

Pontifícia Universidade Católica de Goiás
Pró-Reitoria de Pós-Graduação e Pesquisa
Programa de Pós-Graduação Stricto Sensu em Engenharia de Produção e
Sistemas

MELHORIAS PARA O PROBLEMA DE DESIGNAÇÃO DE SALAS DE AULA DA PUC GOIÁS

Davi Taveira Alencar Alarcão

Goiânia
2015

Pontifícia Universidade Católica de Goiás
Pró-Reitoria de Pós-Graduação e Pesquisa
Programa de Pós-Graduação Stricto Sensu em Engenharia de Produção e
Sistemas

MELHORIAS PARA O PROBLEMA DE DESIGNAÇÃO DE SALAS DE AULA DA PUC GOIÁS

Davi Taveira Alencar Alarcão

Dissertação apresentada ao Programa de Pós-Graduação Stricto Sensu em Engenharia de Produção e Sistemas da Pontifícia Universidade Católica de Goiás, como parte dos requisitos para obtenção do título de Mestre em Engenharia de Produção e Sistemas.

Orientador: Professor Marco Antonio F. Menezes, Dr.

GOIÂNIA – GO
Fevereiro – 2015

Dados Internacionais de Catalogação da Publicação (CIP)
(Sistema de Bibliotecas PUC Goiás)

Alarcão, Davi Taveira Alencar.

A321m Melhorias para o problema de designação de salas de aula da PUC Goiás [manuscrito] / Davi Taveira Alencar Alarcão. – 2015. 85 f.; 30 cm.

Dissertação (mestrado) – Pontifícia Universidade Católica de Goiás, MEPROS / Programa de Pós-Graduação em Engenharia de Produção e Sistemas, Goiânia, 2015.

“Orientador: Prof. Dr. Marco Antonio Figueiredo Menezes”.

1. Programação linear. 2. Algoritmos. 3. Otimização combinatória. I. Menezes, Marco Antonio Figueiredo. II. Título.

CDU: 519.852(043)

*Dedico este trabalho à minha
esposa Amanda Vilela Carvalho Alarcão e
a meus filhos João Pedro Vilela Alarcão e
Anna Cecília Vilela Alarcão.*

AGRADECIMENTOS

Gostaria de agradecer, acima de tudo, A Deus, por ter permitido vivenciar as minhas derrotas. Foram elas que me fizeram crescer e acreditar que eu sou, sim, capaz.

Agradeço a você, Amanda Vilela Carvalho Alarcão, pela esposa que é. Obrigado por me ensinar tanto. Obrigado também por ter me dado filhos tão lindos.

Ao meu orientador, professor Marco Antonio Figueiredo Menezes, por ser essa grande pessoa. Obrigado pela paciência, por comemorar e acreditar em mim e nos resultados que conquistamos. Aprendi muito ao seu lado.

Ao professor Ivon Canedo, pelo grande apoio na busca pelo conhecimento.

À professora Maria José Pereira Dantas, por me ensinar, de forma tão tranquila, como lidar com as dificuldades que tive durante o programa de mestrado e, também, pela sua contribuição por ter participado da minha banca de qualificação.

À minha mãe, Neide Honorata Alencar Alarcão, e ao meu irmão, Daniel Taveira Alencar Alarcão, que continuam sendo o meu porto seguro nos momentos de turbulência.

Ao meu pai, João Batista Alarcão de Moraes, por ter me dado condições de ser o que sou hoje. Sei que o senhor pode ver mais essa conquista.

À minha tia Silvani, por ser essa pessoa tão especial na vida da minha família. A senhora é um presente de Deus. Obrigado por tudo mesmo.

À minha tia Mirna, pelo incentivo. Nunca me esquecerei.

Ao meu sogro, Pedro Carvalho Gomes, pelas palavras de conforto e de confiança.

À minha sogra, Soleni Vilela de Oliveira Carvalho, por todos os momentos de alegria vivenciados junto com a minha família.

E a todos que contribuíram com mais essa vitória em minha vida.

*“ A tarefa não é tanto ver aquilo que ninguém viu,
mas pensar o que ninguém ainda pensou
sobre aquilo que todo mundo vê.”
(Arthur Schopenhauer)*

MELHORIAS PARA O PROBLEMA DE DESIGNAÇÃO DE SALAS DE AULA DA PUC GOIÁS

Davi Taveira Alencar Alarcão

Esta Dissertação julgada adequada para obtenção do título de Mestre em Engenharia de Produção e Sistemas, e aprovada em sua forma final pelo Programa de Pós-graduação em Engenharia de Produção e Sistemas da Pontifícia Universidade Católica de Goiás em Fevereiro de 2015.

Prof. Ricardo Luiz Machado, Dr.
Coordenador do Programa de Pós-Graduação em
Engenharia de Produção e Sistemas

Banca Examinadora:

Prof. Marco Antonio Figueiredo Menezes, Dr.
Pontifícia Universidade Católica de Goiás
Orientador

Prof. Carlos Alberto de Jesus Martinhon, Dr.
Universidade Federal Fluminense

Prof. Clarimar José Coelho, Dr.
Pontifícia Universidade Católica de Goiás

GOIÂNIA – GO

Fevereiro – 2015

RESUMO

O problema de designação de salas de aula em Universidades consiste em distribuir turmas programadas para as devidas salas, respeitando os requisitos estabelecidos em cada situação. O objetivo deste trabalho é o de melhorar o processo de alocação de salas de aula da PUC Goiás. Os testes foram realizados com dados reais da PUC Goiás para um quantitativo de 5116 turmas em 312 salas de aula. Como resultados, resolvemos o problema em aproximadamente 34 minutos e comparamos a qualidade da solução tanto com a designação manual usualmente realizada pela Instituição, a qual leva um mês e meio, quanto com os resultados encontrados em Ribeiro (2012).

Palavras-chave: problema de designação de salas de aula, otimização linear, modelagem.

ABSTRACT

The classroom assignment problem at universities consist in distributing classes scheduled for the appropriate rooms, respecting the requirements in each situation. The objective of this work is to improve the process of allocation of classroom PUC Goiás. The tests were performed with real data from the PUC Goiás for a quantitative of 5116 classes into 312 classrooms. As a result, we solved the problem in approximately 34 minutes and the solution quality was compared both with manual designation usually applied by the institution, which takes a month and a half, as with the results found in Ribeiro (2012).

Keywords: classroom assignment problem, linear optimization, modeling.

LISTA DE TABELAS

Tabela 2.1 – Salas de preleção – 2012.....	12
Tabela 2.2 – Relação de cursos de graduação por UAA.....	14
Tabela 2.3 – Tipos de aulas existentes na Universidade.....	15
Tabela 2.4 – Características da turma com designação de sala.....	16
Tabela 2.5 – Restrições informadas pela CPAC.....	21
Tabela 2.6 – Relação custo, turma e sala para R_3	22
Tabela 2.7 – Pesos para deslocamento dos cursos em relação às áreas e blocos.....	24
Tabela 3.1 – Passos para o processo de alocação.....	28
Tabela 3.2 – Passos para a limpeza do banco de dados.....	34
Tabela 3.3 – Eficiência na alocação de turmas às áreas de origem.....	35
Tabela 3.4 – Sequência da nova ordenação de turmas.....	37
Tabela 4.1 – Eficiência na alocação de turmas às áreas de origem.....	39
Tabela 4.2 – Eficiência, por área, na alocação de turmas às áreas de origem.....	41
Tabela 4.3 – Ocupação das 31 salas de preleção da área III.....	43
Tabela 4.4 – Ciência da Computação – Matutino – 1º Período.....	46
Tabela 4.5 – Comparativo de tempo para alocar as turmas da PUC Goiás.....	49

LISTA DE FIGURAS

Figura 1.1 – Matriz de custos.....	9
Figura 1.2 – Grafo bipartido valorado.....	9
Figura 3.1 – Menu principal do SAPA.....	27
Figura 3.2 – Processo de cópia do Banco de Dados do SAPA.....	27
Figura 3.3 – Saída fornecida pelo algoritmo de alocação.....	28
Figura 3.4 – Manobra para acesso a um único Banco de Dados.....	29
Figura 3.5 – Visão da chamada da rotina implementada no processo de alocação....	30
Figura 3.6 – Relatório com os ajustes finais.....	31
Figura 3.7 – Interação entre os servidores <i>web</i> e a rede da PUC Goiás.....	32
Figura 3.8 – Mensagem de controle de acesso à rotina de designação.....	33
Figura 3.9 – Arquivo turmas.data.....	35
Figura 3.10 – Ordenando o arquivo turmas.data.....	36
Figura 4.1 – Tempo gasto na alocação (em minutos).....	38
Figura 4.2 – Ocupação de salas das áreas I e II.....	40
Figura 4.3 – Remanejamento da área II por baixa capacidade de sala.....	42
Figura 4.4 – Ocupação de salas das áreas III e IV.....	42
Figura 4.5 – Remanejamento da área III por indisponibilidade de sala.....	44
Figura 4.6 – Remanejamento da área III por baixa capacidade de sala.....	45
Figura 4.7 – Ocupação de salas das áreas VI e IX.....	45
Figura 4.8 – Comparação dos resultados de eficiência na alocação.....	47
Figura 4.9 – Remanejamento de turmas da área II.....	48
Figura 4.10 – Remanejamento de turmas da área III.....	49

LISTA DE ABREVIATURAS E SIGLAS

ADM	Unidade Acadêmica Administrativa de Administração
ARQ	Unidade Acadêmica Administrativa de Arquitetura e Urbanismo e Design
BIO	Unidade Acadêmica Administrativa de Biologia
CAER	Coordenação do Curso de Ciências Aeronáuticas
CBB	Unidade Acadêmica Administrativa de Biomedicina e Farmácia
CMP	Unidade Acadêmica Administrativa de Computação
CON	Unidade Acadêmica Administrativa de Ciências Contábeis
COS	Unidade Acadêmica Administrativa de Comunicação Social
CPAC	Coordenação de Programação Acadêmica
ECO	Unidade Acadêmica Administrativa de Economia
EDU	Unidade Acadêmica Administrativa de Educação
EFI	Unidade Acadêmica Administrativa de Educação Física
ENF	Unidade Acad. Administrativa de Enfermagem, Nutrição e Fisioterapia
ENG	Unidade Acadêmica Administrativa de Engenharias
FIT	Unidade Acadêmica Administrativa de Filosofia
FONO	Unidade Acadêmica Administrativa de Fonoaudiologia
HGS	Unidade Acadêmica Administrativa de História e Geografia
HySST	<i>Hyper-heuristic Search Strategies and Timetabling</i>
IGPA	Instituto Goiano de Pré-História e Antropologia
ITC	<i>International Timetabling Competition</i>
ITS	Instituto do Trópico Sub-Úmido
JUR	Unidade Acadêmica Administrativa de Ciências Jurídicas
LET	Unidade Acadêmica Administrativa de Letras
MAF	Unidade Acadêmica Administrativa de Matemática, Física e Química
MED	Unidade Acadêmica Administrativa de Medicina
MISTA	<i>Multidisciplinary International Scheduling Conference: Theory & Application</i>
NPC	NP-Completo
P	Polinomial
PA	Programação Acadêmica
PATAT	Practice and Theory of Automated Timetabling

PDS	Problema de Designação de Salas
PHP	<i>Hypertext Preprocessor</i>
PPL	Problema de Programação Linear
PL	Programação Linear
PLI	Programação Linear Inteira
PROGRAD	Pró-Reitoria de Graduação
PSI	Unidade Acadêmica Administrativa de Psicologia
PUC	Pontifícia Universidade Católica
RAM	<i>Random Access Memory</i>
SAPA	<i>Software</i> de Apoio à Programação Acadêmica
SATS	Sistema de Alocação Turmas Sala
SER	Unidade Acadêmica Administrativa de Serviço Social
SGA	Sistema de Gestão Acadêmica
TCC	Trabalho de Conclusão de Curso
UAA	Unidade Acadêmica Administrativa
ZOO	Unidade Acadêmica Administrativa de Zootecnia

SUMÁRIO

INTRODUÇÃO.....	1
CAPÍTULO I – PRELIMINARES.....	3
1.1 – PROGRAMAÇÃO LINEAR INTEIRA.....	3
1.2 – GRAFOS E O ALGORITMO HÚNGARO.....	4
1.2.1 – Grafo bipartido.....	5
1.2.2 – Emparelhamento.....	6
1.2.3 – O algoritmo Húngaro.....	8
CAPÍTULO II – FORMULAÇÃO EM MODELAGEM.....	12
2.1 - DISTRIBUIÇÃO DE ESPAÇO FÍSICO NA PUC GOIÁS.....	12
2.1.1 – A PUC Goiás em 2012.....	12
2.1.2 – A PUC Goiás em 2014.....	18
2.2 - PROBLEMA DE DESIGNAÇÃO DE SALAS DE AULA.....	18
2.2.1 – Pesquisas.....	19
2.2.2 – Conferências.....	20
2.3 - O MODELO.....	21
CAPÍTULO III – IMPLEMENTAÇÕES.....	26
3.1 – <i>SOFTWARE</i> DE APOIO À PROGRAMAÇÃO ACADÊMICA.....	26
3.2 – PROCESSO DE ALOCAÇÃO.....	27
3.3 – MELHORIAS IMPLEMENTADAS NO PROJETO.....	29
3.3.1 – Acesso às rotinas de designação com o SAPA.....	30
3.3.2 – Visualização dos resultados computacionais.....	31
3.3.3 – Gerenciando o acesso à rotina de designação de salas de aula.....	32
3.3.4 – Limpeza dos registros da base de dados do SAPA.....	33
3.3.5 – Ajuste na ordem de processamento das turmas, por área, da PUC Goiás.....	34
CAPÍTULO IV – RESULTADOS E COMPARAÇÕES.....	38
4.1 – RESULTADOS.....	38

4.2 – COMPARAÇÃO DOS RESULTADOS.....	46
4.2.1 – Eficiência na alocação de turmas às suas áreas de origem.....	47
4.2.2 – Remanejamentos de turmas.....	48
4.2.3 – Tempo de alocação.....	49
4.3 – ANÁLISE DOS RESULTADOS.....	50
CAPÍTULO V – CONSIDERAÇÕES FINAIS.....	51
REFERÊNCIAS BIBLIOGRÁFICAS.....	53
ANEXO I.....	57
ANEXO II.....	60
ANEXO III.....	61
ANEXO IV.....	62
ANEXO V.....	67
ANEXO VI.....	68

INTRODUÇÃO

Timetabling tem sido, por várias décadas, um problema desafiador e importante dentro da pesquisa operacional. A definição, feita por Wren (1996), é apresentada a seguir:

“*Timetabling* é a designação, sujeito a restrições, na entrega de recursos a objetos que estão sendo alocados em um determinado espaço de tempo, de tal modo que se consiga satisfazer, da melhor maneira possível, um conjunto de objetivos desejáveis.”

Esta definição a respeito de *Timetabling* faz com que ele consiga cobrir diversos problemas do mundo real, incluindo *Employee Timetabling*, *Rostering Problems*, *Sports Timetabling* e *Educational Timetabling*. Em *Educational Timetabling* existem diferentes problemas abordados por esta área, sendo eles, *University Course Timetabling*, *High School Timetabling*, *Examination Timetabling* e *Student Sectioning*. Estas nomenclaturas podem ser encontradas em Kristiansen e Stidsen (2013). Dentro desse subconjunto de problemas, trabalharemos com os relacionados com *University Course Timetabling*.

Seguindo a nomenclatura de Schaerf (1999), e Carter e Laporte (1998), resolveremos o problema de designação de salas de aula (*classroom assignment problem*) que consiste em encontrar, se possível, uma sala de aula aceitável para cada turma nos dias e horários especificados. Em virtude deste problema pertencer à classe de problemas NPC, conforme Carter e Tovey (1992), resolveremos este problema como um problema de otimização horário por horário e com requisitos não essenciais na função objetivo, o qual é um problema da classe P, também conforme Carter e Tovey (1992).

Os problemas que envolvem *University Course Timetabling* podem ser abordados de duas formas gerais, denominadas, mais recentemente: *Curriculum-based University Course Timetabling* e *Post Enrollment-based Course Timetabling* (veja Kristiansen e Stidsen (2013)).

É importante observar que, recentemente (Elloumi ET AL. (2014)), o termo *classroom assignment problem* foi utilizado para resolver um problema de designação de salas de aula para o problema da grade de horários para exames, provas (*examination timetabling problem*).

Na PUC Goiás, o problema de designação de salas de aula teve início com Neves (2010), quem propôs e testou o algoritmo Húngaro, com dados aleatórios, apenas para o Departamento de Computação, demonstrando uma eficiência no tempo de execução igual a 3 segundos (para 685 turmas e 34 salas de preleção). Neste mesmo momento, iniciou-se o desenvolvimento do *Software de Apoio à Programação Acadêmica (SAPA)*, do ponto de vista

operacional, tendo como foco a alocação de turmas em salas de aula e, em sintonia, com o modelo matemático desenvolvido a partir das informações dos requisitos essenciais e não essenciais solicitados pela Coordenação de Programação Acadêmica (CPAC). Silva (2011), com o intuito de resolver o problema de alocação de salas de aula para o Centro Técnico e Científico, também implementa o algoritmo Húngaro com dados aleatórios. O tempo de execução não ultrapassou 5 segundos (número de turmas igual a 1617 e o número de salas igual a 79). Em ambos os trabalhos a lista de pesos utilizada foi aleatória. Em Ribeiro (2012), os testes foram realizados com dados reais da PUC Goiás (referentes ao 1º semestre de 2012) para um quantitativo de 5116 turmas em 313 salas de aula divididas em 4 campi, 7 áreas e 20 blocos. O problema foi resolvido em, aproximadamente, 40 minutos. O tempo de execução foi corrigido aqui por causa da detecção de inúmeras duplicações nos registros das turmas do Departamento de Computação da PUC Goiás. O tempo na base de dados anterior foi mantido. Daí, fez-se uma comparação com a designação manual usualmente realizada pela Instituição, a qual leva um mês e meio para ser gerada e, também, fez-se uma comparação com a qualidade da solução, em que Ribeiro (2012) alcança 74% enquanto que a CPAC alcança 88%. A importância deste trabalho consiste em mostrar a possibilidade de execução da alocação, horário por horário, para toda a PUC Goiás e, também, a partir dos resultados apresentados, repensar em como melhorar a qualidade da solução.

