Edmar Hell Kampke

Novas Estratégias para Obtenção de Limitantes Inferiores para o Problema de Tabela-Horário de Universidades

Vitória - ES

2020

Edmar Hell Kampke

Novas Estratégias para Obtenção de Limitantes Inferiores para o Problema de Tabela-Horário de Universidades

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Informática (PPGI) da UFES como parte dos requisitos para obtenção do Grau de Doutor em Ciência da Computação.

Universidade Federal do Espírito Santo - UFES

Centro Tecnológico

Programa de Pós-Graduação em Informática

Orientador: Prof. Dr. Geraldo Regis Mauri

Coorientadora: Prof[a]. Dr[a]. Maria Claudia Silva Boeres

Vitória - ES

2020

Ficha catalográfica disponibilizada pelo Sistema Integrado de Bibliotecas - SIBI/UFES e elaborada pelo autor

---

Kampke, Edmar Hell, 1982-

K15n Novas Estratégias para Obtenção de Limitantes Inferiores para o Problema de Tabela-Horário de Universidades / Edmar Hell Kampke. - 2020.

112 f. : il.

Orientador: Geraldo Regis Mauri.

Coorientadora: Maria Claudia Silva Boeres.

Tese (Doutorado em Informática) - Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Otimização Combinatória. 2. Modelos Matemáticos. 3. Teoria dos Grafos. 4. Programação Linear. 5. Universidades e Faculdades. I. Mauri, Geraldo Regis. II. Boeres, Maria Claudia Silva. III. Universidade Federal do Espírito Santo. Centro Tecnológico. IV. Título.

CDU: 004

Edmar Hell Kampke

Novas Estratégias para Obtenção de Limitantes Inferiores para o Problema de Tabela-Horário de Universidades

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Informática (PPGI) da UFES como parte dos requisitos para obtenção do Grau de Doutor em Ciência da Computação.

Trabalho Aprovado. Vitória - ES, 07 de Agosto de 2020.

Prof. Dr. Geraldo Regis Mauri Orientador	**Profª. Drª. Maria Claudia Silva Boeres** Coorientadora
Prof. Dr. André Renato Sales Amaral Universidade Federal do Espírito Santo (UFES)	**Profª. Drª. Lucia Catabriga** Universidade Federal do Espírito Santo (UFES)
Prof. Dr. Luciano Lessa Lorenzoni Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo (IFES)	**Prof. Dr. Haroldo Gambini Santos** Universidade Federal de Ouro Preto (UFOP)

Vitória - ES
2020

Aos meus pais, Helmar e Ilza.

Agradecimentos

Agradeço a Deus, por guiar os meus passos e renovar minhas forças a cada dia.

Agradeço aos meus pais, Helmar e Ilza, pelo amor e apoio incondicional. Obrigado pelas lições de vida e fé que me tornaram uma pessoa forte e persistente.

Aos meus irmãos, Vera e Edgar, e demais familiares e amigos, pelo carinho, apoio e tantos momentos de alegria.

Agradeço ao meu orientador, Professor Geraldo, pelo conhecimento compartilhado, pela atenção, por confiar em mim e pelas várias oportunidades que me fizeram crescer na vida acadêmica.

Agredeço à minha coorientadora, Professora Maria Claudia, e também a Professora Maria Cristina, por acreditarem em mim e terem me dado essa oportunidade. Faltam palavras para descrever o meu agradecimento pelo incentivo, paciência e compreensão que sempre demonstraram por mim. Vocês sempre serão um grande exemplo para mim.

Aos Membros da Banca Examinadora pela participação na defesa deste trabalho e pelas valiosas contribuições.

Aos demais professores, colaboradores e colegas do Programa de Pós-Graduação em Informática (PPGI) da Universidade Federal do Espírito Santo (UFES), que sempre estiveram dispostos a ajudar. Em especial, agradeço aos colegas do Labotim, que muito me ajudaram no desenvolvimento deste trabalho. Entre eles destaco a colega Erika pela parceria nos anos iniciais do doutorado e o colega André pelo apoio na análise estatística.

