendimento_{c,ep} \leq BigM * atendeTarde_{c,d} \;\; \forall c \in C, d \in D \tag{5}$$

Impede atendimento a tarde caso tenha ocorrido de manhã e vice-versa.

$$\sum_{ep \in EP | p \notin PM} atendimento_{c,ep} \leq BigM * (1 - atendeTarde_{c,d}) \;\; \forall c \in C, d \in D \tag{6}$$

Só pode ocorrer um atendimento de cada especialidade no dia para cada criança.

$$\sum_{ep \in EP | e = esp, p \in D} atendimento_{c,ep} \leq 1 \;\; \forall c \in C, d \in D, esp \in E \tag{7}$$

Garante que atendeHoje é igual a um casa ocorra algum atendimento da especialidade e no dia d.

$$\sum_{ep \in EP | e = esp, p \in D} atendimento_{c,ep} \leq BigM * atendeHoje_{c,d,esp} \;\; \forall c \in C, d \in D, esp \in E \tag{8}$$

Se houve atendimento ontem, não pode haver hoje para a mesma especialidade.

$$atendeHoje_{c,d-1,e} \leq (1 - atendeHoje_{c,d,e}) \ \ \forall c \in C, e \in E, d \in \{D \setminus d_1\} \tag{9}$$

Se houver atendimento amanhã, não pode haver hoje para a mesma especialidade.

$$atendeHoje_{c,d+1,e} \leq (1 - atendeHoje_{c,d,e}) \ \ \forall c \in C, e \in E, d \in \{D \setminus d_5\} \tag{10}$$

Garante que horarioMaximo é maior ou igual a qualquer horário de atendimento.
Caso não ocorra atendimento, a variável fica livre.

$$hMax_{c,d} \geq HR * atendimento_{c,ep} \ \ \forall c \in C, m \in M, h \in H, d \in D \tag{11}$$

Garante que horarioMinimo é menor ou igual a qualquer horário de atendimento.
Caso não ocorra atendimento, a variável fica livre.

$$hMin_{c,d} \leq HR * atendimento_{c,ep} + 19(1 - atendimento_{c,ep}) \tag{12}$$

Define o tempo de espera descontando os atendimentos entre hMax e hMin.

$$tempo_{c,d} = (hMax_{c,d} - hMin_{c,d}) - \sum_{ep \in EP | p \in D} (atemdimento_{c,ep} - 1) \tag{13}$$

$$atendimento_{c,ep} \in \{0,1\}; comparecimento_{m,ep} \in \{0,1\}$$
$$atendeTarde_{c,d} \in \{0,1\}; \ \ atendeHoje_{c,d,e} \in \{0,1\}$$
$$0 \leq hMax_{c,d} \leq 19; \ \ 0 \leq hMin_{c,d} \leq 19$$

2.4 Modelo de Alocação

Para o modelo de otimização simples, será considerada apenas uma variável de decisão binária representando o atendimento, que será indexada em médico, criança, horário e dia. Dessa forma o tratamento feito para os dados no modelo de fluxo não se faz necessário.

Variáveis

- $atendimentos_{c,m,d,h}$ é uma variável binária que representa o atendimento ou não da criança c pelo médico m no dia d e horário h. Para o modelo abaixo vamos assumir que esta variável só pode assumir algum valor diferente de zero caso tanto o médico m quanto a criança c estejam disponíveis no dia d e horário h
- $atendeHoje_{c,d,e}$ é a variável binária representando o atendimento ou não da criança c no dia d pela especialidade e. Se houver atendimento a variável deve assumir valor um e quando se não houver atendimento deve assumir valor zero.
- $atendeTarde_{c,d}$ é a variável binária que representa o atendimento da criança c no período da tarde do dia d. Análogo a variável anterior.
- $hMin_{c,d}$ é o menor horário de atendimento da criança c no dia d.
- $hMax_{c,d}$ é o maior horário de atendimento da criança c no dia d
- $tempo_{c,d}$ é o tempo de espera entre atendimentos.