O objetivo deste trabalho é o de melhorar o processo de alocação de salas de aula da PUC Goiás para um quantitativo de 5116 turmas em 312 salas de aula. Como resultados, resolvemos o problema em, aproximadamente, 34 minutos e comparamos a qualidade da solução tanto com a designação manual usualmente realizada pela Instituição quanto com os resultados encontrados em Ribeiro (2012).

No capítulo 1, apresentaremos os conceitos a respeito da programação linear inteira, grafos e sobre o algoritmo Húngaro. No capítulo 2, apresentaremos a revisão bibliográfica sobre o problema de designação de salas de aula, sobre a PUC Goiás e, ao final, a respeito do modelo matemático. No capítulo 3, discutiremos as implementações desenvolvidas. No capítulo 4, apresentaremos os resultados alcançados e a comparação destes resultados com os alcançados pela Coordenação de Programação Acadêmica da PUC Goiás e por Ribeiro (2012). No último capítulo, apresentaremos as nossas considerações finais.

CAPÍTULO I – PRELIMINARES

Neste capítulo faremos uma introdução necessária, mas breve, a duas disciplinas de otimização, cujos modelos definem problemas de designação de salas de aula. Sugerimos os livros Foulds (1984), Boaventura e Jurkiewicz (2009) e Maculan e Fampa (2006).

1.1 – Programação linear inteira

Um problema de programação linear inteira é um problema de programação linear onde os valores das variáveis de decisão são restritos a só admitirem valores inteiros.

Considere os números inteiros m e n tais que $n > m > 0$. Dados uma matriz numérica com coeficientes reais $A, m \times n$, e vetores $b \in \mathbb{R}^m$ e $c \in \mathbb{R}^n$, uma formulação para o problema de programação linear inteira é o seguinte problema de otimização:

$$\begin{aligned}
 (PLI) \quad & \text{minimizar} && c^T x \\
 & \text{sujeito a:} && Ax = b \\
 & && x \geq 0 \\
 & && x \in \mathbb{Z}^n.
 \end{aligned}$$

Uma formulação para o problema de programação linear inteira 0-1 (binário) é o seguinte problema de otimização:

$$\begin{aligned}
 (PLB) \quad & \text{minimizar} && c^T x \\
 & \text{sujeito a:} && Ax = b \\
 & && x \in \{0,1\}^n.
 \end{aligned}$$

Seguem-se algumas definições associadas a ambos os problemas (PLI) e (PLB).

Definição 1.1 Considere os problemas (PLI) e (PLB).

(a) A função linear $x \mapsto c^T x$ é chamada função objetivo.

(b) Conjunto viável é o conjunto

$$I = \{x \in \mathbb{Z}^n; Ax = b, x \geq 0\}$$

ou

$$B = \{x \in \{0,1\}^n; Ax = b\}.$$

(c) Um ponto $x \in I$ ou $x \in B$ é denominado ponto viável.

(d) Os conjuntos

$$P^* = \{x^* \in I; c^T x^* \leq c^T x, \text{ para todo } x \in I\}$$

e

$$B^* = \{x^* \in B; c^T x^* \leq c^T x, \text{ para todo } x \in B\}$$

são chamados conjuntos de soluções ótimas.

(e) Os problemas (PLI) e (PLB) são chamados de problema inviável quando I ou B são vazios, respectivamente.

(f) O problema (PLI) chama-se problema ilimitado quando existe uma sequência (x^k) tal que $x^k \in P$ e $c^T x^k \rightarrow -\infty$, quando $k \rightarrow \infty$.

A proposição a seguir mostra que o problema (PLI) pode ser reduzido ao problema (PLB), conforme SALKIN (1975).

Proposição 1.2 *Suponha que no problema (PLI) cada $x_j \leq u_j$, com $u_j > 0$, $j = 1, 2, \dots, n$. Então, o problema (PLI) é equivalente ao problema (PLB).*

Podemos olhar um problema de programação linear inteira como um problema em grafos. A seguir faremos uma introdução ao estudo de grafos com o intuito de apresentar o algoritmo Húngaro.

1.2 – Grafos e o algoritmo Húngaro

Um grafo G é definido por um par $G = (V, E)$ que consiste em V , um conjunto finito e não vazio de vértices, e E , um conjunto de arestas. Cada aresta tem um ou dois vértices associados a ela, chamados de suas extremidades. Dizemos que uma aresta liga ou conecta suas extremidades. Um grafo com apenas um vértice é dito trivial.

Considere um grafo $G = (V, E)$ e uma aresta $e = (v_1, v_2) \in E$. A aresta e é dita incidente sobre v_1 (e v_2). Os vértices v_1 e v_2 em V são chamados adjacentes ou vizinhos. O número de arestas incidentes sobre um vértice v_1 de G é chamado o grau desse vértice.

Seguem-se algumas definições associadas a um grafo G .

Definição 1.3 Considere um grafo $G = (V, E)$.

(a) Uma sequência de vértices $w = v_1, v_2, \dots, v_k$, $k \geq 1$, de forma que $(v_j, v_{j+1}) \in E$ para $j = 1, \dots, k-1$ é dita um passeio em G .

(b) Um passeio $w = v_1, v_2, \dots, v_k$ é dito fechado quando $k > 1$ e $v_k = v_1$.

(c) Um passeio sem vértices repetidos é chamado caminho em G .

(d) Um passeio fechado sem vértices repetidos que não o primeiro e o último é chamado circuito ou ciclo em G .

(e) O grafo G é dito completo quando, para quaisquer $v_1 \in V$ e $v_2 \in V$, (v_1, v_2) é uma aresta em G .

O conjunto de vértices adjacentes a um dado vértice v , em um grafo G , é chamado a vizinhança de v em G e é denotado por $N_G(v)$. Analogamente, denota-se por $N_G(T)$ a vizinhança de um conjunto T de vértices no grafo G , isto é, o conjunto de vértices de G adjacentes a, pelo menos, um vértice de T .

Um grafo $H = (W, F)$ é dito um subgrafo de um grafo $G = (V, E)$ caso $W \subseteq V$ e $F \subseteq E$, e é denotado por $H \subseteq G$. Um subgrafo $H \subseteq G$ é dito gerador quando H contém todos os vértices do grafo G .

Um grafo direcionado, ou digrafo, é um grafo com direções atribuídas para suas arestas. Formalmente, um digrafo D é um par $D = (V, A)$, onde V é um conjunto de vértices finito e não vazio e A é um conjunto de pares ordenados de vértices, isto é, $A \subseteq V \times V$.

1.2.1 – Grafo bipartido

Suponha que $G = (V, E)$ é um grafo que tem a seguinte propriedade: o conjunto de vértices V pode ser particionado em dois conjuntos, X e Y , e cada aresta em E é incidente sobre um vértice em X e um vértice em Y . Então G é chamado de grafo bipartido. Nem todo grafo pode ser particionado dessa forma. A condição necessária e suficiente para isso é a seguinte.

Proposição 1.4 Um grafo é bipartido se, e somente se, não possui nenhum ciclo de comprimento ímpar.

Dizemos grafo bipartido completo quando o grafo é bipartido com bipartição (X,Y) tal que, cada vértice de X é ligada a todo vértice de Y e vice-versa.

1.2.2 – Emparelhamento

Um emparelhamento M de um grafo $G = (V,E)$ é um subconjunto de arestas de G com a propriedade de que duas arestas quaisquer de M não compartilham o mesmo vértice. Um emparelhamento em G com o maior número possível de arestas é chamado de emparelhamento máximo. Neste caso, dizemos também que M é emparelhamento de cardinalidade máxima. Um emparelhamento é perfeito quando cada vértice $v \in V$ é tal que alguma aresta de M é incidente a v .

Considere um grafo G juntamente com um emparelhamento fixo M de G . Arestas em M são chamadas arestas cobertas por M ou emparelhadas, enquanto que as arestas restantes são ditas livres. Quando (u,v) é uma aresta coberta por M dizemos que u é par de v e u e v são ditos cobertos, enquanto que os vértices não cobertos são ditos descobertos. Um caminho $P = u_1, u_2, \dots, u_k$ é chamado alternante quando as arestas $u_1u_2, u_3u_4, \dots, u_{2j-1}u_{2j}$ são livres, enquanto que as arestas restantes $u_2u_3, u_4u_5, \dots, u_{2j}u_{2j+1}$ são cobertas. Um caminho alternante $P = u_1, u_2, \dots, u_k$ é chamado aumentante quando ambos u_1 e u_k forem vértices descobertos.

Segundo Figueiredo e Szwarcfiter (1999), pode-se caracterizar quatro tipos de problemas de emparelhamento em grafos, a saber:

- *Emparelhamento de cardinalidade máxima em grafos quaisquer* - Nesse problema, é dado um grafo $G = (V,E)$ e o objetivo é encontrar um emparelhamento máximo M de G , ou seja, com número máximo de arestas.
- *Emparelhamento de cardinalidade máxima em grafos bipartidos* - Esse é um caso particular do problema anterior, onde é dado um grafo bipartido G e o objetivo é encontrar um emparelhamento máximo M de G .
- *Emparelhamento de peso máximo em grafos quaisquer* - Uma outra versão do problema de emparelhamento é aquele em que é dado, além do grafo $G = (V,E)$, um número $w_{ij} \geq 0$ para cada aresta $(v_i, v_j) \in E$, chamado o peso de (v_i, v_j) . Assim, o

objetivo é encontrar um emparelhamento no qual a soma dos pesos das arestas é máxima.

- *Emparelhamento de peso máximo em grafos bipartidos* - Um caso particular do problema anterior, onde é dado um grafo bipartido G e um número $w_{ij} \geq 0$ associado a cada aresta $(v_i, v_j) \in E$, e o objetivo é encontrar um emparelhamento no qual a soma dos pesos das arestas é máxima.

Os próximos resultados constituem a base teórica para os algoritmos de emparelhamento e são fundamentais para a elaboração de um algoritmo eficiente para determinar emparelhamentos máximos. O próximo teorema caracteriza a maximalidade de um emparelhamento M em termos da existência de caminhos aumentantes. O teorema 1.7 adiante caracteriza a existência de emparelhamentos perfeitos num grafo bipartido.

Lema 1.5 *Seja M um emparelhamento em um grafo G e suponha que G contém um caminho aumentante $p = v_0, v_1, \dots, v_{2m+1}$. Então,*

$$M' = (M - \{v_1, v_2, v_3, v_4, \dots, v_{2m-1}, v_{2m}\}) \cup \{p\}$$

é um emparelhamento de cardinalidade $|M|+1$.

Teorema 1.6 *Um emparelhamento M em um grafo G é máximo se, e somente se, não há caminho aumentante em G em relação a M .*

Dessa forma, um algoritmo natural para encontrar um emparelhamento de cardinalidade máxima, que decorre do lema 1.5 e teorema 1.6, é começar com um emparelhamento vazio e, repetidamente, aumentar a cardinalidade do emparelhamento corrente, através do uso sucessivo de caminhos aumentantes. Essa abordagem irá terminar com um emparelhamento de cardinalidade máxima, dado que a cardinalidade do emparelhamento máximo é finita e cada iteração aumenta de uma unidade a cardinalidade do emparelhamento.

Geralmente, no problema de emparelhamento de cardinalidade máxima em um grafo bipartido, o objetivo é encontrar um emparelhamento que cobre todos os vértices deste grafo. Dessa forma, o teorema abaixo descreve condições necessárias e suficientes para a existência de emparelhamentos do tipo desejado.

Teorema 1.7 *Seja dado um grafo bipartido G . Então G admite emparelhamento que cobre todo vértice de V se, e somente se, $|N_G(S)| \geq |S|$, para todo $S \subseteq V$.*

O problema de emparelhamento de peso máximo em grafos bipartidos também é conhecido como problema de designação, porque pode ser aplicado, em princípio, para calcular a melhor designação de tarefas para trabalhadores, assumindo que o valor w_{ij} produzido pelo i -ésimo trabalhador na j -ésima tarefa é conhecido. Esse problema pode ser descrito como um problema de programação linear sendo possível resolver esse problema utilizando algoritmos para problemas de programação linear (contínua), pois a solução ótima será também inteira e binária.

Conforme apresentado em Boaventura e Jurkiewicz (2009), dado um custo c_{ij} , $i, j = 1, 2, \dots, n$, uma formulação de programação linear (inteira) para o problema de emparelhamento de peso mínimo em grafo bipartido é:

$$\begin{aligned} & \text{minimizar} && \sum_{i,j} c_{ij} x_{ij} \\ & \text{sujeito a:} && \sum_{i=1}^n x_{ij} = 1, \quad j = 1, 2, \dots, n \\ & && \sum_{j=1}^n x_{ij} = 1, \quad i = 1, 2, \dots, n \\ & && x_{ij} \in \{0,1\}, \quad i = 1, 2, \dots, n \text{ e } j = 1, 2, \dots, n. \end{aligned}$$

1.2.3 – O algoritmo Húngaro

A seguir apresentaremos a ideia do algoritmo Húngaro aplicado a grafos bipartidos com pesos nas arestas. O objetivo será encontrar um emparelhamento de peso mínimo. O exemplo a seguir pode ser encontrado em Boaventura e Jurkiewicz (2009).

Considere o problema onde temos que alocar 3 turmas para 3 salas. Pelas características de cada turma e de cada sala são atribuídos custos de alocação. Estes custos são apresentados na Figura 1.1.

Turma→ Sala↓	S1	S2	S3
T1	3	5	6
T2	5	4	2
T3	2	3	4

Figura 1.1 – Matriz de custos.

Podemos perceber que, ao atribuir uma turma a cada sala, estamos tomando três elementos da matriz tais que:

- (a) cada elemento está em uma linha diferente;
- (b) cada elemento está em uma coluna diferente;
- (c) cada linha e cada coluna contém exatamente um elemento.

Uma solução com estas características é chamada uma solução viável. Queremos que o custo desta solução seja mínimo. Do ponto de vista da teoria dos grafos temos um grafo bipartido valorado e estamos procurando uma solução viável com custo mínimo. Veja Figura 1.2.

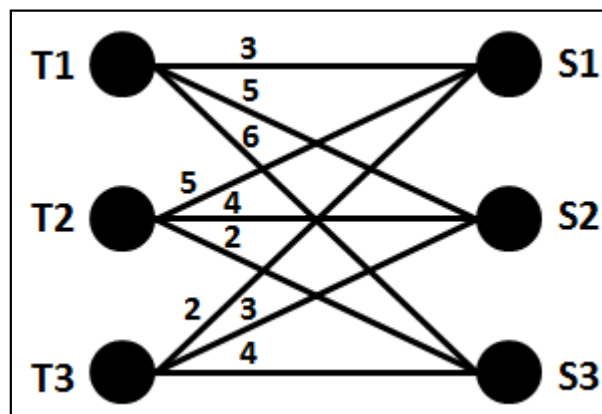


Figura 1.2 – Grafo bipartido valorado.

Podemos olhar também este problema como um problema de programação linear inteira 0-1 (binário), a saber:

$$\begin{aligned}
 \text{Minimizar} \quad & 3x_{11} + 5x_{12} + 6x_{13} + 5x_{21} + 4x_{22} + 2x_{23} + 2x_{31} + 3x_{32} + 4x_{33} \\
 \text{Sujeito a:} \quad & x_{11} + x_{12} + x_{13} = 1 \\
 & x_{21} + x_{22} + x_{23} = 1 \quad (\text{A cada sala corresponde apenas uma turma}) \\
 & x_{31} + x_{32} + x_{33} = 1 \\
 & x_{11} + x_{21} + x_{31} = 1 \\
 & x_{12} + x_{22} + x_{32} = 1 \quad (\text{A cada turma corresponde apenas uma sala}) \\
 & x_{13} + x_{23} + x_{33} = 1 \\
 & x_{ij} \in \{0,1\}, \quad i = 1,2,3 \text{ e } j = 1,2,3.
 \end{aligned}$$

Um exemplo simples de solução viável é $x_{11} = 1, x_{22} = 1, x_{33} = 1$ e as demais variáveis iguais a zero, com custo igual a $3 + 4 + 4 = 11$. O número de soluções viáveis é igual a $3! = 6$ e, por inspeção, podemos verificar que a solução de custo mínimo é $x_{11} = 1, x_{23} = 1, x_{32} = 1$ e as demais variáveis iguais a zero, com custo 8.

Se nossa matriz for de ordem maior, a resolução por inspeção torna-se inviável. Vamos desenvolver então uma ideia bem simples do algoritmo Húngaro.

Primeiramente, observamos que o valor de nossa solução não se altera se adicionarmos ou subtrairmos um mesmo valor de todos os elementos de uma linha (ou coluna). De fato, só um dos elementos afetados estará na solução mínima. A solução da nova matriz terá o seu valor diminuído no número subtraído de unidades, mas os elementos da solução serão os mesmos. Por exemplo, diminuindo 3 na primeira linha e colocando asterisco nos menores elementos das linhas, obtemos

$$\left| \begin{array}{ccc|c} 3 & 5 & 6 & -3 \\ 5 & 4 & 2 & \\ 2 & 3 & 4 & \end{array} \right. \rightarrow \left| \begin{array}{ccc|c} 0 & 2 & 3 & \\ 5 & 4 & 2 & \\ 2 & 3 & 4 & \end{array} \right. \rightarrow \left| \begin{array}{ccc|c} 0^* & 2 & 3 & \\ 5 & 4 & 2^* & \\ 2 & 3^* & 4 & \end{array} \right.$$

A solução é a mesma, isto é, a mesma permutação, mas o valor ficou diminuído em 3 unidades. Vamos completar o trabalho com as linhas e depois aplicar o mesmo princípio às colunas, mas não ao mesmo tempo. Usando esta ideia no nosso exemplo temos:

$$\left| \begin{array}{ccc|c} 3 & 5 & 6 & -3 \\ 5 & 4 & 2 & -2 \\ 2 & 3 & 4 & -2 \end{array} \right. \rightarrow \left| \begin{array}{ccc|c} 0 & 2 & 3 & \\ 3 & 2 & 0 & \\ 0 & 1 & 2 & \end{array} \right. \rightarrow \left| \begin{array}{ccc|c} 0^* & 1 & 3 & \\ 3 & 1 & 0^* & \\ 0 & 0^* & 1 & \end{array} \right.$$

-1

A solução $x_{11} = x_{23} = x_{32} = 1$ e $x_{12} = x_{13} = x_{21} = x_{22} = x_{31} = x_{33} = 0$, está agora bastante evidente, bastando procurar os zeros. Observemos que o custo 8 é dado pela soma dos valores subtraídos. O enunciado do algoritmo Húngaro pode ser encontrado em Kuhn (1955).

O próximo capítulo apresenta uma formulação para o problema de designação de salas de aula da PUC Goiás.