Ao Departamento de Computação (DCOMP) do Centro de Ciências Exatas, Naturais e da Saúde (CCENS) da UFES, pelo apoio prestado na realização deste trabalho. Aos colegas que me acompanharam nessa jornada e estiveram comigo nas muitas viagens. Um agradecimento especial à Professora Juliana, amiga e companheira de estudo e trabalho.

A todos que contribuíram, direta ou indiretamente, para conclusão dessa etapa da minha vida. Cheguei até aqui porque tive a oportunidade de conhecer, conviver e aprender com pessoas muito especiais.

Muito obrigado!

“Pensava que nós seguíamos caminhos já feitos,
mas parece que não os há. O nosso ir faz o caminho."
(C. S. Lewis)

Resumo

O Problema de Tabela-Horário de Universidades (PTHU) é um dos mais pesquisados dentre os problemas de Otimização Combinatória (OC). Este problema consiste em alocar uma sequência de aulas nas salas disponíveis para um período de tempo predeterminado (tabela-horário) considerando necessidades de alunos, professores e satisfazendo algumas restrições. Várias formulações para o PTHU podem ser encontradas na literatura pois as necessidades que devem ser atendidas na construção da tabela-horário variam para cada instituição. Neste trabalho é considerado o PTHU baseado na formulação de currículos de cursos (CB-CTT) proposta na segunda edição do campeonato internacional de tabela-horário (ITC-2007). Até o momento, diversos métodos baseados em Programação Linear Inteira (PLI) foram propostos na literatura para obter valores de limitantes inferiores desse problema de minimização. Neste trabalho são apresentadas novas estratégias baseadas no princípio *dividir para conquistar* com o intuito de obter valores de limitantes inferiores. As estratégias apresentadas usam a biblioteca *METIS* para particionar o problema original em subproblemas que são resolvidos com o otimizador CPLEX. Experimentos computacionais foram executados nas instâncias de referência usadas no ITC-2007 e os resultados foram comparados com os melhores limitantes inferiores encontrados na literatura. Esses resultados mostram que as estratégias propostas são capazes de melhorar o valor do melhor limitante inferior atualmente conhecido em duas das 21 instâncias e provar a otimalidade em outras 15 instâncias.

Palavras-chaves: Tabela-Horário de Universidades. Limitantes Inferiores. Particionamento. Relaxações.

Abstract

The University Timetabling Problem (UTP) is one of the most researched topics in the field of combinatorial optimization. This problem consists of allocating a set of lectures to available rooms and periods (timetable) considering students and teachers requests and constraints. Several mathematical formulations for the UTP can be found in the literature once specificities of this problem can vary for each institution. The formulation considered in this work is based on courses curricula of an university, proposed in the second International Timetabling Competition (ITC-2007). So far, several methods based on integer linear programming have been proposed in the literature to obtain lower bounds for this minimization problem. Here, new strategies based on the *divide and conquer* principle are presented in order to obtain lower bounds for the problem. In this way, the original problem is partitioned with *METIS* library into sub-problems, each one solved by CPLEX solver. Computational experiments were performed on the ITC-2007 benchmark instances and the results were compared to the best lower bounds found in the literature. These results show the strategies proposed are able to improve the best known lower bounds for two of 21 instances tested and prove the optimality for another 15 instances.

Keywords: University Timetabling. Lower Bounds. Partitioning. Relaxations.

Lista de figuras

Lista de tabelas

Lista de abreviaturas e siglas

API	*Application Programming Interface*
CB-CTT	*Curriculum-Based Course Timetabling*
GRASP	*Greedy Randomized Adaptive Search Procedure*
ITC	*International Timetabling Competition*
OC	Otimização Combinatória
PLI	Programação Linear Inteira
PTHE	Problema de Tabela-Horário de Escolas
PTHU	Problema de Tabela-Horário de Universidades
RFt	Restrições Fortes
RFc	Restrições Fracas