Restrições

$$\text{Max} \sum_{c\in C, m\in M, d\in D, h\in H} A * atendimentos_{c,m,d,h} - \sum_{c\in C, d\in D} tempo_{c,d}$$

sujeito a

Limita os atendimentos por especialidade de acordo com a demanda.

$$\sum_{d\in D, h\in H, m\in ME_e} atendimentos_{c,m,d,h} \leq demanda_{c,e} \; \forall c \in C, e \in E \tag{14}$$

Garante que só haverá uma criança por atendimento.

$$\sum_{c\in C} atendimentos_{c,m,d,h} \leq 1 \;\; \forall m \in M, d \in D, h \in H \tag{15}$$

Garante que só haverá um médico por atendimento.

$$\sum_{m\in M} atendimentos_{c,m,d,h} \leq 1 \;\; \forall c \in C, d \in D, h \in H \tag{16}$$

Garante que atendeTarde é igual a um casa ocorra algum atendimento no período da manhã.

$$\sum_{m\in M, h\in PM} atendimentos_{c,m,d,h} \leq BigM * atendeTarde_{c,d} \;\; \forall c \in C, d \in D \tag{17}$$

Impede atendimento a tarde caso tenha ocorrido de manhã e vice-versa.

$$\sum_{m\in M, h\notin PM} atendimentos_{c,m,d,h} \leq BigM * (1 - atendeTarde_{c,d}) \;\; \forall c \in C, d \in D \tag{18}$$

Só pode ocorrer um atendimento de cada especialidade no dia para cada criança.

$$\sum_{m\in ME_e, h\in H} atendimentos_{c,m,d,h} \leq 1 \;\; \forall c \in C, d \in D, e \in E \tag{19}$$

Garante que atendeHoje é igual a um casa ocorra algum atendimento da especialidade e no dia d.

$$\sum_{m\in ME_e, h\in H} atendimentos_{c,m,d,h} \leq BigM * atendeHoje_{c,d,e} \;\; \forall c \in C, d \in D, e \in E \tag{20}$$

Se houve atendimento ontem, não pode haver hoje para a mesma especialidade.

$$atendeHoje_{c,d-1,e} \leq (1 - atendeHoje_{c,d,e}) \;\; \forall c \in C, e \in E, d \in \{D \setminus d_1\} \tag{21}$$

Se houver atendimento amanhã, não pode haver hoje para a mesma especialidade.

$$atendeHoje_{c,d+1,e} \leq (1 - atendeHoje_{c,d,e}) \ \ \forall c \in C, e \in E, d \in \{D \setminus d_5\} \tag{22}$$

Garante que horarioMaximo é maior ou igual a qualquer horário de atendimento.
Caso não ocorra atendimento, a variável fica livre.

$$hMax_{c,d} \geq HR * atendimentos_{c,m,d,h} \ \ \forall c \in C, m \in M, h \in H, d \in D \tag{23}$$

Garante que horarioMinimo é menor ou igual a qualquer horário de atendimento.
Caso não ocorra atendimento, a variável fica livre.

$$hMin_{c,d} \leq HR * atendimentos_{c,m,d,h} + 19(1 - atendimentos_{c,m,d,h}) \tag{24}$$

Define o tempo de espera descontando os atendimentos entre hMax e hMin.

$$tempo_{c,d} = (hMax_{c,d} - hMin_{c,d}) - \sum_{ep \in EP | p \in D} (atemdimentos_{c,d,m,d,h} - 1) \tag{25}$$

$$atendimentos_{c,m,d,h} \in \{0,1\}; \ \ atendeTarde_{c,d} \in \{0,1\}; \ \ atendeHoje_{c,d,e} \in \{0,1\}$$
$$0 \leq horarioMaximo_{c,d} \leq 19; \ \ 0 \leq horarioMinimo_{c,d} \leq 19$$

2.5 Alternativa Para a Minimização de Tempo

Para ambos os modelos também foi estudada uma formulação alternativa para a função objetivo: Ao invés de tratar o intervalo de espera entre atendimentos como algo a ser minimizado, também foi estudada a possibilidade de tratar o intervalo como uma restrição, de forma que seja definido um tempo máximo de espera entre atendimentos.
Com isso temos uma função objetivo simplificada na seguinte forma para ambos os modelos:

$$\text{Max} \sum atendimentos$$

E a adição da seguinte restrição:

$$tempo_{c,d} \leq H_{max} \;\; \forall c \in C, d \in D$$

Ou ainda:

$$\sum_{c \in C, d \in D} tempo_{c,d} \leq H_{max} \;\; \forall c \in C, d \in D$$

Onde H_{max} é o horário máximo de espera definido pelo usuário da ferramenta. No primeiro caso pode ser definido um tempo máximo de espera para cada criança no dia de atendimento, no segundo seria definido o tempo máximo de espera total na semana toda de atendimento.