CAPÍTULO II – FORMULAÇÃO EM MODELAGEM

Iniciaremos o nosso propósito apresentando a distribuição de espaço físico da PUC Goiás, definindo o problema de designação de salas de aula e, ao final, o modelo matemático.

2.1 - Distribuição de espaço físico na PUC Goiás

A proposta deste trabalho foi a de analisar os dados da PUC Goiás referentes ao primeiro semestre de 2012. Desta forma, as informações apresentadas abaixo estão direcionadas para este período, em específico. Ao final deste tópico, teremos uma visão geral de como a Universidade se encontra atualmente.

2.1.1 – A PUC Goiás em 2012

A Universidade está dividida em quatro campi, sendo todos eles em Goiânia. Cada um destes campus possuem, o que chamamos de áreas. As áreas são divididas em blocos. Já os blocos abrigam as estruturas: administrativa, acadêmica, laboratórios, ateliers, espaços de convivência, salas de aula e de orientação entre outros.

A quantidade de salas de preleção presentes em cada área pode ser observada na Tabela 2.1. Estas informações são importantes para o estudo do problema em questão. A propósito, o Campus III não tem aulas de preleção e o Campus IV foi desativado na cidade de Ipameri/GO.

Tabela 2.1 - SALAS DE PRELEÇÃO – 2012.

CAMPUS	ÁREA	Salas
Campus I	Área I	67
Campus I	Área II	50
Campus I	Área III	31
Campus I	Área IV	49
Campus II	Área VI	39
Campus V	Área IX	76
TOTAL - SALAS DE PRELEÇÃO		312

Fonte: PUC em dados 2013.

A quantidade de alunos presente na PUC Goiás é de, aproximadamente, 30.000, conforme PUC em dados, 2013. Eles se dividem em cursos de graduação, pós-graduação e de extensão que, por sua vez, são gerenciados pelas Unidades Acadêmicas Administrativas (UAAs), denominados departamentos. Estas UAAs são responsáveis por organizar a Programação Acadêmica dos seus cursos e das disciplinas que são ofertadas às demais graduações da Universidade. Além disso, também tem a responsabilidade pela formulação das turmas e organização dos horários, garantindo, assim, o progresso do discente na grade curricular.

As turmas analisadas neste trabalho estão relacionadas com os cursos de graduação presentes na Tabela 2.2.

Tabela 2.2 - RELAÇÃO DE CURSOS DE GRADUAÇÃO POR UAA.

CURSO	UAA
Administração	ADM
Curso Superior de Tecnologia Em Agronegócios	ADM
Curso Superior de Tecnologia Em Gastronomia	ADM
Curso Superior de Tecnologia Em Eventos	ADM
Ciências Aeronáuticas	CAER
Ciências Contábeis	CON.
Ciências Econômicas	ECO
Serviço Social	SER
Pedagogia	EDU
Filosofia	FIT
Geografia	HGS
História	HGS
Relações Internacionais	HGS
Arqueologia	IGPA
Letras - Português	LET
Curso Superior de Tecnologia em Secretariado	LET
Design	ARQ
Arquitetura e Urbanismo	ARQ
Ciência da Computação	CMP
Engenharia de Computação	CMP
Curso Superior Tecnol. em Análise e Des. de Sistema	CMP
Engenharia Ambiental	ENG
Engenharia - Habilitação em Eng. Civil	ENG
Engenharia Elétrica	ENG
Engenharia de Produção	ENG
Engenharia de Controle e Automação - Mecatrônica	ENG
Licenciatura Plena em Física	MAF
Química	MAF
Engenharia - Hab. em Eng. de Alimentos	MAF
Matemática	MAF
Biologia	BIO
Enfermagem	ENF
Fisioterapia	ENF
Nutrição	ENF
Medicina	MED
Psicologia	PSI
Farmácia	CBB
Ciências Biológicas-Modalidade Médica	CBB
Fonoaudiologia	FONO
Educação Física	EFI
Curso Superior de Tecnologia em Gestão Ambiental	ITS
Zootecnia	ZOO
Comunicação Social - Jornalismo	COS
Comunicação Social - Publicidade E Propaganda	COS
Direito	JUR

A Pró-Reitoria de graduação possui uma coordenação responsável por gerenciar as programações destas Unidades denominada Coordenação de Programação Acadêmica (CPAC). Esta torna-se responsável não só por garantir o melhor aproveitamento em uma equação que envolve alunos, turmas, espaço físico e o número de vagas a serem ofertadas, mas também pela distribuição de carga horária docente.

Os critérios estabelecidos pela CPAC, na gestão das programações acadêmicas, possibilitam aos alunos se matricularem nas turmas, distribuírem as cargas horárias aos professores e ocuparem o mínimo de espaço da Universidade. Desta forma, uma turma fica exatamente o horário necessário dentro de uma sala de aula (oferecendo margem para a utilização da sala por outra turma), além disso, oferece a possibilidade do docente construir a sua grade de horários e, para o aluno, conseguir cumprir com o seu cronograma de aulas por período acadêmico.

São cinco os tipos de horários presentes em uma programação acadêmica. Alguns necessitam de reserva de sala de aula, já outros não, por serem disciplinas práticas ou de orientação e estágio (são realizadas em espaços físicos específicos). Existem alguns casos que são tratados separadamente (após a distribuição do espaço físico), como as disciplinas que necessitam de uma sala de aula, mesmo não sendo da categoria de preleção (tratados separadamente, após a distribuição do espaço físico) e os Trabalhos Finais de Curso (TCC).

Na Tabela 2.3, é possível observar os tipos de aulas existentes na Universidade.

Tabela 2.3 - Tipos de aulas existentes na Universidade.

SIGLA	Descrição
PRE	Disciplinas que necessitam de sala de aula no horário programado.
PRA	Disciplinas que podem acontecer em sala de aula e/ou fora da Universidade no horário programado.
ORI	Disciplinas de orientação individual ou coletiva. Utilizam espaços diferentes da sala de aula e raramente ocupam espaço físico definitivo.
LAB	Disciplinas que necessitam de espaços laboratoriais especificados e determinados pelas UAAs.
EST	Disciplinas que iniciam suas atividades dentro da Universidade, mas acontecem de fato em campos de estágio fora da Universidade.

O conceito de turma, a ser trabalhado no problema de designação de salas de aula na PUC Goiás, é a de um grupo de alunos e professores de uma disciplina de um curso que se localiza em uma área, campus, em um período do dia (matutino, vespertino ou noturno) com início e término do horário de aula, dia da semana, número de créditos e quantidade de hora-aula.

Uma sala de aula será especificada como sendo um espaço físico localizado em uma área, campus e em um bloco, com número da sala. É nela que ocorrerá os encontros para as aulas de preleção.

Para ficar mais claro, observe o exemplo apresentado na Tabela 2.4.

Tabela 2.4 – Características da turma com designação de sala.

Código	Disciplina	Turma	Dia	Horário	Tipo	Créditos	Área Bloco Sala	Período e turno	Grade	Curso
MAF1870	HISTÓRIA DA MATEMÁTICA	C01	SEG	1845~2015	PRE	2	02 D 406	7/N	2009 1	MATEMÁTICA

Neste caso, a turma C01, da disciplina História da Matemática, código MAF1870, pertencente ao 7º Período Noturno do Curso de Matemática, foi alocada na sala 02 D 406 (Área II, Bloco D, Sala número 406).

Cada UAA deverá realizar a sua programação de turmas para o próximo semestre. Para que o resultado seja satisfatório, os coordenadores devem compreender bem o perfil do curso e outros assuntos que o envolvam, tais como, o quantitativo de alunos por período.

Após a finalização dos trabalhos por parte das UAAs, a programação é encaminhada à CPAC. Por meio de relatórios estatísticos e históricos envolvendo as programações acadêmicas, sugere-se alterações, inserções e cortes de turmas. Ao final, o resultado é apresentado à Pro-Reitoria de Graduação. Somente com a programação aprovada, são liberadas as turmas para o procedimento de matrículas nas UAAs.

O processo de matrícula acontece no final do semestre letivo e início do próximo. Em um primeiro momento, os alunos regulares (vida acadêmica e financeira) realizam a pré-matrícula, mas observa-se uma grande quantidade de matrículas sendo efetuadas nas semanas iniciais do semestre.

Neste período, a CPAC começa a monitorar o comportamento das matrículas, verificando quais turmas serão bloqueadas e/ou canceladas. O resultado desta análise deverá

ser o equilíbrio entre os recursos existentes (professores, espaço-físico e financeiro) e as requisições (vagas para os alunos e preenchimento ótimo das turmas).

Em paralelo com esta atividade, acontece a distribuição de carga horária docente. Esta distribuição é realizada pela coordenação das UAAs junto com a coordenação de cada curso e o colegiado de professores. Nesta atividade não há intervenção da CPAC, pois a UAA acadêmica conhece a formação de cada docente e a ementa de cada disciplina.

Retomando para a análise da CPAC, tem-se a seguinte situação: as matrículas acontecendo, as turmas começando a serem preenchidas pelos alunos, os professores sendo remanejados para as turmas. Surge, então, a necessidade de se saber qual é a sala de aula responsável por atender a todos os requisitos essenciais para o bom andamento do curso.

Desta forma, inicia-se o entendimento de como esta etapa do processo é realizada atualmente. A alocação começa a ser feita já no período de férias. As salas são cadastradas para as turmas no SGA (Sistema de Gestão Acadêmica), sem que haja preocupação se a sua capacidade será satisfatória ou não.

Este processo de alocação de salas para as turmas, incluindo o remanejamento, via SGA, demora em torno de um mês. Cada curso, com sua programação por período, são direcionados para um conjunto de salas, acreditando assim que o quesito capacidade será atendido.

Como, no início do semestre, as turmas também recebem requisições de vagas dos alunos durante o processo de matrícula, aquelas que já foram alocadas em determinadas salas de aula podem receber mais alunos, fazendo com que a capacidade da sala seja inviabilizada.

Do mesmo modo, as turmas também podem perder alunos durante o processo de pré-matrícula e, desta forma, deve haver o remanejamento para uma sala com o quantitativo de alunos próximo da sua realidade atual. Portanto, o que se observa é um quantitativo de remanejamentos das turmas para o aproveitamento ótimo do espaço físico.

Oferecer um sistema de alocação de espaço físico que possa, em pouco tempo, designar salas para as turmas, respeitando todos os requisitos necessários para o bom desempenho tanto do discente quanto do docente, é mais do que necessário.

O sistema em questão deverá realizar a alocação na véspera do início das aulas por oferecer maior garantia de sucesso, pois irá alocar as turmas em salas com a devida capacidade respeitada, diminuindo a locomoção de professores e alunos e os remanejamentos.

A CPAC tem tentado, semestralmente, diminuir o número de remanejamentos. O grande problema é a existência do sistema de créditos nas grades curriculares. Não é fácil realizar uma previsão de ocupação das turmas. Sendo assim, visando garantir que o aluno

possa ter acesso à disciplina durante a etapa de matrícula, muitas vezes, é ofertada uma quantidade de vagas mais do que o necessário, dificultando a distribuição de espaço físico.

2.1.2 – A PUC Goiás em 2014

A respeito do espaço físico da PUC Goiás, um novo prédio será inaugurado em breve, com aproximadamente 40 salas de aula. Quanto ao quantitativo de alunos, são 26.000 somente na graduação. Contabilizando pós-graduação e extensão, continuamos com o aproximado de 30.000 alunos.

Atualmente, a Universidade se encontra em um período de transição do modelo de departamentos para o modelo de Escolas. Esta nova configuração visa agrupar, ainda mais, os cursos de natureza afim, seus alunos e corpo docente.

Esta reestruturação, epistemológica e física, favorecerá o controle dos processos internos da Universidade, tomando como foco, a formação delineada para os cursos da Escola. Desta forma, não há mais decisões isoladas em cada curso, tudo agora é conjunto.

Neste horizonte, encontram-se estudos de otimização, entrelaçamento de currículos e compartilhamento de espaços físicos (salas e laboratórios).

2.2 – Problema de designação de salas de aula

O problema que será estudado a seguir necessita do entendimento de alguns termos para que a visão do todo seja melhor compreendida. Estes termos são especificados por Carter e Laporte (1998), sendo eles: programa (a *program*), disciplina (a *course*) e turma (a *class* ou a *course section*).

A definição de programa consiste em um conjunto de disciplinas agrupadas por grade curricular, representando projetos pedagógicos. Sendo assim, cada aluno deverá cumprí-las para concluir sua formação. Disciplina significa uma ementa a ser cumprida geralmente em um semestre. Turma trata-se de subdivisões de uma disciplina em grupos distintos de estudantes, podendo ser ministradas por professores diferentes e, também, em dias, horários e salas diferentes.

A partir destas informações, é possível dizer que um problema de programação de horários de disciplinas (*course timetabling problem*), representa um problema de designação multi-dimensional em que alunos e professores são designados para disciplinas, turmas ou períodos com encontros realizados em salas de aula e em determinados horários.

Um problema de designação de salas de aula (*classroom assignment problem*) consiste em encontrar, se possível, uma sala de aula aceitável para cada turma nos dias e horários especificados. Neste tipo de problema, consideram-se as características de cada turma e de cada sala de aula. Em particular, o problema de designação de salas de aula para a PUC Goiás consiste em alocar turmas de preleção (código da disciplina, código da turma, nome do professor, início e término do horário de aula, nome do curso, campus, área, dia da semana, número de créditos, quantidade de hora-aula) nas salas disponíveis (número da sala, letra do bloco, área, campus), fixando seus respectivos horários diários.

De uma maneira geral para o problema de designação de salas de aula, a primeira hipótese é a de que as turmas já deverão estar alocadas em um determinado horário do dia. A segunda hipótese é a de que existam salas suficientes e de tamanho apropriado para acomodar todas as turmas em todos os períodos, se for necessário. Neste caso, vale lembrar que os professores já estarão designados para determinadas turmas com seus respectivos horários do dia.

Burke ET AL. (1997), apresentam dois tipos de restrições presentes nos problemas de designação de salas de aula. O primeiro é denominado requisitos essenciais (*hard constraint*) no qual, não satisfazê-los, inviabiliza o problema e, o segundo, é denominado requisitos não essenciais (*soft constraint*) no qual, não satisfazê-los, não inviabiliza o problema.

Segundo Carter e Tovey (1992), o problema de designação de salas de aula é da classe NPC (NP-Completo). Todavia, se pensarmos neste problema horário por horário e com requisitos não essenciais na função objetivo, temos um problema da classe P (Polinomial), porque passa a ser um problema de emparelhamento de peso mínimo em grafo bipartido.

2.2.1 – Pesquisas

A respeito das publicações existentes sobre programação de horários de disciplinas para Universidades (*University Course Timetabling*) pode-se destacar o trabalho desenvolvido por Carter e Laporte (1998) em que escrevem sobre o assunto analisando artigos publicados entre 1980 e 1998. Neste artigo em questão, informam sobre a presença de implementações práticas com dados reais de Escolas e Universidades. Discorrendo sobre as publicações que envolvem o assunto designação de salas de aula (*Classroom Assignment*) destacam-se três trabalhos: o de Glassey e Mizrach (1986), que resolve um modelo de programação linear por uma decomposição heurística; o de Gosselin e Truchon (1986), que resolve um modelo de programação linear inteira repetidamente com diferentes valores na função penalizada

minimizada; e o de Carter (1989), que resolve um modelo de programação inteira com relaxação lagrangiana.

McCollum (2007) fornece, até 2006, informações sobre o tema programação de horários de disciplinas para a Universidade (*University Course Timetabling*). O objetivo é motivar pesquisadores no preenchimento da lacuna existente entre a teoria e a prática, e constata que uma década depois ainda haviam poucos trabalhos práticos para a resolução de problemas de horários em Universidades.

Em Kristiansen e Stidsen (2013), na Tabela A.4 sobre *University Course Timetabling*, descrevem os autores, áreas de pesquisa (nomenclatura recente) e comentários de trabalhos de 2001 até 2013, porém, sem nenhuma publicação específica sobre o problema de designação de salas de aula.

Constantino ET AL. (2010) resolvem o problema de designação de salas de aula com dados reais para uma Universidade usando três algoritmos heurísticos: em particular, o primeiro e o mais eficiente, consiste na resolução sucessiva (horário por horário) que combina o método húngaro e o método do menor caminho aumentante proposto por Carpaneto e Toth (1987). Neste trabalho, o maior problema resolvido, em aproximadamente 40 minutos, é de 4016 turmas por 192 salas. Subramanian ET AL. (2011) resolvem o problema de designação de salas de aula para um Centro de Tecnologia em uma Universidade usando um algoritmo baseado em busca Tabu.

2.2.2 – Conferências

Algumas conferências se dedicam à arte da automatização de programação de horários (*Timetabling*), como é o caso da PATAT (*International Conference on the Practice and Theory of Automated Timetabling*) e da MISTA (*Multidisciplinary International Scheduling Conference: Theory & Application*). Ambas acontecem a cada dois anos.

Existe também a ITC (*International Timetabling Competition*). A PATAT é um dos patrocinadores. Elas já aconteceram três vezes e foram analisadas tanto em McCollum (2007) quanto em McCollum ET AL. (2010). Tanto as regras oficiais quanto a forma de escolha dos vencedores podem ser visualizadas no próprio *website* de cada uma das competições.

A primeira ITC foi realizada em 2003 e se baseou no assunto programação de horários de disciplinas para a Universidade (*University Course Timetabling*). O vencedor foi Kostuch (2004). Os resultados foram apresentados na PATAT 2004.

A segunda ITC aconteceu em 2007 e foi composta por 3 etapas. O vencedor da primeira e da terceira etapa foi Müller (2009). Já a segunda etapa foi vencida por Cambazart ET AL. (2008). Todos foram apresentados na PATAT 2008.

A terceira e, mais recente ITC, foi realizada em 2011 e também foi composta por 3 etapas. Ela teve como foco o problema programação de horários para escolas (*High School Timetabling*). O vencedor da primeira etapa foi o time HySST (*Hyper-heuristic Search Strategies and Timetabling*). Fonseca ET AL. (2012) venceram a segunda e a terceira etapa, conquistando o título. Todos foram apresentados na PATAT 2012.

2.3 – O modelo

Para a gestão do problema de designação de salas de aulas, a CPAC discorre a respeito de seis restrições importantes. A Tabela 2.5 apresenta as duas primeiras restrições sendo os requisitos essenciais (*hard constraint*) e as quatro últimas sendo os requisitos não essenciais (*soft constraint*).

Tabela 2.5 - Restrições informadas pela CPAC.