Lista de símbolos

$\mathbb{B}$	Conjunto binário $\{0, 1\}$
$\mathbb{Z}$	Conjunto de números inteiros
C	Conjunto de disciplinas
U	Conjunto de currículos
D	Conjunto de dias
P	Conjunto de períodos
R	Conjunto de salas
T	Conjunto de professores
C_u	Conjunto de disciplinas que pertencem ao currículo $u \in U$
C_t	Conjunto de disciplinas lecionadas pelo professor $t \in T$
P_d	Conjunto de períodos do dia $d \in D$
U_c	Conjunto de currículos no qual a disciplina $c \in C$ pertence
d_p	Dia do período $p \in P$
lct_c	Quantidade de aulas da disciplina $c \in C$
mld_c	Número mínimo de dias de aula da disciplina $c \in C$
st_c	Quantidade de alunos matriculados na disciplina $c \in C$
cap_r	Capacidade da sala $r \in R$
N	Subconjunto de $C \times P$ em que (c, p) representa a disponibilidade da disciplina c ser lecionada no período p
$\overline{N}$	Subconjunto de $C \times P$ em que (c, p) representa a indisponibilidade da disciplina c ser lecionada no período p
N_c	Conjunto de períodos que a disciplina $c \in C$ não possui indisponibilidade
N_p	Conjunto de disciplinas que não possuem indisponibilidade no período $p \in P$
Ψ	Conjunto de todas as diferentes capacidades de sala

$C_{\geq\psi}$	Conjunto de disciplinas que a quantidade de alunos matriculados é maior que $\psi \in \Psi$
$R_{\geq\psi}$	Conjunto de salas que possuem capacidade maior que $\psi \in \Psi$
W^{RC}	Penalidade da restrição fraca **S1** (capacidade de sala)
W^{MD}	Penalidade da restrição fraca **S2** (número mínimo de dias de aula)
W^{CC}	Penalidade da restrição fraca **S3** (aulas isoladas)
W^{RS}	Penalidade da restrição fraca **S4** (estabilidade de sala)

Sumário

1 Introdução

Os problemas de Otimização Combinatória (OC) estão presentes em diversas atividades cotidianas e são comuns em diversas áreas, como exemplo, na criação de rotas de veículos para entrega de encomendas, no planejamento das tarefas a serem executadas por uma indústria e na elaboração de horários escolares.

Esses problemas podem ser representados como a minimização ou maximização de uma função matemática, também denominada função objetivo. As variáveis da função objetivo são denominadas de variáveis de decisão e devem ser estritamente inteiras, além de obedecer a certas restrições. Dessa forma, um problema de OC pode ser matematicamente definido como (GOLDBARG; LUNA, 2005):

$$Minimizar\ (Maximizar)\ f(X)$$

sujeito a

$$g_j(X)\ \{\leqslant, =, \geqslant\}\ b_j \qquad \forall\ j \in \{1, 2, \ldots, m\} \tag{1.1}$$

$$x_i^L\ \leqslant x_i \leqslant x_i^U \qquad \forall\ i \in \{1, 2, \ldots, n\} \tag{1.2}$$

$$x_i \in \mathbb{Z} \qquad \forall\ i \in \{1, 2, \ldots, n\} \tag{1.3}$$

sendo $X = (x_1, x_2, \ldots, x_n)$ o vetor das n variáveis de decisão, $f(X)$ a função objetivo, $g_j(X)$ a j-ésima restrição, podendo esta ser uma desigualdade ou uma igualdade, e b_j que é denominado de termo independente da j-ésima restrição. O número de restrições de desigualdade ou igualdade é m. Os limites das variáveis de decisão são determinados através das expressões (1.2), em que x_i^L é o limite inferior e x_i^U é o limite superior da i-ésima variável de decisão. Por fim, o conjunto de restrições (1.3) garante que as variáveis de decisão assumam apenas valores inteiros.

Segundo Wolsey (1998), se a função objetivo e as restrições de um problema de OC forem lineares, então ele é considerado como um problema de Programação Linear Inteira (PLI).

De forma geral, um problema de PLI, com dimensões reais, possui muitas variáveis de decisão. Por isso, a quantidade de combinações para esses valores é muito grande, e encontrar a melhor combinação de valores que atenda as restrições do problema e de forma rápida é uma tarefa árdua.

Os problemas de escalonamento (*scheduling*) estão entre os mais pesquisados dentre os problemas de OC (BABAEI; KARIMPOUR; HADIDI, 2015), e tratam da alocação de

recursos em determinados horários. Os problemas de tabela-horário são um subconjunto importante dos problemas de *scheduling*, cujas aplicações são variadas, como exemplo, na elaboração das escalas de trabalho de funcionários (DELLA CROCE; SALASSA, 2014), no agendamento de partidas para campeonatos esportivos (RIBEIRO; URRUTIA, 2012) e na elaboração de horários escolares (SANTOS; OCHI; SOUZA, 2005).