2.6 Implementação e Informações Técnicas

Ambos os modelos foram implementados em Julia, uma linguagem de programação dinâmica de alto nível. Os dados de input foram lidos utilizando o pacote ExcelReaders, que coleta os dados da planilha de input e os organiza em estruturas de Data Frame.

Com os dados lidos e organizados com o ExcelReaders, foram construídos os conjuntos necessários que por sua vez foram utilizados para a construção das variáveis e cálculos de disponibilidade e demanda (Apêndices A,B e C). Todas as informações de disponibilidade e demanda também foram lidas da planilha de input e tratadas internamente em Julia.

Para a modelagem do problema em sí foi utilizado o pacote JuMP e para a solução, o solver Gurobi. Ambos os modelos foram resolvidos com uma máquina com 8gb de RAM e processador Intel Core i7-5500 CPU 2.40GHz. Constam a seguir os resultados obtidos.

3 ANÁLISE DOS RESULTADOS

3.1 Minimização de Tempo na Função Objetivo

Os resultados abaixo se referem a solução do problema utilizando os modelos como definidos na metodologia e incluído o tempo de espera entre atendimentos na função objetivo.

Vale salientar que só foi disponibilizada uma amostra dos dados de entrada para o problema, sendo assim, os resultados abaixo se referem exclusivamente a solução encontrada pelos modelos desenvolvidos em função apenas de tais dados.

Se $A \geq 25$, onde A é o peso definido, dado que a demanda total de atendimentos no caso resolvido é de 401 atendimentos na semana, então o maior valor possível para a função objetivo é igual a 10025. Neste caso, um único atraso equivale a um GAP inferior a 0,01%, que é o GAP mínimo considerado padrão pelo Gurobi. Ou seja, a partir desse valor uma solução com um único atraso já seria considerada como a melhor solução possível, o que não é verdade. Sendo assim, os pesos considerados para o atendimento na função objetivo foram valores entre 1 e 24.

O modelo de alocação foi capaz de encontrar solução com um GAP de 0%, o que implica em 100% das demandas atendidas e nenhuma espera entre os atendimentos na semana.

No caso do modelo de alocação não foi possível encontrar uma relação direta entre o peso dos atendimentos com o tempo de execução, como visto na Figura 3. A média do tempo de execução foi de aproximadamente 2 minutos.

Figura 3: Tempo de execução vs. Peso para o modelo de alocação

O modelo de fluxos em rede também encontrou uma solução com GAP de 0%, mas neste caso foi possível observar uma relação entre o peso escolhido para a função objetivo e o tempo de execução, como visto na Figura 4. O menor tempo de execução foi 110 segundos com peso 24 para os atendimentos.