Requisitos	Descrição
R_1	Uma sala terá, no máximo, uma turma no mesmo horário.
R_2	Cada turma que necessitar de uma sala com destinação própria, deverá ser alocada para, exatamente, uma sala desse tipo, com sua capacidade mínima e máxima respeitada.
R_3	Cada turma poderá ter a necessidade de salas diferenciadas, tais como: volume de ruído razoável, temperatura agradável e recursos especiais (quadro branco, tela de projeção e luminosidade adequada).
R_4	Cada turma deverá ser alocada para uma sala próxima ao bloco de seu curso.
R_5	Turmas de mesmo curso deverão ser alocadas preferencialmente para salas no mesmo bloco na semana.
R_6	Turmas de mesmo período da grade curricular de um curso deverão ser alocadas preferencialmente para a mesma sala na semana.

Nesse caso, é definido x_{ij} , $i=1,2,\dots,m$ e $j=1,2,\dots,n$, como sendo as variáveis de decisão que se pretende encontrar, se existir, a saber:

$$x_{ij} = \begin{cases} 1, & \text{se a turma } i \text{ é designada para a sala } j, \\ 0, & \text{caso contrário.} \end{cases}$$

Cada sala receberá a alocação de, no máximo, uma turma no seu respectivo horário, ou seja, uma sala não poderá ter mais de uma turma no mesmo horário. Sendo assim, considere os horários k , $k = 1, 2, \dots, t$. Defina-se P_k como o conjunto que representa todas as turmas que se encontram no horário k . Sendo assim, deve-se impor:

$$\sum_{i \in P_k} x_{ij} \leq 1, \quad j = 1, 2, \dots, n, k = 1, 2, \dots, t.$$

Cada turma deverá ser alocada para exatamente uma sala, respeitando a sua capacidade. Assim, define-se um conjunto de salas S_i que respeitem a capacidade de cada turma (de preleção) i , $i = 1, 2, \dots, m$. Deve-se impor:

$$\sum_{j \in S_i} x_{ij} = 1, \quad i = 1, 2, \dots, m.$$

O objetivo é o de minimizar um custo para a designação que se chamará c_{ij} , $i = 1, 2, \dots, m$ e $j = 1, 2, \dots, n$, a saber:

$$\sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij}$$

Para entendimento do modelo, o custo c_{ij} deverá ser compreendido. Observe que, segundo o requisito não essencial R_3 , cada turma poderá ter a necessidade de salas diferenciadas. Sendo assim, a Tabela 2.6 apresenta a relação custo, turma e sala.

Tabela 2.6 - Relação custo, turma e sala para R_3 .

Item	Alocação	Custo
A	Turmas em salas cujo volume de ruído, naquele horário, é exagerado.	30
B	Turmas em salas cuja temperatura, naquele horário, é bastante elevada.	20
C	Turmas em salas que ofereçam recursos especiais (a disciplina exige condições especiais).	1
D	Turmas em salas que não ofereçam recursos especiais (disciplina exige condições especiais).	70
E	Turmas que não atendam aos itens A, B, C e D.	10

Segundo o requisito não essencial R_4 , deve-se alocar cada turma para uma sala próxima ao bloco do seu curso. Para que fosse possível a análise de proximidade entre o bloco e a sala alocada, a CPAC forneceu uma tabela de pesos com valores referentes ao deslocamento de cada curso em relação às áreas e blocos da Universidade, baseando-se na distância, em metros, na qual foi multiplicada por mil. Veja a Tabela 2.7.

Ainda na Tabela 2.7, é importante afirmar que os cursos oriundos da área V, considerando que esta área possui apenas cinco salas de preleção de pequena capacidade, são alocados preferencialmente para a área IV. Desta forma, eles tiveram os mesmos pesos utilizados para os cursos da área IV. Ressalta-se, ainda, que na área IV existe, tradicionalmente, a preferência de alguns cursos para determinados blocos. Sendo assim, houve o estabelecimento de pesos utilizando as distâncias internas entre os blocos.

Segundo o requisito não essencial R_5 , as turmas de mesmo curso devem ser alocadas em salas que estejam no mesmo bloco naquela semana. Esse requisito está incorporado em R_4 , tomando parâmetros iguais a 0 (zero) para atender a preferência de cursos por determinado bloco ou determinados blocos. Como exemplo, podemos citar uma turma com horários na segunda-feira e na quinta-feira. O sistema inicia o processo de alocação e define a sala 03 F 206 na segunda-feira. Logo em seguida, atribuirá o menor custo, zero (para esta sala), com o intuito de manter o mesmo local para a aula de quinta-feira.

Segundo o requisito não essencial R_6 , as turmas de mesmo período da grade curricular de um curso devem ser alocadas em uma mesma sala naquela semana. Isso sugere a resolução do problema, horário por horário, aproveitando a solução atual para a introdução de pesos no valor da função objetivo. O objetivo é o de atender aos requisitos nos próximos horários. Este procedimento é feito pelo sistema, após a primeira designação e assim sucessivamente, quando necessário.

Ao final, o custo é calculado pela soma dos pesos dos requisitos não essenciais R_3 , R_4 , R_5 e R_6 . Sendo assim, uma formulação para o problema de designação de salas de aula é dada pelo problema de programação linear inteira 0-1,

$$\begin{aligned}
 (PDS) \text{ minimizar } & \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij} \\
 \text{sujeito a: } & \sum_{i \in P_k} x_{ij} \leq 1, \quad j = 1, 2, \dots, n, k = 1, 2, \dots, t, \\
 & \sum_{j \in S_i} x_{ij} = 1, \quad i = 1, 2, \dots, m, \\
 & x_{ij} \in \{0, 1\}, \quad i = 1, 2, \dots, m \text{ e } j = 1, 2, \dots, n
 \end{aligned}$$

No próximo capítulo, o modelo (PDS) será resolvido horário por horário (k fixo) e com requisitos não essenciais na função objetivo, através do algoritmo Húngaro, onde discutiremos a respeito das implementações.

CAPÍTULO III – IMPLEMENTAÇÕES

Neste capítulo, as implementações apresentadas são as seguintes:

- a) Oferecer o acesso à rotina de designação de salas de aula da PUC Goiás pelo *Software* de Apoio à Programação Acadêmica (SAPA);
- b) Oferecer a visualização dos resultados computacionais obtidos após a execução da rotina de designação de salas de aula;
- c) Gerenciar o controle de acesso enquanto a rotina de designação de salas de aula estiver sendo executada;
- d) Realizar a limpeza dos registros da base de dados do SAPA (primeiro semestre de 2012);
- e) Realizar o ajuste na ordem de processamento das turmas, por área, da PUC Goiás.

É importante lembrar que o acesso à rotina, a visualização dos resultados computacionais e o gerenciamento do controle de acesso à rotina de designação são realizados via INTRANET, havendo apenas a necessidade de se possuir o navegador Internet Explorer 9.0 instalado na Máquina que for acessar o SAPA.

Todas as etapas relatadas foram realizadas em dois computadores, ambos configurados como servidores *web*. Instalamos o SAPA em um computador com processador Intel Core i3, com um clock de 2.27 GHz, 2 GB de memória RAM disponível e com Windows 7 Ultimate de 32 bits.

Os *softwares* e demais códigos foram instalados em outro computador, apresentando as seguintes configurações: processador AMD Dual Core, com um clock de 1.30 GHz, 2 GB de memória RAM disponível e com Linux Mint 17.

No decorrer do capítulo, utilizamos o termo processo para apresentar apenas a etapa responsável por designar salas para as turmas (em um determinado horário). O termo projeto significa todo o trabalho, ou seja, as implementações realizadas no SAPA (juntamente com o banco de dados) e no processo de alocação.

3.1 – *Software* de apoio à programação acadêmica

O SAPA é um *software* de apoio acadêmico desenvolvido pelo professor Ivon Canedo. Ele foi formulado utilizando a linguagem de programação PHP e apresenta rotinas em *JavaScript*. Possui

uma interface *web* que recebe os dados do usuário (geralmente um funcionário do departamento) e os salva em um banco de dados. Na Figura 3.1 é possível verificar o menu principal deste sistema.

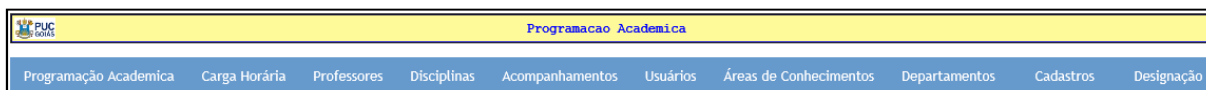


Figura 3.1 – Menu principal do SAPA.

Através dele, foram cadastradas as informações referentes ao primeiro semestre de 2012. Desta forma, o problema apresenta 1242 disciplinas dispostas em 43 horários diferentes durante a semana nos turnos matutino, vespertino e noturno, resultando um total de 5116 turmas que devem ser alocadas para 312 salas, divididas em 4 campi, 7 áreas e 20 blocos.

A partir dos dados cadastrados, é possível dar continuidade ao processo de alocação, pois o banco de dados do SAPA está com as informações necessárias para o próximo passo. É importante observar que, como o *software* ainda é um sistema em desenvolvimento, houve a necessidade de algumas intervenções a fim de que ele conseguisse realizar o que foi proposto.

3.2 – Processo de alocação

O processo de alocação é constituído por algumas etapas. Para dar início, o banco de dados deve ser replicado, manualmente, para o computador que possua a implementação do algoritmo Húngaro para o modelo (PDS), como pode ser observado na Figura 3.2.



Figura 3.2 – Processo de cópia do Banco de Dados do SAPA.

Após a replicação do banco de dados, o processo de alocação é, então, executado, seguindo os passos apresentados na Tabela 3.1.

Tabela 3.1 - Passos para o processo de alocação.

Passos	Descrição
1	Consulta ao banco de dados. Obtenção dos dados das turmas e salas ordenadas por horários. Ao final ocorre o armazenamento em um arquivo para o programa executável fazer a leitura do mesmo.
2	O programa executável, denominado SATS (Sistema de Alocação de Turmas Sala), lê do arquivo para a memória, as informações das turmas e salas.
3	O programa executável calcula a quantidade de horários e a quantidade de turmas por horário.
4	O programa executável seleciona os dados do próximo horário a ser executado.
5	O programa executável calcula a matriz de custos com base nos requisitos R_3 , R_4 , R_5 e R_6 .
6	Se não for o primeiro horário da semana, o programa verifica a solução do horário anterior e atualiza a matriz de custos de acordo com o requisito R_6 .
7	O programa executa o algoritmo e salva o resultado em arquivo.
8	Se ainda houver horários a serem processados, então o programa retorna para a etapa 4.
9	Gravação de todos os resultados em uma tabela no banco de dados, finalizando a alocação.

O resultado deste trabalho é uma saída não formatada, salva em uma tabela dentro do banco de dados do computador responsável por iniciar o processo de alocação. Esta saída informa qual é a sala designada para a turma (em um determinado horário), conforme pode ser verificado na Figura 3.3, que é uma amostra deste arquivo.

	frkTurmaTurmaSala integer	frkSalaTurmaSala integer	frkHorarioAulasTurmaSala integer
1	9348	903	17072
2	9349	931	17073
3	26289	1189	45169
4	26290	1190	45170
5	48023	1192	84252
6	48024	-1	84253

Figura 3.3 – Saída fornecida pelo algoritmo de alocação.

Observe que não existe nenhum tratamento para a saída, cabendo ao usuário, responsável pela execução da rotina de alocação, criar as ferramentas necessárias para conseguir compreender o resultado que foi gerado.

Vale lembrar que esta saída se encontra em outro computador, ficando, desta forma, totalmente incomunicável com o SAPA, *software* responsável por fornecer o banco de dados para a realização do processo de alocação.

3.3 – Melhorias implementadas no projeto

Para se alcançar as melhorias foram necessários vários ajustes, tanto no SAPA, quanto no computador responsável pelo processo de alocação. O intuito foi sempre o de facilitar a utilização do sistema, principalmente no que tange ao acesso à rotina de designação de salas de aula, a visualização dos resultados computacionais gerados e o gerenciamento do controle de acesso enquanto a rotina estiver sendo executada.

Primeiramente, a questão da replicação do banco de dados foi solucionado informando ao SAPA para se conectar, remotamente, ao banco de dados do computador responsável pelo processo de alocação. Desta forma, tanto as modificações feitas pelos usuários do SAPA quanto as realizadas pelo computador que executa a outra etapa, são direcionadas para um único ponto de armazenamento de dados, conforme pode ser observado na Figura 3.4.

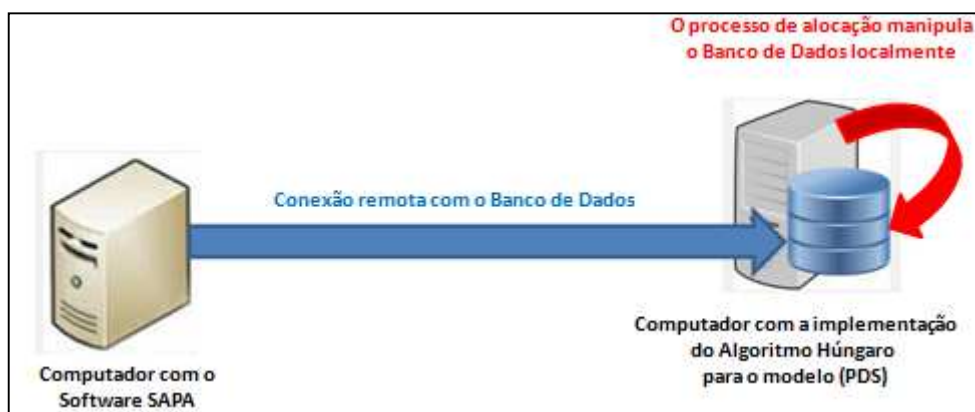


Figura 3.4 – Manobra para acesso a um único Banco de Dados.

O resultado desta manobra é a possibilidade de se trabalhar com as duas aplicações utilizando apenas um banco de dados. Além disso, fica estabelecida a comunicação entre os dois pontos do projeto que, antes, se encontravam isolados; o SAPA e o outro computador.

Desta forma, o SAPA continua atualizando os dados das turmas e salas e o processo de alocação também consegue buscar estes dados instantaneamente e processá-los, gerando a saída representada na Figura 3.3.

3.3.1 – Acesso às rotinas de designação com o SAPA

Com a comunicação estabelecida entre as duas aplicações, executar a alocação através do menu principal do SAPA foi perfeitamente possível. Foi criada uma rotina que pudesse ser acessada via *web*, através da INTRANET. Ela foi responsável por realizar a chamada das etapas especificadas na Tabela 3.1.

A rotina ficou responsável por iniciar o processo através da execução dos passos especificados na Tabela 3.1, conforme pode ser observado na Figura 3.5.

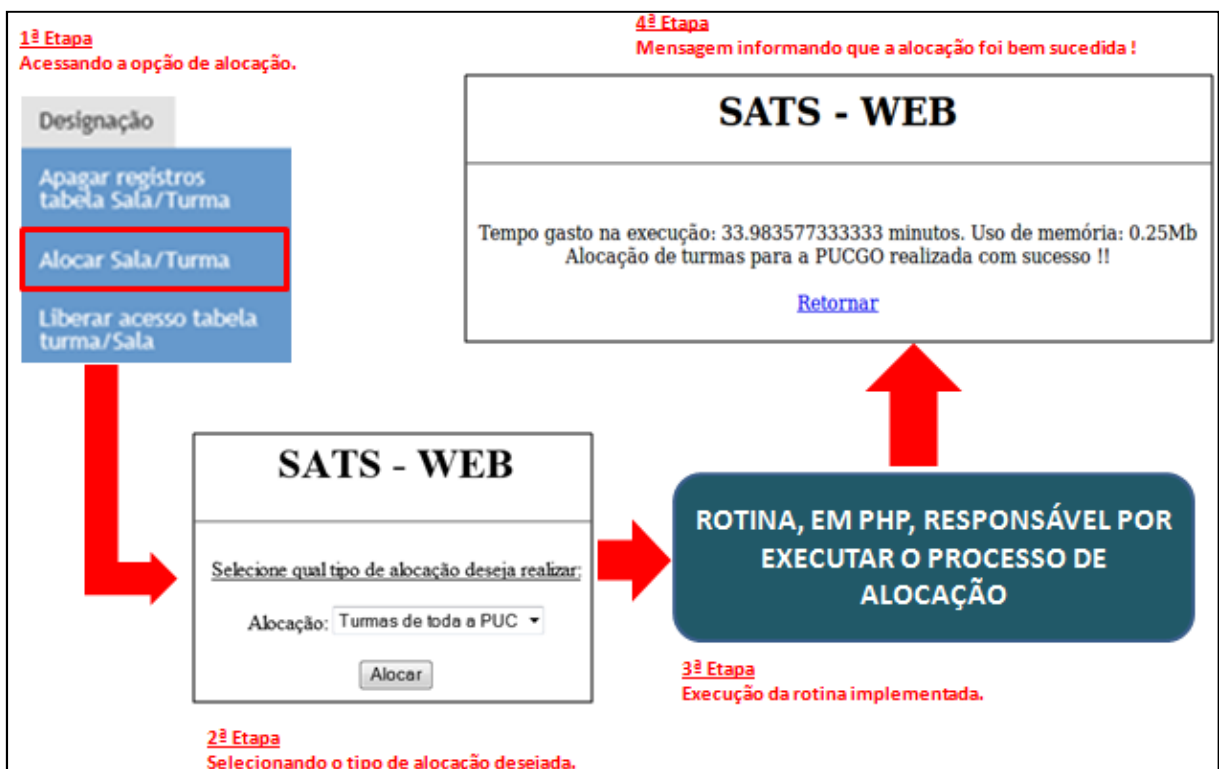


Figura 3.5 – Visão da chamada da rotina implementada no processo de alocação.

3.3.2 – Visualização dos resultados computacionais

Após a alocação, a tabela no banco de dados, responsável por armazenar as informações sobre a designação, é atualizada. Os dados presentes na tabela precisam ser tratados para que seja possível a visualização dos resultados.

O SAPA já possui uma ferramenta gerencial. Ela é responsável por apresentar a relação entre os horários e as turmas, para um determinado período da grade curricular. O relatório foi melhorado, acrescentando os dados referentes à área, bloco e sala designada para cada turma em questão.

Como estas informações já se encontravam em tabela específica dentro do banco de dados, foi necessário realizar ajustes no código do relatório, solicitando a busca destes outros dados. O resultado passou, então, a retornar a disciplina, turma, tipo de aula, vagas disponíveis e onde a mesma foi alocada (área, bloco e sala), como pode ser observado na Figura 3.6.