No caso específico da tabela-horário educacional, o problema consiste em alocar um conjunto de aulas em um número pré-determinado de dias e horários, satisfazendo restrições pessoais (professores e alunos), do espaço físico disponível e da instituição. A solução manual deste tipo de problema não é uma tarefa simples, já que é difícil contemplar todos os anseios das partes envolvidas. Além disso, as escolas (universidades) precisam resolver o problema anualmente (semestralmente).

Dessa maneira, atenção especial tem sido dada a solução automática de tabela-horário. Começando com o trabalho de Gotlieb (1962), o problema de tabela-horário educacional recebeu grande destaque na área de OC, tendo sido publicados diversos trabalhos desde então, como mostrado nas pesquisas de Schaerf (1999), Lewis (2008), Babaei, Karimpour e Hadidi (2015) e Bettinelli et al. (2015).

1.1 Motivação

Na literatura, diversas formulações para o problema de tabela-horário educacional são apresentadas, já que as restrições variam de acordo com a instituição envolvida. No entanto, todas as formulações podem ser classificadas em uma das três categorias (SCHAERF, 1999):

- **Tabela-horário de Escola** - Programação semanal de aulas para as turmas da escola, de tal forma que professores não lecionem aulas para duas turmas ao mesmo tempo e que turmas não assistam mais de uma aula no mesmo horário.
- **Tabela-horário de Universidade** - Programação semanal das aulas das disciplinas universitárias sem sobreposição de aulas que tenham alunos em comum.
- **Tabela-horário de Exames** - Programação para aplicação dos exames das disciplinas, sem sobreposição de exames das disciplinas que tenham alunos em comum e procurando alocar os exames dos alunos em dias com o maior intervalo possível.

Segundo Schaerf (1999), o Problema de Tabela-Horário de Universidades (PTHU) geralmente tem mais restrições que o Problema de Tabela-Horário de Escolas (PTHE), pois no PTHE as turmas são conjuntos disjuntos de alunos, enquanto que no PTHU um aluno pode estar matriculado em mais de uma disciplina, ou seja, nesse caso é considerada a restrição que as disciplinas com alunos em comum não podem ter aulas alocadas no

mesmo horário. Além disso, no PTHU normalmente as salas e suas capacidades também são consideradas, enquanto no PTHE são normalmente desconsideradas, pois na maioria dos casos se assume que cada turma terá suas aulas em uma única sala, que por sua vez possui capacidade adequada para a quantidade de alunos da turma. Schaerf (1999) afirma também que o PTHU está entre os mais difíceis da área de OC.

O PTHU pode ser classificado NP-completo para a maioria das formulações, uma vez que o problema de coloração de grafos pode ser reduzido em tempo polinomial ao PTHU (SCHAERF, 1999; LEWIS, 2008). Assim, a solução exata só pode ser garantida em tempo computacional viável para instâncias bem pequenas, que não correspondem às instâncias reais da maioria das universidades.

Contudo, mesmo com tempo ilimitado, em alguns casos não é possível encontrar uma solução que atenda todas as restrições consideradas na formulação do PTHU. Por isso, o conjunto de restrições é particionado normalmente em dois grupos: Restrições Fortes (RFt) e Restrições Fracas (RFc).

O conjunto RFt é composto por aquelas restrições que não podem ser violadas. Elas restringem o conjunto de soluções para impedir situações irreais, como exemplo, alunos assistindo mais de uma aula no mesmo horário ou uma sala sendo alocada para mais de uma aula no mesmo horário. Se uma tabela-horário não viola nenhuma restrição forte ela é dita ser uma solução viável do PTHU.

Já o conjunto RFc é composto por aquelas restrições que não interferem na viabilidade da solução, mas refletem certas preferências das instituições. É desejável, por exemplo, que um grupo de alunos não tenha aulas isoladas durante o dia e que aulas da mesma disciplina sejam lecionadas na mesma sala. As restrições fracas podem possuir pesos diferentes para refletir sua importância na qualidade da solução do PTHU.