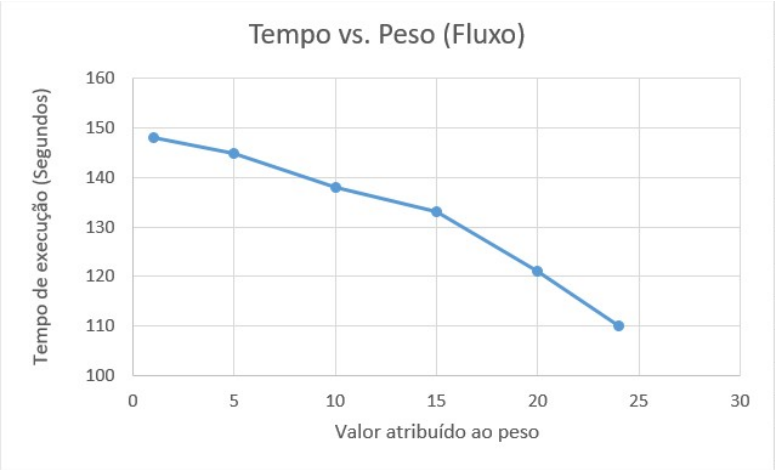

Figura 4: Tempo de execução vs. Peso para o modelo de fluxos em rede

3.2 Minimização de Tempo Como Restrição

De forma a obter resultados comparáveis aos obtidos considerando o tempo na função objetivo, H_{max} foi definido como zero em ambas as alternativas consideradas para a restrição, dado que o GAP encontrado nas soluções anteriores foi 0%. Dessa forma a restrição diz estritamente que não pode existir tempo de espera entre os atendimentos. Em um caso mais geral, caso não seja conhecido o tempo mínimo de espera tal que o problema seja factível, a estratégia seria definir o tempo máximo como zero, ou outro valor considerado razoável, e aumentá-lo a cada rodada até encontrar um tempo de espera tal que exista solução.

Ao restringir o tempo de espera isoladamente por criança e por dia, como descrito pela primeira restrição na seção 2.5, o modelo de alocação conseguiu encontrar solução em 164 segundos enquanto o modelo de fluxos fez o mesmo em torno de 78 segundos

Restringindo o tempo de espera total entre atendimentos, como descrito pela segunda restrição da seção 2.5, o modelo de alocação encontra solução em 174 segundos e o modelo de fluxos em torno de 80 segundos

4 CONCLUSÃO

Ambos os modelos desenvolvidos foram capazes de resolver o probelma apresentado em uma fração do tempo que seria gasto encontrando uma solução empírica, e usando uma fração do esforço. Dito isso, o único cenário estudado foi aquele referente aos dados de entrada disponibilizados para a solução do desafio. Sendo assim, não é possível dizer qual modelo se sairía melhor no caso geral.

Apesar disso, podemos destacar algumas vantagens e desvantagens quando tratando do problema específico em mãos:

Considerando o tempo de espera como uma restrição, houve uma diferença clara entre os dois modelos apresentados. O modelo de fluxos em rede foi capaz de encontrar uma solução de igual qualidade e aproximadamente duas vezes mais rápido quando comparado ao modelo de alocação, mas se tratando das duas possibilidades de modelagem da restrição, a diferença foi mínima.

Tratando a minimização do tempo de espera na função objetivo, quando levamos em consideração a média aproximada do tempo de execução variando os pesos para a função de atendimentos, a diferença entre os modelos é mínima. O modelo em fluxos conseguiu resolver o problema com um tempo de execução médio de aproximadamente 2 minutos e 10 segundos enquanto o modelo de alocação fez o mesmo em aproximadamente 2 minutos. Porém, dada a relação entre o peso atribuído e o tempo de execução do modelo de fluxos, este acaba sendo mais vantajoso para maiores pesos.

Lembrando mais uma vez que isso vale apenas para o problema específico dado, não havendo garantia de que o tempo de execução manteria a mesma relação com o peso atribuído e nem de que o modelo de fluxos realmente seria mais rápido que o de alocação.

REFERÊNCIAS

[1] yworks yed live. Disponível em: <https://www.yworks.com/yed-live/>, Acesso em: 10/11/2019.

[2] BAZAARA Mohktar. **Linear Programming and Network Flows**. Wiley, 2011.

[3] Gregory Okiemute, Agadaga Nsikan, and Nsikan Akpan. Transshipment problem and its variants: A review. 03 2017.

[4] Alexander Schrijver. On the history of combinatorial optimization (till 1960). *Handbooks in Operations Research and Management Science*, 12, 08 2001.

[5] WINSTON Wayne. *Introduction to Mathematical Programming: Operations Research*, chapter **TRANSPORTATION, ASSIGNMENT, AND TRANSSHIPMENT PROBLEMS**. Cengage Learning, Inc, 2003.