Programação Acadêmica 2012/1							
Administração de Empresas - Matutino							
1º Período							
Disciplinas	MAF1125-MATEMÁTICA PARA ADMINISTRAÇÃO I	PRE: 04 LAB: 00	CSA1000-TEÓRIAS DA ADMINISTRAÇÃO I	PRE: 04 LAB: 00	PRE: 04 LAB: 00		
	CSA1002-FUNDAMENTOS DE MARKETING	PRE: 04 LAB: 00	RSB1003-TEÓRIAS SOCIOLOGICAS	PRE: 04 LAB: 00			
	LET4101-LÍNGUA PORTUGUESA I	PRE: 04 LAB: 00					
	seg	ter	qua	qui	sex	sáb	
07:10	CSA1000 A01 P V:60 01 G 301 CSA1000 A02 P V:60 01 F 401	MAF1125 A01 P V:60 01 F 309 MAF1125 A02 P V:60 01 F 310 MAF1125 A03 P V:60 01 G 303 MAF1125 A04 P V:60 01 F 402	CSA1002 A01 P V:60 01 F 309 CSA1002 A02 P V:60 01 F 310	CSA1000 A01 P V:60 01 G 301 CSA1000 A02 P V:60 01 F 401	MAF1125 A01 P V:60 01 F 309 MAF1125 A02 P V:60 01 F 310 MAF1125 A03 P V:60 01 G 303 MAF1125 A04 P V:60 01 F 402		07:10
09:00	LET4101 A08 P V:60 01 F 304 LET4101 A17 P V:60 01 F 305 LET4101 A21 P V:60 01 F 306	CSA1002 A01 P V:60 01 F 402 CSA1002 A02 P V:60 01 F 403	CSA1002 A01 P V:60 01 F 309 CSA1002 A02 P V:60 01 F 310	LET4101 A08 P V:60 01 F 304 LET4101 A17 P V:60 01 F 305 LET4101 A21 P V:60 01 F 306	HGS1003 A01 P V:60 01 F 402 HGS1003 A02 P V:60 01 F 403		09:00
10:50							10:50

07:10	Disciplina	CSA1000
	Turma	A01 P V:60
	Tipo de Aula	01 G 301
	Vagas	01 F 401

Área →
 Bloco →
 Sala →

Figura 3.6 – Relatório com os ajustes finais.

3.3.3 – Gerenciando o acesso à rotina de designação de salas de aula

Para facilitar a compreensão desta melhoria é importante entender como os dois computadores do projeto (servidores *web*) estão se interagindo com a rede de dados da PUC Goiás.

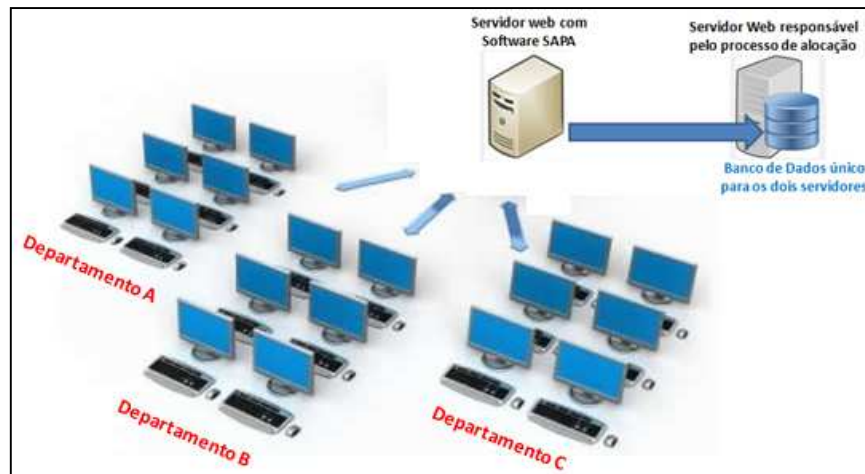


Figura 3.7 – Interação entre os servidores *web* e a rede da PUC Goiás.

Observe, na Figura 3.7, que o único ponto para acessar o projeto é pelo SAPA. As requisições de alocação são solicitadas apenas por usuários autorizados e dentro dos seus níveis de permissão.

O SAPA pode ser visualizado por qualquer usuário da rede interna da PUC Goiás (INTRANET) que possua a permissão de acesso ao *software*. Por este motivo, o controle à rotina de designação de salas de aula deve ser gerenciada.

Para se alcançar tal nível de segurança, primeiramente, deve-se liberar acesso à rotina de alocação somente para usuários cadastrados com essa permissão. Caso o usuário não a tenha, não conseguirá executar a rotina.

Com esta segurança implementada, dois ou mais clientes, com a mesma permissão, não poderão solicitar a execução do mesmo processo de alocação, simultaneamente. A Figura 3.8 apresenta a mensagem de controle quando se tenta burlar esse mecanismo.

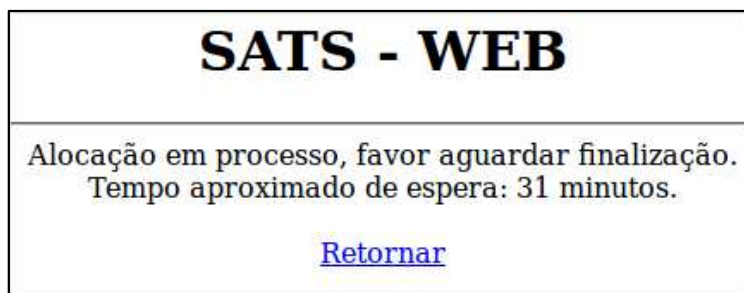


Figura 3.8 – Mensagem de controle de acesso à rotina de designação.

3.3.4 – Limpeza dos registros da base de dados do SAPA

A base de dados do SAPA é de suma importância para todo o trabalho. Possuir uma íntegra e confiável é mais do que necessário para que os processos envolvidos sejam executados corretamente.

Sendo assim, iniciou-se uma análise criteriosa para verificar se a base de dados estava ou não em condições de ser utilizada. Em conversa com o professor Ivon Canedo, desenvolvedor e responsável direto pelo SAPA, foi detectado que os registros do primeiro semestre de 2012, das turmas do departamento de computação da PUC Goiás, apresentavam inúmeras duplicações.

Desta forma, todo o processo de alocação ficaria prejudicado, pois aconteceriam designações de salas para turmas que, sequer existiam. Além do mais, o recurso sala estaria sendo utilizado de forma inadequada. Diante desta situação, iniciou-se o processo de limpeza destes registros duplicados.

Primeiramente, foi feito o levantamento de quais e do exato número de turmas que cada curso da PUC Goiás deveria possuir no primeiro semestre de 2012. Posteriormente, foi feita a confirmação destas informações junto à CPAC. Este trabalho foi muito importante, pois mostrou em que momento a limpeza deveria ser finalizada.

De posse desses dados, iniciou-se o desenvolvimento de rotinas com o objetivo de realizar a limpeza dos registros duplicados na base de dados. A Tabela 3.2 apresenta a sequência utilizada para se alcançar o resultado.

Tabela 3.2 - Passos para a limpeza do banco de dados.

Passos	Descrição
1	Inserir os dados de todas as turmas (com os registros duplicados) e as demais informações (nº da turma, período, nome da disciplina, código da turma, código da subturma, código do curso e nome do curso) em uma tabela, dentro do banco de dados, denominada SAPA.
2	Inserir os dados levantados junto à CPAC (todas as turmas existentes no primeiro período de 2012) em outra tabela, do banco de dados, denominada CPAC.
3	Verificar quais dados, entre a tabela CPAC e SAPA, são exatamente iguais e marcar (nas duas tabelas) no campo de controle do registro.
4	Buscar inconsistências nos registros que não foram marcados (a verificação deverá ser realizada campo a campo) e corrigi-los.
5	Executar o passo 3 novamente até que a quantidade de registros da tabela CPAC se iguale à quantidade de registros da tabela SAPA.
6	Deletar todos os registros da tabela TURMA (presente no banco de dados do SAPA) que não foram marcados na tabela SAPA.

Após a realização de todas as etapas informadas acima, a base de dados do primeiro semestre de 2012 estará pronta para dar início ao processo de alocação (descrito na Tabela 3.1).

3.3.5 – Ajuste na ordem de processamento das turmas, por área, da PUC Goiás

A percepção de que seria interessante realizar a alteração na ordem de processamento das turmas, por área, da PUC Goiás, veio a partir da análise dos resultados de Ribeiro (2012). Ele resolve o mesmo problema, apresentando eficiência na alocação de turmas às áreas de origem. Assim, é possível identificar quais áreas são e quais não são autosuficientes em termos de espaço físico. Estas informações podem ser observadas na Tabela 3.3.

Tabela 3.3 - Eficiência na alocação de turmas às áreas de origem.

Área	Eficiência	Situação
I	100%	Autosuficiente
II	80%	Não é autosuficiente
III	30%	Não é autosuficiente
IV	100%	Autosuficiente
V	100%	Autosuficiente
VI	100%	Autosuficiente
IX	100%	Autosuficiente

Fonte: Ribeiro (2012)

Diante delas, foi proposta a tentativa de melhorar os resultados tanto da área II quanto da área III. As demais áreas já possuíam resultados muito satisfatórios. Desta forma, iniciou-se uma investigação mais detalhada sobre como o processo de alocação realizava a busca pelas turmas e se as áreas estavam influenciando nesta busca.

A primeira etapa do processo de alocação se resume, basicamente, em trazer as turmas e salas, da base de dados do SAPA, seguindo a ordem dos horários das turmas.

O resultado destas buscas são armazenados em arquivos. A Figura 3.9 apresenta o arquivo turmas.data. Nele, é possível observar a ordenação sendo feita pelo identificador do horário das aulas da turma.

1 turmas.data			
43833	1	1;22;4;20:30	76878 0;60
43834	1	1;22;2;20:30	76879 0;60
43834	1	1;22;5;20:30	76880 0;60
43835	6	2;27;2;18:45	76881 0;60
43835	6	2;27;5;18:45	76882 0;60
43836	3	1;16;4;20:30	76885 0;30
43836	3	1;16;7;18:45	76883 0;30
43836	3	1;16;7;20:30	76884 0;30
43837	3	1;16;4;18:45	76887 0;60
43837	3	1;16;7;17:00	76886 0;60
43838	5	1;26;3;17:00	76888 0;30
43839	5	1;26;6;17:00	76889 0;15
43842	5	1;26;4;17:00	76892 0;30

A ordenação acontece pelo campo que armazena os horários das aulas da turma.

O campo responsável por informar a área não influencia a ordem da listagem das turmas

Figura 3.9 – Arquivo turmas.data.

O processo de alocação não realiza a designação das turmas de uma área por vez, mas sim de uma porção de turmas de diversas áreas.

A partir dessa forma de processamento, as áreas que são autosuficientes em termos de espaço físico, não apresentarão remanejamentos desnecessários, pois continuarão conseguindo alocar todas as suas turmas em suas áreas de origem.

A questão muda quando se analisa áreas que não são autosuficientes. Caso o processo de alocação fosse iniciado por estas, uma quantidade maior de turmas (da área) conseguiria ser alocada em sua área de origem, melhorando assim, o quadro de eficiência (Tabela 3.3).

Para que fosse possível realizar essa mudança na ordem de processamento, foi necessário alterar o código que realizava a busca das informações referentes às turmas, na base de dados do SAPA. A Figura 3.10 apresenta estes ajustes.

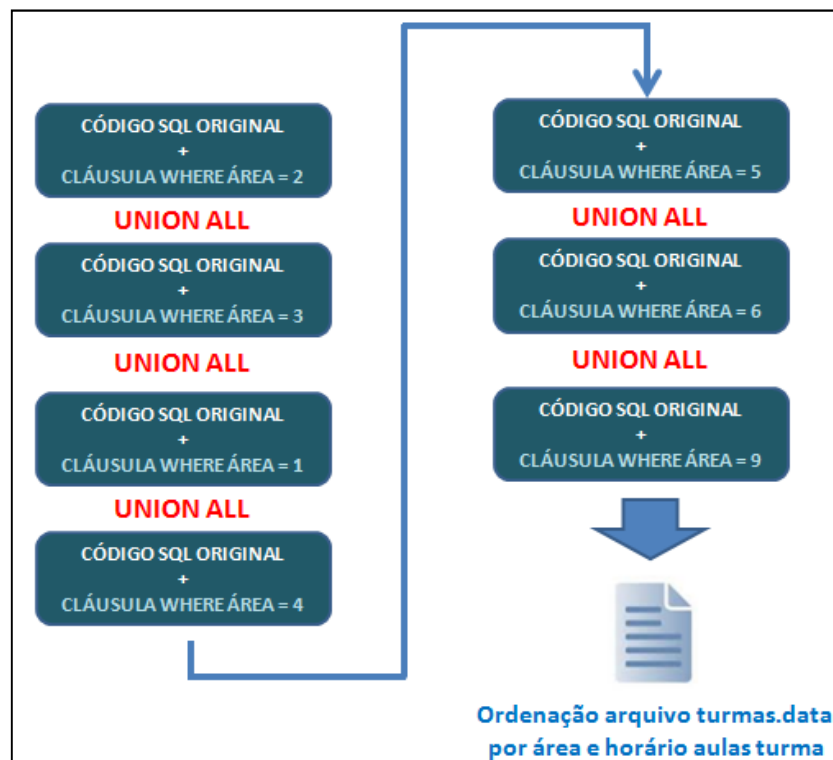


Figura 3.10 – Ordenando o arquivo turmas.data.

Após a alteração desta busca e a execução dos passos da Tabela 3.1 irão gerar um novo arquivo turmas.data. Agora, assim, ele virá ordenado por área e por horário das aulas da turma. A sequência, por área, seguirá a apresentada na Tabela 3.4.

Tabela 3.4 - Sequência da nova ordenação de turmas.

Posição	Área
1 ^a	II
2 ^a	III
3 ^a	I
4 ^a	IV
5 ^a	V
6 ^a	VI
7 ^a	IX

Por causa dessa alteração, as áreas II e III serão as primeiras a participarem do processo de alocação. As comparações e os resultados alcançados tanto por essa mudança na ordenação quanto pela limpeza na base de dados do SAPA serão apresentados no próximo capítulo.

CAPÍTULO IV – RESULTADOS E COMPARAÇÕES

Neste capítulo apresentaremos os resultados alcançados com o uso do *software* desenvolvido neste trabalho e a comparação destes resultados com os de Ribeiro (2012). Ambos aplicaram o algoritmo Húngaro para o modelo (PDS), porém, neste trabalho, houve o ajuste na base de dados e no processo de ordenação das turmas.

Em ambos os trabalhos, foram cadastrados dados reais da PUC Goiás relativo ao primeiro semestre de 2012. O banco de dados utilizado foi o do SAPA. Desta forma, os problemas apresentam 1242 disciplinas dispostas em 43 horários diferentes durante a semana nos turnos matutino, vespertino e noturno. Isto resulta em um total de 5116 turmas que devem ser alocadas em 312 salas, divididas em 4 campi, 7 áreas e 20 blocos.

Após a execução do processo de alocação (descrito na Tabela 3.1), os dados com as turmas e salas alocadas são armazenados em uma tabela específica no banco de dados. A partir daí, foram desenvolvidos relatórios para melhor compreensão dos resultados.

4.1 – Resultados

Após a limpeza da base de dados e da ordenação das turmas por área, o processo de alocação foi executado. O tempo gasto para a designação das salas de aula foi de, aproximadamente, 34 minutos, conforme pode ser observado na Figura 4.1. É importante ressaltar que todos os requisitos essenciais foram atendidos e, com o menor custo.

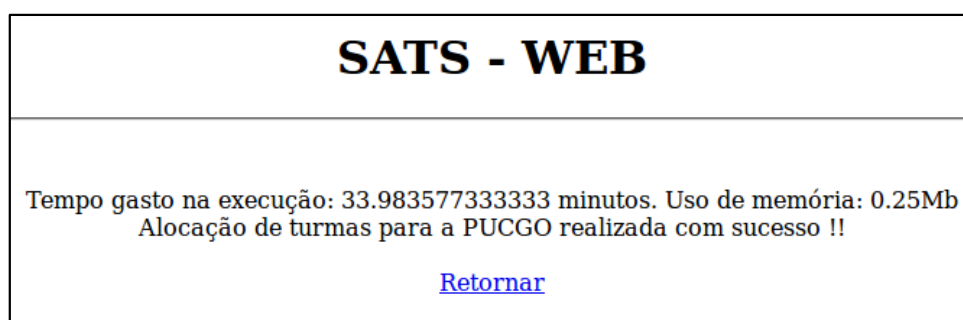


Figura 4.1 – Tempo gasto na alocação (em minutos).

Finalizada a designação das salas, iniciamos o desenvolvimento de alguns relatórios gerenciais com o intuito de compreender melhor os resultados alcançados. O primeiro relatório desenvolvido (Tabela 4.1) tem como objetivo visualizar a eficiência de alocação das turmas às suas áreas de origem.

Na Tabela 4.1 é possível observar os cursos de graduação existentes na PUC Goiás e em qual área o mesmo se encontra, o quantitativo de turmas alocadas em cada uma das áreas, o totalizado de turmas alocadas em suas áreas de origem, o percentual de eficiência alcançado pelo SATS e pela CPAC. Ao final da Tabela 4.1 apresentamos o totalizado de turmas alocadas em cada área da PUC Goiás, o total de turmas analisadas e o percentual final de eficiência na alocação de turmas às áreas de origem apresentado pelo SATS e pela CPAC.

Dando continuidade aos relatórios gerenciais, o próximo foi formulado para informar, em cada área, o quantitativo de salas que estão sendo ocupadas por dia da semana, horário por horário. A Figura 4.2 dá início a estas análises.