Dessa forma, o objetivo do PTHU é encontrar uma tabela-horário viável e que minimize a quantidade de violações das restrições do conjunto RFc, sendo assim um problema de minimização.

Levando em consideração que existem diferentes formulações para o PTHU, já que as restrições consideradas variam entre as universidades, uma dificuldade adicional surge ao comparar a qualidade dos resultados obtidos pelos métodos propostos para cada formulação.

Com o intuito de estabelecer uma definição precisa do problema, além de apresentar um conjunto de instâncias para serem usadas como *benchmark* e incentivar a comunidade acadêmica a estudar os problemas de tabela-horário, a *European Metaheuristics Network* (http://www.metaheuristics.net), financiada pela conferência internacional sobre teoria e prática de tabela-horário - PATAT (http://www.patatconference.org), organiza campeonatos internacionais de tabela-horário (*International Timetabling Competition-ITC*). Esses

campeonatos permitem a comparação de performance entre diferentes métodos de solução, já que a formulação e o conjunto de instâncias é igual para todos os competidores.

Em 2002 foi realizada a primeira edição do ITC (ITC-2002) com o objetivo de atrair pesquisadores. O ITC-2002 abordou o PTHU que considera a matrícula de alunos em disciplinas, sendo que cada disciplina é representada por uma única aula (evento) e possui um conjunto de requisitos (exigências) para sua realização. Cada disciplina deve ser alocada em uma sala com capacidade superior ao número de alunos matriculados e que atenda todas as exigências da disciplina. Após o sucesso do ITC-2002, a segunda edição foi realizada em 2007 (ITC-2007).

A inovação do ITC-2007 foi distinguir o problema de tabela-horário em três formulações distintas, cada uma delas com seu próprio conjunto de instâncias. Essas formulações correspondem ao problema de tabela-horário de exames, ao PTHU considerando a solicitação de matrícula dos estudantes (*Post Enrolment based Course Timetabling*, PE-CTT) e ao PTHU considerando os currículos dos cursos (*Curriculum-Based Course Timetabling*, CB-CTT). Além disso, os organizadores do ITC-2007 disponibilizaram dois programas, denominados *benchmark* e *validator*. O programa *benchmark* analisa as configurações do computador com o objetivo de informar o tempo limite de execução do algoritmo, sendo que essa quantidade de tempo é equivalente ao tempo limite de execução do algoritmo nos computadores do campeonato. Já o programa *validator* faz a validação dos resultados obtidos, permitindo assim que os competidores verificassem alguma incongruência nos seus algoritmos. O *website* da competição (http://www.cs.qub.ac.uk/itc2007) apresenta outras informações sobre o ITC-2007.

Por fim, em 2011 foi realizada a terceira edição (ITC-2011), abordando pela primeira vez o problema de tabela-horário de escolas de ensino médio. O objetivo dessa competição foi permitir que pesquisadores experimentem suas técnicas em um conjunto de instâncias extraídas de escolas reais que representam bem o problema que é enfrentado na prática.

Durante o desenvolvimento dessa pesquisa foi lançada a quarta edição da competição (ITC-2019). Essa edição também aborda o PTHU, porém como a previsão de término da competição é o mês de agosto de 2020, neste trabalho o problema abordado foi o PTHU da formulação CB-CTT do ITC-2007.

A formulação CB-CTT foi escolhida dentre as três formulações do ITC-2007 por ser a que mais se aproxima da realidade das universidades brasileiras.

É importante ressaltar que uma grande contribuição do ITC-2007 é a disponibilização de um conjunto de instâncias usadas no campeonato. No caso da formulação CB-CTT, o ITC-2007 disponibilizou 21 instâncias reais da Universidade de Udine, Itália, de diferentes tamanhos, para serem usadas como *benchmark*. Essas instâncias foram divididas em três grupos de sete instâncias cada, sendo que as instâncias do primeiro grupo foram

disponibilizadas no primeiro dia da competição. Duas semanas antes do fim da competição, as instâncias do segundo grupo foram disponibilizadas para os competidores. O último grupo de instâncias só foi disponibilizado após o término da competição, pois foram usadas para classificar os melhores competidores.