ANEXOS E APÊNDICES

A seguir constam os códigos auxiliares utilizados para a contrução dos modelos em Julia, assim como as tabelas de input dos dados

A - Códigos utilizados no modelo padrão

A.1 Cálculo de disponibilidade dos profissionais

```
using ExcelReaders, JuMP, Dates

function getDispProfissional(M::Array{String,1}, D::Array{Any,2}, H::Array{Any,2})

    f = openxl(joinpath(@__DIR__,"modelo-de-dados-ccp.xlsx"));
    #Horários de manhã
    h1 = readxl(f, "Auxiliar!F2:F11");
    #Horários de tarde
    h2 = readxl(f, "Auxiliar!F12:F20");
    disponibilidade = JuMP.Containers.DenseAxisArray{Int64}(undef, M, D, H);
    fill!(disponibilidade, zero(Int64));

    mapDias = Dict{Int64,String}();
    for i in 2:6
        push!(mapDias, i => D[i-1])
    end
    for m in M
        sheet = m*"!A2:F20";
        grade = readxl(f, sheet);
        for h = 1:19
            hora = grade[h,1]
            for d = 2:6
                if grade[h,d] != "Indisponível"
                    disponibilidade[m,mapDias[d],hora] = 1;
                end
            end
        end
    end

    return disponibilidade
end
```

A.2 Cálculo de disponibilidade das crianças

```
using ExcelReaders, JuMP, Dates

function getDispCrianca(C::Array{Any,2}, D::Array{Any,2}, H::Array{Any,2})

    f = openxl(joinpath(@__DIR__,"modelo-de-dados-ccp.xlsx"));
    #Horários de manhã
    h1 = readxl(f, "Auxiliar!F2:F11");
    #Horários de tarde
    h2 = readxl(f, "Auxiliar!F12:F20");

    disponibilidade = JuMP.Containers.DenseAxisArray{Int64}(undef, C, D, H);
    fill!(disponibilidade, zero(Int64));
    #Disponibilidade da criança
    dispExcel = readxl(f, "Disponibilidade da Criança!A2:E75");

    for i = 1:length(dispExcel[:,1])
       crianca = dispExcel[i,1]
       dia = dispExcel[i,2];
       peri = dispExcel[i,3];
       ini = dispExcel[i,4];
       fim = dispExcel[i,5];

        if peri == "Manhã"
           if dia == "Todos"
               for d in D, h in h1
                   disponibilidade[crianca,d,h] = 1;
               end
           else
               for h in h1
                   disponibilidade[crianca,dia,h1] = 1;
               end
           end
       elseif peri == "Tarde"
           if dia == "Todos"
               for d in D, h in h2
                   disponibilidade[crianca,d,h] = 1;
                end
            else
                for h in h2
                    disponibilidade[crianca,dia,h] = 1;
                end
```

```
            end
        elseif peri == "Ambos"
            if dia == "Todos"
                for d in D, h in H
                    disponibilidade[crianca,d,h] = 1;
                end
            else
                for h in H
                    disponibilidade[crianca,dia,h]
                end
            end
        elseif peri == "Horário"
            if fim >= Dates.Time(12,00,00)
                peri = "Tarde"
            else
                peri = "Manhã"
            end
            if dia == "Todos"
                for d in D, h in H
                    if h >= ini && h <= fim
                        disponibilidade[crianca,d,h] = 1;
                    end
                end
            else
                for h in H
                    if h >= ini && h <= fim
                        disponibilidade[crianca,dia,h] = 1;
                    end
                end
            end
         end
    end

    return disponibilidade;
end
```

A.3 Cálculo de disponibilidade combinada

```
include(joinpath(@__DIR__,"/disponibilidadeCriancas.jl"));
include(joinpath(@__DIR__,"/disponibilidadeProfissionais.jl"));

using ExcelReaders, JuMP, Dates
```

```
5
6
7 function getDisponibilidade(C::Array{Any,2}, M::Array{String,1}, H::Array{Any,2},
  ↪  D::Array{Any,2})
8
9     disponibilidadeCrianca = getDispCrianca(C,D,H);
10    disponibilidadeProfissional = getDispProfissional(M,D,H);
11
12    disponibilidade = JuMP.Containers.DenseAxisArray{Int64}(undef, C, M, H, D);
13    fill!(disponibilidade, false);
14
15    for c in C, m in M, d in D, h in H
16        if disponibilidadeProfissional[m,d,h] == disponibilidadeCrianca[c,d,h]
17            disponibilidade[c,m,h,d] = true;
18        end
19    end
20
21    return disponibilidade
22 end
```