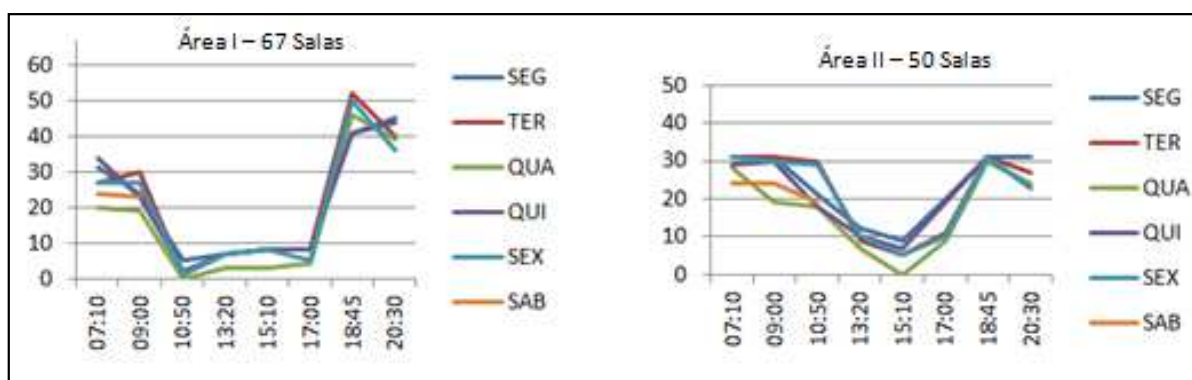


Figura 4.2 – Ocupação de salas das áreas I e II.

Na Figura 4.2, nos é apresentado a informação de que, na área I, existe uma oferta de 67 salas, mas em nenhum momento essa capacidade é superada. Além disso, de acordo com a Tabela 4.1, todas as alocações das suas turmas (área I) estão em sua área de origem. A maior ocupação de salas de aula acontece nos períodos matutino e noturno.

A partir das informações apresentadas na Tabela 4.1, é possível calcular também, por área, a porcentagem de eficiência na alocação de turmas em suas áreas de origem, a saber: na Tabela 4.1, faz-se a adição das turmas das respectivas áreas, conforme cada coluna no cabeçalho ÁREAS. Toma-se a sua soma e a divide pela soma total da área correspondente na coluna TOTAL. A Tabela 4.2 apresenta estes resultados.

Tabela 4.2 - Eficiência, por área, na alocação de turmas às áreas de origem.

Área	Valores alcançados
I	100%
II	94%
III	74%
IV	100%
V	100%
VI	100%
IX	100%

É possível observar, a partir da Tabela 4.2, que 5 das 7 áreas da PUC Goiás conseguem alcançar 100% de eficiência. A área II alcança 94% e a área III, 74%. Desta forma, surge a pergunta: por que estas duas áreas não conseguiram alcançar 100% de eficiência como as demais?

Daremos início às explicações a partir da área II. Nela, temos à disposição 50 salas de aula. É interessante observar, através da Figura 4.2, que a distribuição de aulas acontece, com maior frequência, nos turnos matutino e noturno e, em momento algum, a capacidade total de salas da área II é alcançada. A explicação a respeito dos remanejamentos para as áreas I e IV se dá pelo fato das turmas realocadas necessitarem de salas com capacidade superior às existentes na área II, naquele determinado horário. Esta conclusão foi possível a partir da análise dos seus remanejamentos, conforme pode ser observado na Figura 4.3.

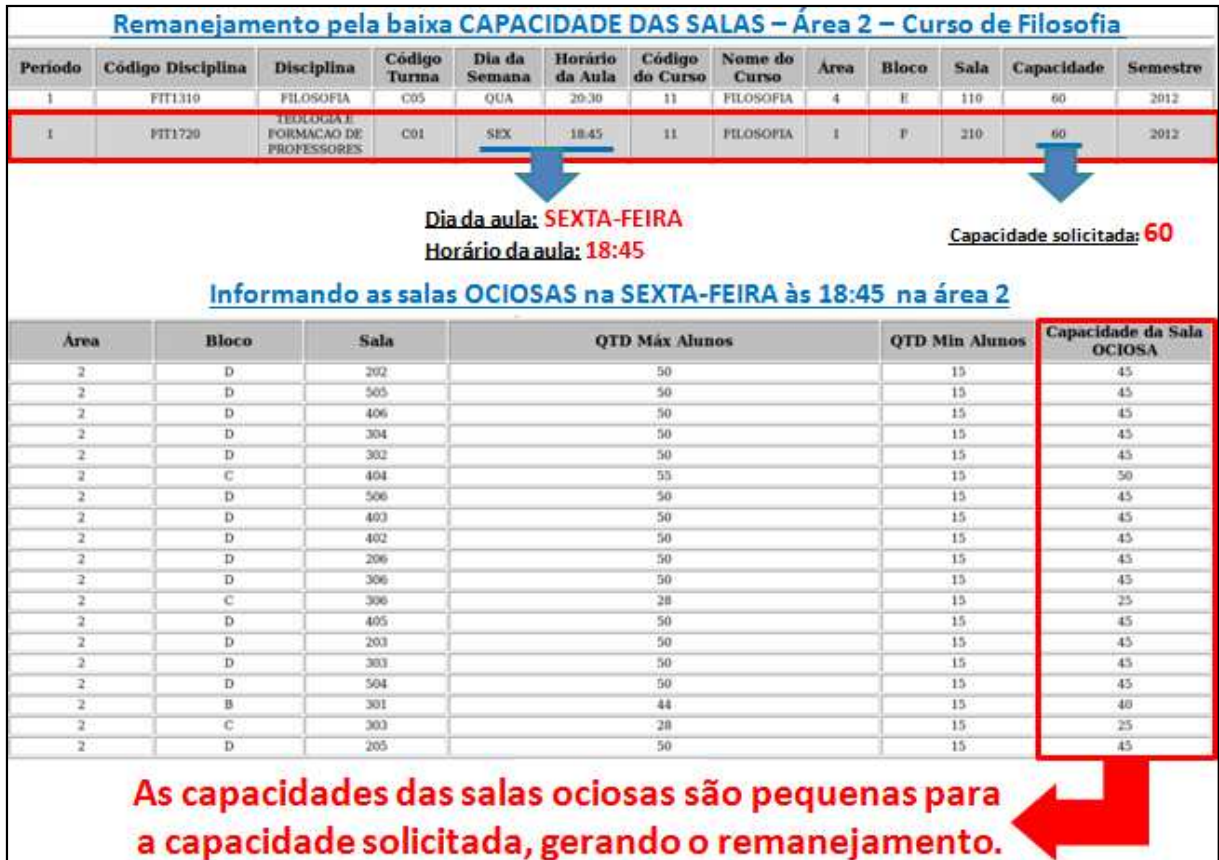


Figura 4.3 – Remanejamento da área II por baixa capacidade de sala.

Analisando as informações apresentadas na Figura 4.3, a turma referente à disciplina “Teologia e formação de professores”, do 1º período do curso de Filosofia (horário de sexta-feira - 18:45), precisava de uma sala de preleção que comportasse 60 alunos. As salas ociosas, naquele determinado horário, possuem capacidade inferior à necessária. Por esse motivo, a turma teve que ser remanejada para a área I.

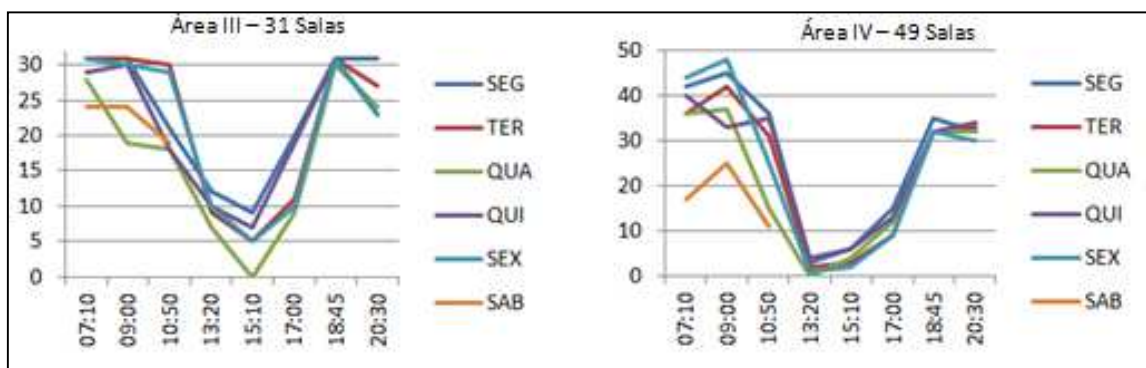


Figura 4.4 – Ocupação de salas das áreas III e IV.

A explicação sobre o percentual de eficiência da área III inicia-se com a análise da Figura 4.4. Ela apresenta a ocupação das salas das áreas III e IV. Na área III temos uma ocupação acentuada das 31 salas de preleção em, praticamente, 2 turnos. Conforme Tabela 4.2, a eficiência na alocação de suas turmas na área de origem é de 74%.

Os remanejamentos da área III para as áreas I, II e IV acontecem por dois motivos. O primeiro motivo está relacionado com a indisponibilidade de salas de preleção, em determinados horários nos turnos matutino e noturno, como pode ser observado na Tabela 4.3, em que os dados apresentados representam o número de salas ocupadas.

Tabela 4.3 - Ocupação das 31 salas de preleção da área III.

Horários	Segunda	Terça	Quarta	Quinta	Sexta	Sábado
07:10	31	31	28	29	31	24
09:00	31	31	19	30	30	24
10:50	21	30	18	18	29	19
13:20	12	9	7	10	10	
15:10	9	5	0	7	5	
17:00	20	11	9	19	10	
18:45	31	31	30	31	31	
20:30	31	27	24	23	23	

Isso faz com que os remanejamentos aconteçam nas turmas que se enquadrarem nestes horários, pois não haverá sala de preleção disponível. A Figura 4.5 apresenta esta situação, analisando a turma referente à disciplina “Matemática computacional III”, do 9º período do curso de Ciência da Computação (horário de segunda-feira – 18:45).

Remanejamento por INDISPONIBILIDADE DE SALA – Área 3 – Curso de Ciência da Computação																								
Período	Código Disciplina	Disciplina	Código Turma	Dia da Semana	Horário da Aula	Código do Curso	Nome do Curso	Área	Bloco	Sala	Capacidade	Semestre												
9	CMP1035	MATEMATICA COMPUTACIONAL III	C01	SEG	18:45	28	CIENCIA DA COMPUTACAO	2	D	503	45	2012												
<p>Dia da aula: SEGUNDA-FEIRA Horário da aula: 18:45</p> <p>Informando as salas OCIOSAS na SEGUNDA-FEIRA às 18:45 na área 3</p> <table border="1"> <thead> <tr> <th>Área</th> <th>Bloco</th> <th>Sala</th> <th>QTD Máx Alunos</th> <th>QTD Min Alunos</th> <th>Capacidade da Sala OCIOSA</th> </tr> </thead> <tbody> <tr> <td colspan="6" style="text-align: center;">Quantidade de salas disponíveis: 0</td> </tr> </tbody> </table> <p>A indisponibilidade de salas de aula motivou o remanejamento.</p>													Área	Bloco	Sala	QTD Máx Alunos	QTD Min Alunos	Capacidade da Sala OCIOSA	Quantidade de salas disponíveis: 0					
Área	Bloco	Sala	QTD Máx Alunos	QTD Min Alunos	Capacidade da Sala OCIOSA																			
Quantidade de salas disponíveis: 0																								

Figura 4.5 – Remanejamento da área III por indisponibilidade de sala.

O segundo motivo para os remanejamentos na área III envolve a baixa capacidade das salas ociosas, em determinados horários. A Figura 4.6 faz a análise da turma referente à disciplina “História da Engenharia de Computação”, do 1º período do curso de Engenharia de Computação (horário de quinta-feira - 07:10).

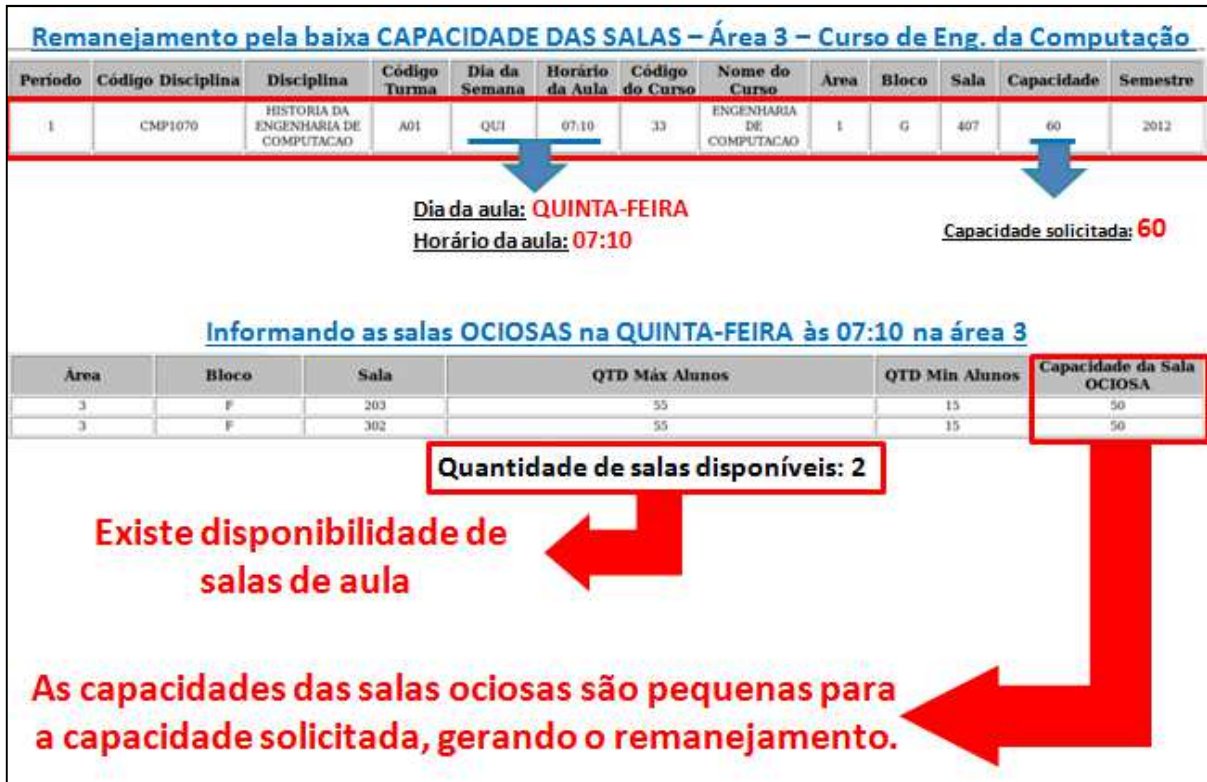


Figura 4.6 – Remanejamento da área III por baixa capacidade de sala.

Portanto, a indisponibilidade de salas de aula e a baixa capacidade das salas ociosas são os responsáveis pelos remanejamentos das turmas da área III.

A área IV conseguiu alocar as suas turmas com 100% de eficiência. As maiores ocupações de suas salas de aula acontecem no turno matutino e noturno. Não se observa também, nesta área, o quantitativo total de salas sendo alcançado. É importante lembrar que, como a área V apresenta mais laboratórios na área da saúde e poucas salas de preleção, as suas turmas acabam tendo aulas na área IV.

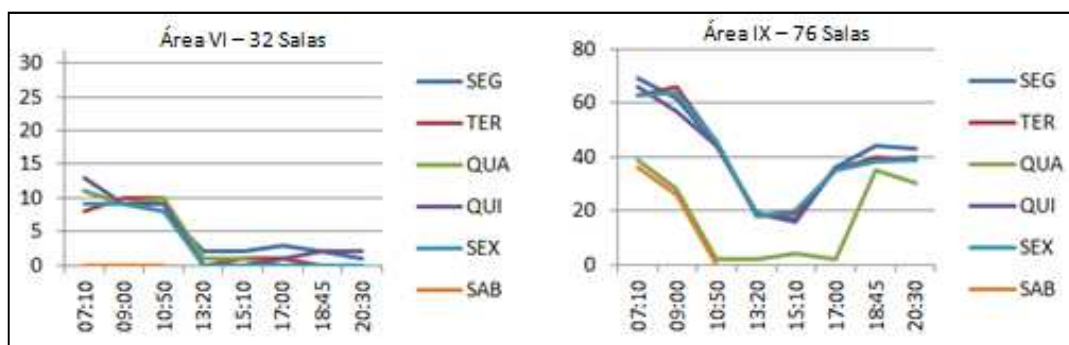


Figura 4.7 – Ocupação de salas das áreas VI e IX.

A Figura 4.7 apresenta as últimas análises a respeito da ocupação das salas de aula da PUC Goiás durante a semana, horário por horário. Nela, informamos os resultados para as áreas VI e IX. Ambas as áreas conseguem alocar, com 100% de eficiência, as suas turmas. A área VI, localizada no campus II da PUC Goiás, recebe a maior quantidade de aulas no turno matutino. A área IX é a que apresenta a maior quantidade de salas de aula. As salas, também, possuem as maiores capacidades. Observa-se uma grande ocupação durante o turno matutino, em alguns dias da semana. Já em outros dias da semana, como na quarta-feira, verifica-se uma ocupação mínima no turno vespertino.

O último relatório gerencial, apresentado pela Tabela 4.4, informa, por curso e período, o horário da aula e em qual área, bloco e sala as turmas estão sendo alocadas. Como exemplo, podemos citar a respeito da disciplina “Cálculo Diferencial e Integral”, A12, do 1º período do curso de Ciência da Computação. De acordo com a Tabela 4.4, todas as aulas de segunda-feira, quarta-feira e quinta-feira, na semana, foram designadas para a área II, bloco C, sala 205.

Tabela 4.4 - Ciência da Computação – Matutino – 1º Período.

Disciplina	Segunda	Terça	Quarta	Quinta	Sexta	Sábado
Cálculo Diferencial e Integral – A12	09:00 02 C 205	-	09:00 02 C 205	09:00 02 C 205	-	-
Física para Computação – A01	-	07:10 03 C 504	-	-	07:10 03 C 504	-
História da Ciência da Computação – A01	-	07:10 03 E 303	-	-	-	-
Lógica Computacional Algorítmica I – A01	07:10 03 E 403	-	-	-	-	-
Programação de Computadores I – A05	09:00 03 E 305	-	-	09:00 03 E 305	-	-

4.2 – Comparação dos resultados

Diante das mudanças propostas, os resultados de alocação foram gerados e os dados foram comparados com os alcançados por Ribeiro (2012) e pela CPAC. O presente trabalho possibilitou confrontar algumas informações, confirmando, assim, a melhoria dos resultados.

4.2.1 - Eficiência na alocação de turmas às suas áreas de origem

A Figura 4.8 apresenta os resultados da eficiência na alocação de turmas em suas áreas de origem, por área e no geral.