Diante disso, pode-se resumir que o ITC-2007 contribuiu de várias formas com a comunidade acadêmica que pesquisa problemas de tabela-horário, mas principalmente com a definição das formulações e a disponibilização das instâncias. Isso pode ser comprovado com vários trabalhos na literatura que propõem métodos de solução para a formulação CB-CTT (BETTINELLI et al., 2015), mesmo após o término da competição.

Dentre os métodos de solução, os mais comuns são os métodos heurísticos, como *Simulated Annealing* (BELLIO et al., 2016), *Greedy Randomized Adaptive Search Procedure* (GRASP) (KAMPKE et al., 2015) e Busca Tabu (LÜ; HAO, 2010).

Além de métodos de solução, alguns trabalhos na literatura apresentam modelos matemáticos e métodos para encontrar limitantes inferiores para o PTHU da formulação CB-CTT. A importância desses métodos está no fato de permitir a avaliação da qualidade das soluções obtidas e até mesmo provar a otimalidade delas. Assim, considerando que as instâncias e a formulação CB-CTT do ITC-2007 são amplamente conhecidas, e que mesmo 13 anos após o término do campeonato a otimilidade das melhores soluções para algumas instâncias ainda não foi provada, surge portanto a motivação para o desenvolvimento deste trabalho.

Por fim, uma outra motivação para o desenvolvimento deste trabalho está no fato que até o momento não foi encontrado na literatura algum método que use a abordagem *dividir para conquistar* com a biblioteca *METIS* para encontrar limitantes inferiores para a formulação CB-CTT, sendo que em outros problemas essa combinação encontrou limitantes de qualidade e de forma rápida (RIBEIRO; LORENA, 2008; MAURI; LORENA, 2012; MAURI, 2019).

1.2 Objetivos

Esta seção é dividida em objetivos gerais (Subseção 1.2.1) e específicos (Subseção 1.2.2).

1.2.1 Objetivos gerais

Propor e analisar novas estatégias para obtenção de limitantes inferiores para a formulação CB-CTT do PTHU proposto no ITC-2007.

1.2.2 Objetivos específicos

1. Identificar e compreender modelos matemáticos para serem usados na obtenção de limitantes inferiores da formulação CB-CTT do PTHU;
2. Detectar relaxações nos modelos matemáticos que forneçam valores de limitantes inferiores de qualidade com baixo tempo computacional;
3. Definir maneiras de dividir o problema original, relaxando restrições, em subproblemas independentes;
4. Implementar métodos que dividem o problema original em subproblemas e os resolver com um método de programação matemática;
5. Identificar meios de tornar as relaxações mais fortes, ou seja, modelar as restrições de forma a ficarem mais próximas das originais e obter melhores valores de limitantes inferiores;
6. Comparar os resultados experimentais dos valores de limitantes inferiores obtidos e a quantidade de tempo usada para obter esses valores, em relação ao que é encontrado atualmente na literatura.

1.3 Principais contribuições

As principais contribuições desta tese estão relacionadas com a apresentação de quatro novos métodos, baseados na abordagem de programação *dividir para conquistar*. Esses métodos usam a biblioteca *METIS* para particionar (dividir) o problema original em problemas menores, relaxando assim algumas restrições. Dessa forma, encontram valores de limitantes inferiores para o PTHU abordado.

Além da apresentação dos métodos, uma outra contribuição está relacionada com o tempo necessário para se encontrar os resultados, já que foi possível encontrar resultados 45,19% mais rápido do que outras abordagens da literatura, como as codificações SAT de Achá e Nieuwenhuis (2014).

Em relação aos valores de limitantes inferiores obtidos pelos métodos, foi possível observar que os resultados foram, em média, melhores que a maioria dos trabalhos da literatura que também apresentam valores de limitantes inferiores para o PTHU da formulação CB-CTT. Além disso, em duas instâncias o valor do melhor limitante inferior encontrado até então foi atualizado.

Por fim, um dos métodos apresentados nesta tese encontrou o valor ótimo em 15 das 21 instâncias, sendo que o trabalho disponível na literatura que provou a otimalidade para o maior número de instâncias encontrou o valor ótimo em 13 instâncias.