B - Códigos utilizados no modelo em fluxos

B.1 Cálculo de disponibilidade dos profissionais

```
using ExcelReaders, JuMP, Dates

function getDispEspecialidade(M::Array{String,1}, E::Array{String,1},
    medicoEspecialidade::JuMP.Containers.DenseAxisArray{Bool})
    f = openxl(joinpath(@__DIR__,"modelo-de-dados-ccp.xlsx"));
    P = 1:95
    disponibilidade = JuMP.Containers.DenseAxisArray{Int64}(undef,M,P);
    fill!(disponibilidade, zero(Int64));

    for m in M, e in E
        if medicoEspecialidade[m,e]
            sheet = m*"!A2:F20";
            grade = readxl(f, sheet);
            for p in 1:19
                for d in 1:5
                    peri = p + (d-1)*19
                    if grade[p,d+1] != "Indisponível"
                        disponibilidade[m,peri] = 1;
                    end
                end
            end
        end
    end

    return disponibilidade
end
```

B.2 Cálculo de disponibilidade das crianças

```
using ExcelReaders, JuMP, Dates

function getDispCrianca(C::Array{Any,2}
                        ,mapDias::Dict{String,Int64}
                        ,mapPeriInicial::Dict{Dates.Time,Int64}
                        ,horarioDoPeriodo::JuMP.Containers.DenseAxisArray{Bool})

    f = openxl(joinpath(@__DIR__,"modelo-de-dados-ccp.xlsx"));
    P = 1:95;
    disponibilidade = JuMP.Containers.DenseAxisArray{Int64}(undef, C, P);
    fill!(disponibilidade, zero(Int64));
    #Disponibilidade da criança
    dispExcel = readxl(f, "Disponibilidade da Criança!A2:E75");

    for i = 1:length(dispExcel[:,1])
       crianca = dispExcel[i,1]
       dia = dispExcel[i,2];
       peri = dispExcel[i,3];
       ini = dispExcel[i,4];
       fim = dispExcel[i,5];
       #Sé está disponível no período da manhã
        if peri == "Manhã"
           if dia == "Todos"
               for p in P
                   if horarioDoPeriodo[p,peri]
                       disponibilidade[crianca,p] = 1;
                   end
               end
           else
               n = mapDias[dia]
               for p in ((n-1)*19+1):(n*19-10)
                   disponibilidade[crianca,p] = 1;
               end
           end
```

```
        #Se está disponível no período da tarde
        elseif peri == "Tarde"
            if dia == "Todos"
                for p in P
                    if horarioDoPeriodo[p,peri]
                        disponibilidade[crianca,p] = 1;
                    end
                end
            else
                n = mapDias[dia]
                for p in ((n-1)*19+10):n*19
                    disponibilidade[crianca,p] = 1;
                end
            end
        #Se está disponível em ambos os períodos
         elseif peri == "Ambos"
            if dia == "Todos"
                for p in P
                    disponibilidade[crianca,p] = 1;
                end
            else
                n = mapDias[dia]
                for p in ((n-1)*19+1):n*19
                    disponibilidade[crianca,p]
                end
            end
```

```
        #Se está disponível em horário específico
        elseif peri == "Horário"
            if dia == "Todos"
                for d in 1:5
                    for p in
                    ↪  (mapPeriInicial[ini]+(d-1)*19):(mapPeriInicial[fim]+(d-1)*19)
                        disponibilidade[crianca,p] = 1;
                    end
                end
            else
                d = mapDias[dia];
                for p in (mapPeriInicial[ini]+(d-1)*19):(mapPeriInicial[fim]+(d-1)*19)
                    disponibilidade[crianca,p] = 1;
                end
            end
        end
    end

    return disponibilidade;
end
```

C - Códigos utilizados em ambos os modelos

C.1 Cálculo de demanda das crianças

```
using ExcelReaders, JuMP

function getDemanda(C::Array{Any,2},E::Array{String,1})

    f = openxl(joinpath(@__DIR__,"modelo-de-dados-ccp.xlsx"))
    demanda = JuMP.Containers.DenseAxisArray{Int64}(undef, C, E);
    demandaExcel = readxl(f, "Atendimento Regular!A2:C277");
    fill!(demanda, 0);
    for i = 1:length(demandaExcel[:,1])
        crianca = demandaExcel[i,1]
        especialidade = demandaExcel[i,2];
        demanda[crianca,especialidade] = demandaExcel[i,3]
    end

    return demanda;
end;
```