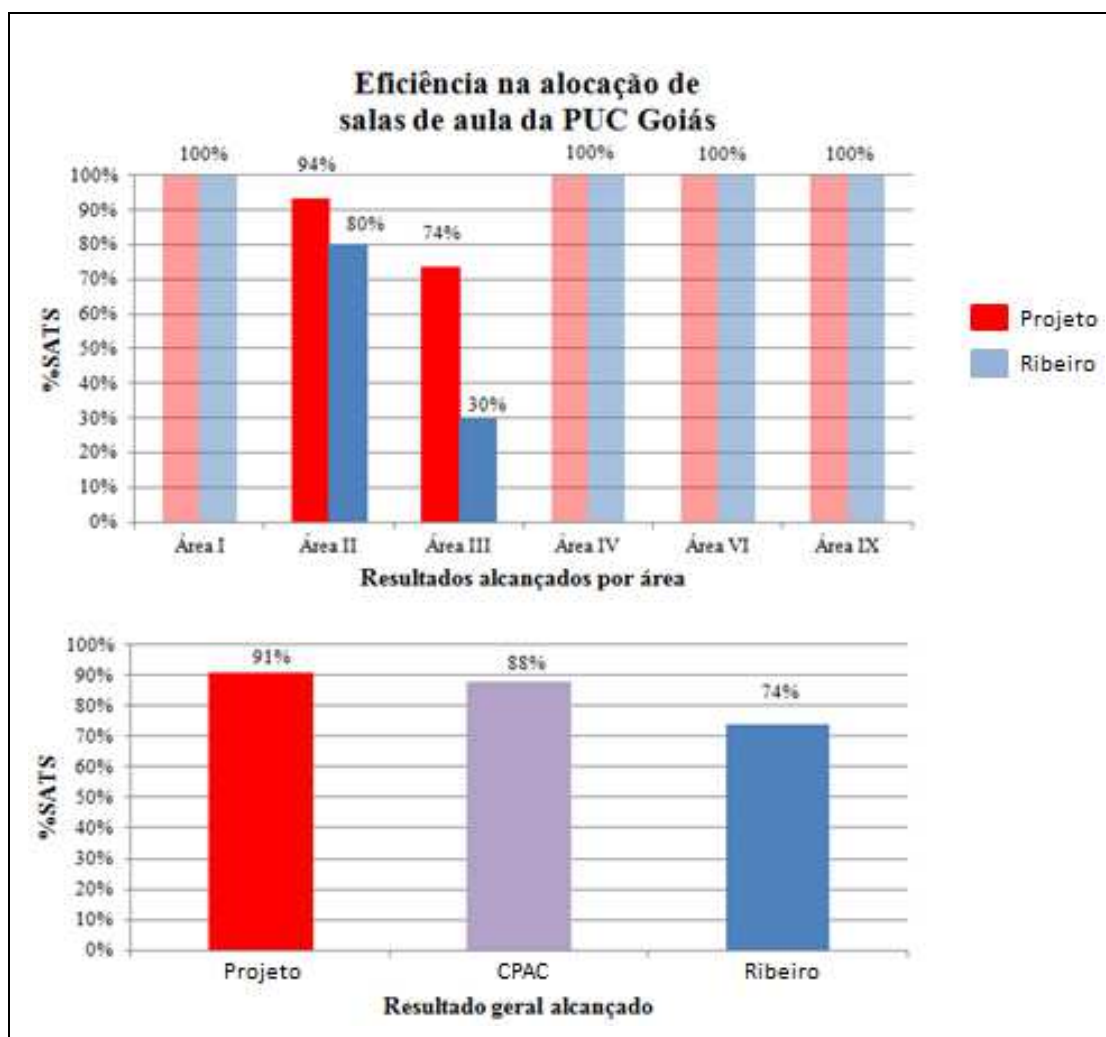


Figura 4.8 – Comparação dos resultados de eficiência na alocação.

No primeiro gráfico, observa-se uma melhora significativa no percentual de eficiência na alocação de salas de aula tanto na área II (94%) quanto na área III (74%) em relação aos dados encontrados por Ribeiro (2012).

Desta forma, o resultado geral alcançado, que foi de 91%, ultrapassou tanto os valores obtidos pela CPAC, de 88%, quanto os de Ribeiro (2012), 74%. Estas informações podem ser verificadas no segundo gráfico.

4.2.2 - Remanejamentos de turmas

Caso uma turma não seja alocada em sua área de origem, ela é considerada um remanejamento para a área. A partir das análises anteriores realizadas (Tabela 4.2), é possível verificar que houve remanejamento, de turma fora da área de origem, somente nas áreas II e III.

A Figura 4.9 dá início às apresentações destes dados. Ela informa tanto os remanejamentos da área II neste trabalho quanto os apresentados no trabalho de Ribeiro (2012). No trabalho em questão, os remanejamentos foram direcionados apenas para as áreas I e IV. De um total de 562 turmas, apenas 36 não ficaram na área II, sendo que 6 foram para a área I e 30 para a área IV.

No trabalho comparado, os remanejamentos foram direcionados para as áreas I, IV, VI e IX. Do total de 562 turmas, 59 não ficaram na área II, sendo que 11 foram para a área I, 37 foram para a área IV, uma foi para a área VI e 10 foram para a área IX.

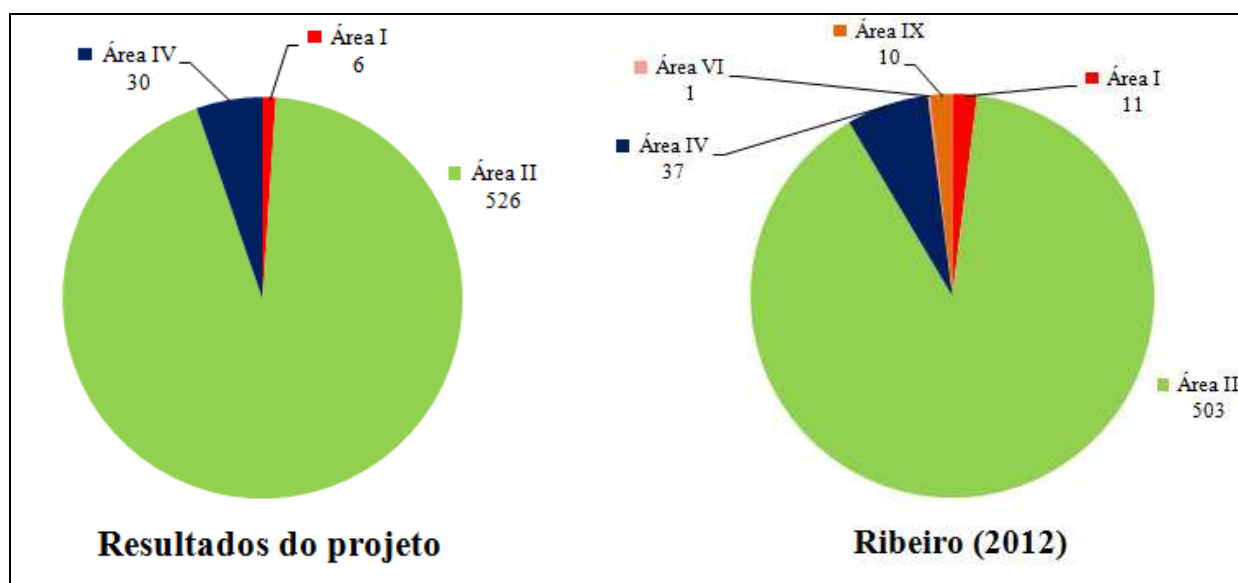


Figura 4.9 – Remanejamento de turmas da área II.

A Figura 4.10 apresenta os remanejamentos da área III neste trabalho e no de Ribeiro (2012). No trabalho em questão, os remanejamentos foram direcionados apenas para as áreas I, II e IV. De um total de 1324 turmas, apenas 425 não ficaram na área III, sendo que 81 foram para a área I, 155 foram para área II e 189 para a área IV.

No trabalho de Ribeiro (2012), os remanejamentos foram direcionados para as áreas I, II, IV, VI e IX. Do total de turmas, 933 não ficaram na área III, sendo que 305 foram para a área I, 314 foram para a área II, 161 foram para a área IV, 50 foram para a área VI e 107 foram para a área IX.

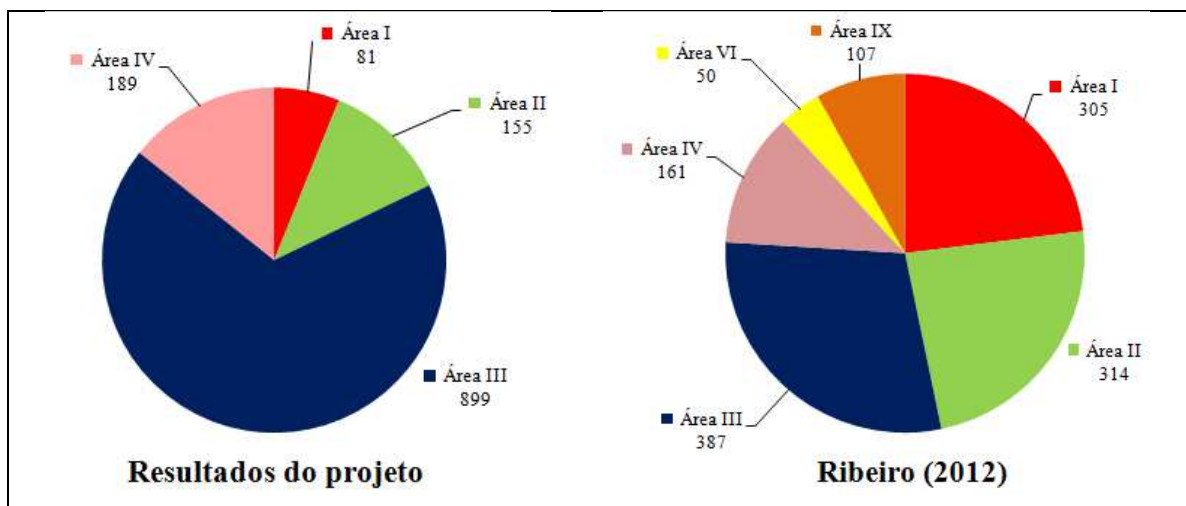


Figura 4.10 – Remanejamento de turmas da área III.

4.2.3 – Tempo de alocação

A Tabela 4.5 apresenta os resultados finais após a execução do processo de designação da CPAC, de Ribeiro (2012) e do projeto em questão. O procedimento da CPAC é manual. No sistema operacional Linux, as configurações foram as seguintes: em Ribeiro (2012), processador Intel Core 2 Duo, com clock de 2,67 GHz e 2 GB de memória RAM disponível com Linux kernel 2.7 de 64 bits, enquanto aqui utilizamos um processador AMD Dual Core, com um clock de 1.30 GHz, 2 GB de memória RAM disponível e com Linux Mint 17.

Tabela 4.5 - Comparativo de tempo para alocar as turmas da PUC Goiás.

Projeto	Ribeiro (2012)	CPAC
34 minutos	40 minutos	1 mês e meio

O motivo pelo qual o processo de alocação de salas, realizado pela CPAC, exige um mês e meio de trabalho inicia-se no período de férias. As turmas são designadas para salas de aula e todas as etapas são registradas no SGA. É importante observar que as turmas poderão vir a sofrer reajustes no quantitativo de alunos ao longo do período de matrícula. Desta forma, a sala alocada poderá não ser mais capaz de suportar essa nova demanda. Em um segundo momento, já no retorno às aulas, o cenário encontrado é o seguinte: turmas precisando ser remanejadas para outras salas de aula, alunos já com a informação dos locais de aulas e cancelamentos de turmas. Soma-se a isto o grande volume de trabalho nas UAAs no momento de matrícula, dificultando ainda mais o processo de comunicação entre todos os envolvidos. Pela razão do fracionamento do processo de alocação, realizado pela CPAC, é que o mesmo

se estende por todo esse tempo (1 mês e meio).

O processo de alocação apresentado neste trabalho é executado em, aproximadamente, 34 minutos. O motivo é que todos os dados necessários para iniciar o procedimento já estão disponíveis na base de dados do SAPA. Assim, não é preciso fracionar o processo em mais etapas, sendo apenas necessário executar os passos descritos na Tabela 3.1 e aguardar o tempo computacional exigido para a designação de turmas em salas de aula, horário por horário.

4.3 – Análise dos resultados

A partir dos resultados apresentados acima, foi possível concluir que, as áreas I, IV, V, VI e IX são autosuficientes em termos de espaço físico, pois conseguiram alocar, com 100% de eficiência, suas respectivas turmas. Em contrapartida, as áreas II e III não são autosuficientes em termos de espaço físico, pois não conseguiram alocar, com 100% de eficiência, suas respectivas turmas.

Foi possível verificar, pela Tabela 4.1, que, o percentual de turmas alocadas pelo SATS, nas áreas recomendadas, foi de 91%. Desta forma, conseguiu-se superar os 88% de turmas alocadas pela CPAC, nas áreas recomendadas e os 74% de turmas alocadas por Ribeiro (2012).

Com relação ao tempo, a CPAC consegue resolver o problema de alocação na PUC Goiás em, aproximadamente, 1 mês e meio. A solução computacional levou, aproximadamente, 34 minutos e Ribeiro (2012) executa o processo em, aproximadamente, 40 minutos.

No próximo capítulo apresentaremos as considerações finais.

CAPÍTULO V - CONSIDERAÇÕES FINAIS

Estudamos o problema de designação de salas de aula da PUC Goiás e partimos da descrição dos requisitos essenciais e não essenciais apresentados pela Coordenação de Programação Acadêmica (CPAC), do modelo matemático para o problema com base nesses requisitos, do banco de dados do *Software* de Apoio à Programação Acadêmica (SAPA), analisando os dados referentes ao 1º semestre de 2012 da PUC Goiás e de uma implementação do algoritmo Húngaro para a resolução do problema.

Oferecemos o acesso à rotina de designação de salas de aula pelo SAPA e a visualização dos resultados computacionais obtidos após a execução da rotina de designação de salas de aula. Gerenciamos o controle de acesso enquanto a rotina de designação de salas de aula estiver sendo executada. Realizamos tanto a limpeza dos registros da base de dados do SAPA quanto o ajuste na ordem de processamento das turmas, por área, da PUC Goiás. Resolvemos o problema, horário por horário, com 5116 turmas e 312 salas de aula em, aproximadamente, 34 minutos. Os resultados apresentados foram muito satisfatórios, pois conseguimos alcançar 91% de eficiência na alocação das turmas em suas áreas de origem. Além disso, foi possível verificar o que realmente causou os remanejamentos de turmas nas áreas II e III. Todos estes resultados foram comparados com os alcançados pela CPAC e por Ribeiro (2012).

Ainda, considerando o artigo de Constantino ET AL. (2010), resolvemos problemas parecidos com tempos parecidos e, se entendermos a “distância entre as salas alocadas e o ponto próximo ao bloco didático desejável de cada curso” como os remanejamentos que o gestor deve gerir, acreditamos que se faz necessário relatórios que venham esclarecer e sugerir alternativas para o gestor; conforme Figuras 4.3, 4.5 e 4.6.

Uma proposta de trabalho futuro é a prática. Ou seja, resolver de maneira efetiva, a cada semestre, o problema de designação de salas de aula da PUC Goiás. Isso possibilitaria o trabalho com o número real de alunos, nas turmas, no momento da designação, que, em princípio, poderia ser às vésperas do início das aulas. Além disso, através de uma análise dos próprios resultados, pode-se inferir para a Administração Superior da PUC Goiás caminhos efetivos para contornar, no futuro, aqueles problemas que causam os remanejamentos em determinada área da Instituição.

Adiantamos que já estamos em contato com a coordenadora da CPAC, professora Ivana Martelli e o autor do Sistema de Gestão Acadêmica da PUC Goiás, professor André Luiz Alves, discutindo sobre uma implementação prática. Os anexos I a VI são sugestões do professor André Luiz Alves como uma espécie de tutorial para o futuro.

A necessidade de designação de salas de aula automatizada já é uma realidade, em virtude de

algumas vantagens como, por exemplo, a minimização de custos para a Instituição, a minimização do deslocamento de alunos e professores para outras áreas da Instituição, etc.. Além disso, inferimos que existem várias Universidades no mundo com problemas semelhantes ao nosso de alocação de salas de aula. Portanto, sugerimos a mesma estratégia de solução, isto é, horário por horário e com os requisitos não essenciais na função objetivo, em virtude da qualidade da solução obtida.

REFERÊNCIAS BIBLIOGRÁFICAS

BOAVENTURA, P. O., JURKIEWICZ, S., **Grafos: Introdução e prática**. São Paulo. Editora Blucher, 2009.

BURKE, E.; JACKSON, K.; KINGSTON, J.; WEARE, R., **Automated University Timetabling: The State of the Art**. The Computer Journal, Vol. 40, No. 9, pp. 565-571, 1997.

CAMBAZARD, H.; HEBRARD, E.; O'SULLIVAN, B. AND PAPADOPOULOS, A., **Local search and constraint programming for the post enrolment-based course timetabling problem**. In PATAT2008: Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling, 2008.

CARTER, M. W., **A lagrangian Relaxation Approach to the Classroom Assignment Problem**. INFOR, Vol. 27, No 2, pp. 230-246, 1989.

CARTER, M. W.; TOVEY, C. A., **When is the Classroom Assignment Problem Hard?** Operations Research, vol. 40, p. 28-41, 1992.

CARTER, M. W.; LAPORTE, G., **Recent Developments in Practical Course Timetabling**. Practice and Theory of Automated Timetabling PATAT'97, E. K. Burke, and M. W. Carter (Eds.) Springer Verlag Lecture Notes in Computer Science, 1408, pp. 3-19, 1998.

CONSTANTINO, A. C.; MARCONDES, W. F.; LANDA, S. J. D., **Iterated Heuristic Algorithms for the Classroom Assignment Problem**. Practice and Theory of Automated Timetabling PATAT 2010. Proceedings p. 152-166, 2010.

ELLOUMI, A.; KAMOUN, H.; JARBOUI, B.; DAMMAK, A., **The classroom assignment problem: Complexity, size reduction andheuristics**. Applied Soft Computing 14, 677–686, 2014.

FIGUEIREDO, C. M. H. D.; SZWARCFITER, J. L., **Emparelhamento em grafos: Algoritmos e complexidade**. Jornada de Atualização em Informática (JAI), Congresso da SBC, 1999.

FONSECA, G.H.G; SANTOS, H.G.; TOFFOLO, T.A.M; BRITO, S.S. AND SOUZA, M.J.F., **A sa-ils approach for the high school timetabling problem**. In Proceedings of the Ninth International Conference on the Practice and Theory of Automated Timetabling (PATAT 2012), 2012.

FOULDS, L. R., **Combinatorial Optimization for Undergraduates**. Springer-Verlag New York Inc., 1984.

GLASSEY, C. R.; MIZRACH, M., **A Decision Support System for Assigning Classes to Rooms**. Interfaces, Vol. 16, No. 5, pp. 92-100, 1986.

GOSSELIN, K.; TRUCHON, M., **Allocation of Classrooms by Linear Programming**. J. Operational Research Soc., Vol. 37, No. 6, pp. 561-569, 1986.

KOSTUCH, P., **Timetabling competition - sa-based heuristic**. In PATAT 2004: Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling, 2004.

KRISTIANSEN, S.; STIDSEN, T. R., **A Comprehensive Study of Education Timetabling, a Survey**. DTU Management Engineering, 2013.

KUHN, H. W., **The hungarian method for the assignment problem**. Naval Research Logistics Quarterly, v. 2, p. 83-97, 1955.

MACULAN, N., FAMPA M. H., **Otimização linear**. Brasília. Editora UnB, 2006.

MCCOLLUM, B., **A perspective on bridging the gap between theory and practice in university timetabling**. In E. Burke & H. Rudová (Eds.), LNCS: Vol. 3867. Practice and Theory of Automated Timetabling VI (pp. 3-23). Berlin: Springer., 2007.

MCCOLLUM, B.; MCMULLAN, P.; PAECHTER, B.; LEWIS, R.; SCHAERF, A.; DI GASPERO, L.; PARKES, A. J., QU, R.; BURKE, E. K., **Setting the research agenda in automated timetabling: the second international timetabling competition**. *INFORMS Journal on Computing*, Vol. 22, p. 20-130, 2010.

MÜLLER, T., **Itc2007 solver description: a hybrid approach**. *Annals of Operations Research*, 172:429-446, ISSN 0254-5330. 2009.

NEVES, K. F. S., **Uma implementação para o problema de designação de salas de aula para a PUC Goiás: Um estudo de caso no Departamento de Computação**. Monografia, Departamento de Computação da PUC Goiás. 2010.

PUC EM DADOS 2013, 2013, Goiânia.

RIBEIRO, J., **O problema de designação de salas de aula da PUC Goiás**. Dissertação de mestrado. Departamento de Computação da PUC Goiás. 2012.

SALKIN, H. M.; **Integer Programming**. Addison-Wesley Publishing Company, Inc. Philippines, 1975.

SCHAERF, A., **A Survey of Automated Timetabling**. *Artificial Intelligence Review*. Vol. 13, p.87-127, 1999.

SILVA , B. M. N., **Uma implementação para o problema de designação de salas de aula para o Centro Técnico e Científico da PUC Goiás**. Monografia, Departamento de Computação da PUC Goiás. 2011.

SUBRAMANIAN, A.; MEDEIROS, J. M. F.; FORMIGA, L. A., SOUZA, M. J. F., “Aplicação da Metaheurística Busca Tabu ao Problema de Alocação de Aulas a Salas em uma Instituição Universitária”. **Revista Produção Online**, v.11,n.1, p.54-75, mar., 2011.

WREN, A., **Scheduling, timetabling and rostering - a special relationship?** In Edmund Burke and Peter Ross, editors, Practice and Theory of Automated Timetabling, volume 1153 of Lecture Notes in Computer Science, pages 46-75. Springer Berlin / Heidelberg, 1996.

ANEXO I – ESPECIFICAÇÃO DOS LAYOUTS DOS ARQUIVOS DE ENTRADA (TURMAS, SALAS)

O processo de alocação solicita dois arquivos denominados **TURMAS.data** e **SALAS.data**. A seguir serão apresentadas as suas especificações.

Arquivo TURMAS.data

Abaixo segue as especificações de cada campo do arquivo **TURMAS.data**.

```
id integer,  
area integer,  
campus integer,  
curso integer,  
"diaSemana" integer,  
horario character varying(5),  
"horarioAulasTurma" integer,  
"salaEspecial" integer,  
vaga integer
```

Id - Chave primária da tabela TURMA

Tipo: Inteiro

Area - Código da área da turma

Tipo: Inteiro

Campus - Código do campus da turma

Tipo: Inteiro

Curso - Código do curso da turma

Tipo: Inteiro

diaSemana - Número do dia da semana da turma

Tipo: Inteiro

Horario - Horário do início da aula da turma

Tipo: character varying (5)

horarioAulasTurma - Chave primária da tabela horarioAulasTurma da turma

Tipo: Inteiro

SalaEspecial - Campo informando se disciplina necessita de sala especial

Tipo: Inteiro

Vaga - Vagas oferecidas pela turma

Tipo: Inteiro

Arquivo SALAS.data

Abaixo segue as especificações de cada campo do arquivo **SALAS.data**.

```
id integer,  
area integer,  
campus integer,  
curso integer,  
salaEspecial integer,  
capacidadeSala integer,  
max integer,  
min integer,  
peso integer
```

id - Chave primária da tabela SALA

Tipo: Inteiro

area - Código da área da SALA

Tipo: Inteiro

campus - Código do campus da SALA

Tipo: Inteiro

Curso - Código do curso da SALA

Tipo: Inteiro

salaEspecial - Campo informando se a SALA é especial

Tipo: Inteiro

capacidadeSala - Campo informando a capacidade da SALA

Tipo: Inteiro

Max - Campo informando a quantidade máxima de alunos da SALA

Tipo: Inteiro

Min - Campo informando a quantidade mínima de alunos da SALA

Tipo: Inteiro

Peso - Campo informando o valor do PESO para alocação no BLOCO do curso

Tipo: Inteiro

ANEXO II – ESPECIFICAÇÃO DO LAYOUT DA TABELA DE DESTINAÇÃO DE ESPAÇO FÍSICO POR CURSO

A tabela de destinação de espaço físico por curso no banco de dados do SAPA é denominada **pesoAlocacaoBloco**. A seguir será apresentada a sua especificação.

Tabela pesoAlocacaoBloco

Abaixo segue as especificações de cada campo da tabela **pesoAlocacaoBloco**.

```
prk integer,  
peso integer,  
Curso integer,  
Bloco integer
```

prk - Chave primária da tabela

Tipo: Inteiro

peso – Peso da alocação para cada SALA

Tipo: Inteiro

Curso - Código do curso para cada SALA

Tipo: Inteiro

Bloco – Bloco referente à SALA

Tipo: Inteiro

ANEXO III – ESPECIFICAÇÃO DO LAYOUT DE SAÍDA

A tabela de saída no banco de dados do SAPA é denominada **turmaSala**. A seguir será apresentada a sua especificação.

Tabela turmaSala

Abaixo segue as especificações de cada campo da tabela **turmaSala**.

```
Turma integer,  
Sala integer,  
HorarioAulasTurmaSala integer
```

Turma – Identificador da turma alocada

Tipo: Inteiro

Sala – Identificador da sala alocada

Tipo: Inteiro

HorarioAulasTurmaSala – Horário alocado para a turma na sala

Tipo: Inteiro

ANEXO IV – DOCUMENTAÇÃO PARA INSTALAÇÃO DOS SOFTWARES

A seguir será apresentado o passo a passo da instalação do *software* responsável pelo processo de alocação em um computador e, em outro computador, o SAPA. É importante lembrar que ambos os computadores deverão estar configurados em rede e possuir ip fixo. A última observação é a da necessidade dos códigos do projeto e também o backup do banco de dados **POSTGRES** (estão presentes dentro do CD disponível).

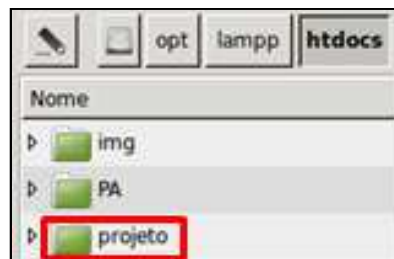
Software responsável pelo processo de alocação

O computador escolhido para receber a instalação do *software* responsável pelo processo de alocação já deverá estar com as seguintes especificações:

- a) Sistema operacional Linux Mint 17 instalado;
- b) Banco de dados **POSTGRES** (acima da versão 1.16.1) instalado;
- c) Aplicação **LAMPP** (Servidor *web* apache e php) instalado;

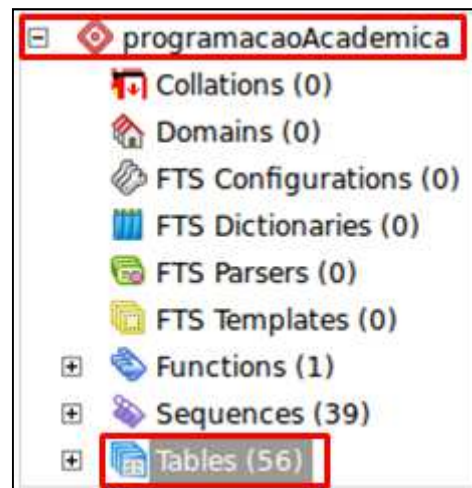
1º Passo – Cópia dos arquivos com os códigos implementados para o servidor web apache

Nesta etapa, toda a pasta **projeto** (copiada do CD disponível) deverá ser inserida dentro da pasta **htdocs** do servidor *web* apache. A Figura abaixo apresenta o resultado.



2º Passo – Inserção do backup do banco de dados **POSTGRES**

Realizar a inserção dos registros do backup para o banco de dados **POSTGRES**. Desta forma, o banco de dados terá todas as informações das turmas e salas referentes ao primeiro semestre de 2012 da PUC Goiás. A Figura abaixo apresenta o resultado.



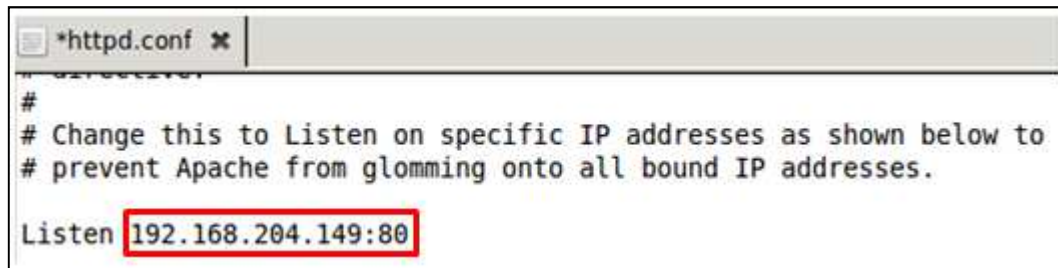
3º Passo – Configuração do IP da máquina **Windows** dentro do projeto

- 1 – Acessar a pasta **OPT/LAMPP/htdocs/PROJETO/sats**;
- 2 – Abrir o arquivo **ip_win.php**;
- 3 – Inserir o ip da máquina (**Windows**) que estiver com o **SAPA** instalado.

```
1 ip_win.php *  
- <?php  
    session_start();  
    $_SESSION['ip_win']="192.168.204.142";  
?>
```

*4º Passo – Configuração do IP da máquina **Linux** dentro do projeto*

- 1 – Acessar a pasta **OPT/LAMPP/etc/**;
- 2 – Abrir o arquivo **httpd.conf**;
- 3 – Inserir o ip da máquina (**Linux**) que estiver com o processo de alocação instalado.



```
*httpd.conf ✕  
#  
# Change this to Listen on specific IP addresses as shown below to  
# prevent Apache from glomming onto all bound IP addresses.  
Listen 192.168.204.149:80
```

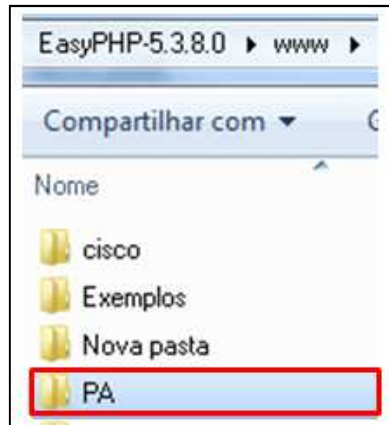
Software de apoio à programação acadêmica - SAPA

O computador escolhido para receber a instalação do **SAPA** já deverá estar com as seguintes especificações:

- a) Sistema operacional Windows 7 (ou superior) instalado;
- b) Aplicação **EASYPHP** (Servidor *web* apache e php) instalado;
- c) Internet Explore 9.

*1º Passo – Cópia dos arquivos do **SAPA** para o servidor web apache*

Nesta etapa, toda a pasta **PA** (copiada do CD disponível) deverá ser inserida dentro da pasta **www** do servidor *web* apache. A Figura abaixo apresenta o resultado.



*2º Passo – Configuração do IP da máquina **Windows** dentro do projeto*

- 1 – Acessar a pasta **EasyPHP-x.x.x.x/conf_files/**;
- 2 – Abrir o arquivo **httpd.conf**;
- 3 – Inserir o ip da máquina (**Windows**) que estiver com o **SAPA** instalado.

 A screenshot of a text editor window showing the 'httpd.conf' file. The editor has a tab labeled 'httpd.conf'. The visible code includes:


```

45 # Change this to Listen on specific IP addresses as shown below to
46 # prevent Apache from glomming onto all bound IP addresses.
47 #
48 Listen 192.168.204.142:80
  
```

 The IP address '192.168.204.142:80' on line 48 is highlighted with a red rectangular box.

*3º Passo – Configuração do IP da máquina **Linux** dentro do projeto*

- 1 – Acessar a pasta **EasyPHP-x.x.x.x/WWW/PA/**;
- 2 – Abrir o arquivo **post.inc.php**;
- 3 – Inserir o ip da máquina (**Linux**) que estiver com o processo de alocação instalado.

 A screenshot of a text editor window showing the 'post.inc.php' file. The editor has a tab labeled 'post.inc.php'. The visible code includes:


```

1 <?php
2
3     session_start();
4     $_SESSION['ip_lnx'] = "192.168.204.149";
  
```

 The IP address '"192.168.204.149"' on line 4 is highlighted with a red rectangular box.

Acessando a página inicial do SAPA

Após realizar todas as etapas descritas, basta abrir o navegador Internet Explorer 9 (de qualquer computador da rede) e digitar o endereço **192.168.204.142/PA/**.

Neste momento, abrirá a página inicial do **SAPA**. Acesse com o usuário administrativo (login: **admin** / Senha: **admin**) para ter acesso ao menu do *software*. A Figura abaixo apresenta o resultado.



ANEXO V – ESPECIFICAÇÃO DOS REQUISITOS DE HARDWARE E SOFTWARE PARA FUNCIONAMENTO DO PROJETO

A seguir serão apresentados os requisitos mínimos para a execução do projeto. É importante lembrar que serão necessários dois computadores, sendo um para a execução do SAPA e o outro para a execução do processo de alocação.

Computador responsável pela execução do processo de alocação

- a) Processador AMD Dual Core, com um clock de 1.30 GHz;
- b) 2 GB de memória RAM disponível;
- c) Linux Mint 17.

Computador responsável pela execução do SAPA

- a) Processador Intel Core i3, com um clock de 2.27 GHz;
- b) 2 GB de memória RAM disponível;
- c) Windows 7 Ultimate (32 bits).

ANEXO VI – MANUAL DO USUÁRIO: COMO PROCEDER A CADA SEMESTRE PARA EXECUTAR O PROJETO

A seguir serão apresentadas as etapas que deverão ser seguidas, a cada semestre, para que o processo de alocação possa ser executado com sucesso. É importante lembrar que todas as etapas anteriores já deverão ter sido realizadas e funcionando corretamente (conforme foram apresentadas). Desta forma, será possível realizar os procedimentos que serão descritos abaixo para o início de cada semestre na PUC Goiás.

Procedimentos que deverão ser realizados no SAPA

Todas as configurações descritas abaixo deverão ser realizadas no SAPA, pois ele é o sistema de gestão acadêmica utilizado neste projeto.

1º Passo – Cadastro de novos departamentos, cursos e disciplinas para o ano letivo

Caso seja necessário realizar o cadastro de algum departamento, curso ou disciplina, o procedimento que deverá ser realizado é o informado abaixo:

- 1 – Logar no **SAPA** como administrador;
- 2 – Acessar a opção **Cadastros** no menu principal;
 - a. Clicar na opção **Departamentos** (para cadastrar um novo departamento);
 - b. Clicar na opção **Cursos** (para cadastrar um novo curso dentro do departamento);
 - c. Clicar na opção **Disciplinas** (para cadastrar uma nova disciplina dentro de um curso).



2º Passo – Cadastro das turmas das disciplinas

Com todos os departamentos, cursos e disciplinas criados, inicia-se o cadastro das turmas que irão compor o quadro de horários do ano letivo. O procedimento para cadastrar todas as turmas será o mesmo e está informado abaixo:

- 1 – Logar no **SAPA** como administrador;
- 2 – Acessar a opção **Elaboração** no menu principal;
- 3 – Informar as opções de **departamento, ano, semestre, curso, grade, turno, período** e **disciplina** que estão sendo solicitados na tela;
- 4 – Preencher os campos abertos com o **código da turma, subturma, vagas, créditos, tipo de aula** e qual o **horário** que será estabelecido para a turma em questão;
- 5 – Clicar no botão **gravar**.

	seg	ter	qua	qui	sex	sab
07:10						
09:00						
10:50						

3º Passo – Cadastro das salas e dos seus pesos para o processo de alocação

Caso seja necessário cadastrar alguma nova sala, juntamente com o seu peso no processo de alocação, seguem as etapas:

- 1 – Logar no **SAPA** como administrador;
- 2 – Acessar a opção **Cadastros** no menu principal;
- 3 – Preencher os campos **Local, Campus, Area, Bloco, Tipo, Capacidade, Requisitos, Horário** e o **peso** para o processo de alocação;
- 4 – Ao final, clicar no botão **Gravar**.

4º Passo – Executando o processo de alocação

Assim que todo o cadastro de departamento, curso, disciplina, turma e salas de aula forem finalizados, será possível executar o processo de alocação. Abaixo seguem as etapas para acessar a tela para tal procedimento:

- 1 – Logar no **SAPA** como administrador;
- 2 – Acessar a opção **Designação** no menu principal;
- 3 – Confirmar o processo de designação clicando no botão **Alocar**;
- 4 – Aguardar a mensagem informando que o procedimento foi bem sucedido.

5º Passo – Visualizando os resultados obtidos pelo processo de alocação

Assim que o processo de alocação foi executado, é possível visualizar os resultados alcançados seguindo as etapas descritas abaixo:

- 1 – Logar no **SAPA** como administrador;
- 2 – Acessar a opção **Programação Acadêmica** no menu principal;
- 3 – Preencher os campos **Departamento, Ano, Semestre, Curso, Grade, Turno e Período**;
- 4 – Clicar no botão **Publicacao**.