1.4 Publicações

Os seguintes artigos e resumos foram publicados durante o desenvolvimento deste trabalho:

- Trabalhos completos publicados em anais de congressos

  - CARVALHO, A. S.; MARIANO, G. P.; KAMPKE, E. H.; MAURI, G. R. Simulated Annealing Aplicado ao Problema de Programação de Horários do CCA-UFES. In: Blucher Marine Engineering Proceedings - XVIII SPOLM - Rio de Janeiro, RJ. CASNAV, 2015. v. 2, n. 1, p. 341-352.
  - SEGATTO, E. A.; BOERES, M. C. S.; RANGEL, M. C.; KAMPKE, E. H. Um Algoritmo GRASP com Cadeia de Kempe Aplicado ao Problema de Tabela-horário para Universidades. In: Anais do XLVII SBPO - Simpósio Brasileiro de Pesquisa Operacional - Porto de Galinhas, PE. SOBRAPO, 2015. p. 2643-2654.
  - KAMPKE, E. H.; ROCHA, W. S.; BOERES, M. C. S.; RANGEL, M. C. A grasp algorithm with path relinking for the university courses timetabling problem. In: Proceeding Series of the Brazilian Society of Computational and Applied Mathematics - CMAC-SE 2015 - Vitória, ES. SBMAC, 2015. v. 3, n. 2, p. 1081–1087.
  - MOREIRA, L. V.; MONTEIRO, R. C.; KAMPKE, E. H.; MAURI, G. R. Meta-heurística GRASP para o Problema de Tabela-horário de Disciplinas do Departamento de Computação do CCA-UFES. In: Anais do XLVIII SBPO - Simpósio Brasileiro de Pesquisa Operacional - Vitória, ES. SOBRAPO, 2016. p. 2171-2182.
  - KAMPKE, E. H.; SEGATTO, E. A.; BOERES, M. C. S.; RANGEL, M. C.; MAURI, G. R. Neighborhood analysis on the university timetabling problem. In: GERVASI, O. et al. (Ed.). Lecture Notes in Computer Science - Computational Science and Its Applications – ICCSA 2017 - Trieste, Italy. Cham: Springer International Publishing, 2017. p. 148–164.
  - MONTEIRO, R. C.; KAMPKE, E. H.; BISSOLI, D. C.; MAURI, G. R. Algoritmo Híbrido Iterated Local Search e Simulated Annealing para o Problema de Tabela-Horário de Universidades. In: Anais do XLIX SBPO - Simpósio Brasileiro de Pesquisa Operacional - Blumenau, SC. SOBRAPO, 2017. p. 1-12.
  - KAMPKE, E. H.; SCHEIDEGER, L. M.; MAURI, G. R.; BOERES, M. C. S. A network flow based construction for a grasp+sa algorithm to solve the university timetabling problem. In: MISRA, S. et al. (Ed.). Lecture Notes in Computer Science - Computational Science and Its Applications – ICCSA 2019

- Saint Petersburg, Russia. Cham: Springer International Publishing, 2019. p. 215–231.

- Resumos expandidos publicados em anais de congressos

  - CARVALHO, A. S.; MARIANO, G. P.; KAMPKE, E. H.; MAURI, G. R. Simulated Annealing e Hill Climbing Aplicado ao Problema de Programação de Horários do CCA-UFES. In: Proceeding Series of the Brazilian Society of Computational and Applied Mathematics - CMAC-SE 2015 - Vitória, ES. SBMAC, 2015. v. 3, n. 2, p. 1201–1202.

  - MONTEIRO, R. C.; KAMPKE, E. H. Heurísticas de Construção no Problema de Tabela-Hórario de Universidades. In: Anais do XXXVII congresso da sociedade brasileira de computação (CSBC)- 2º Encontro de Teoria da Computação (ETC) - São Paulo, SP. SBC, 2017. v. 1, n. 1, p. 158-161.

- Resumos publicados em anais de congressos

  - CARVALHO, A. S.; MARIANO, G. P.; KAMPKE, E. H.; MAURI, G. R. Simulated Annealing e Adaptive Large Neighborhood Search aplicado ao problema de programação de horários do CCA-UFES. In: Anais do XLVII SBPO - Simpósio Brasileiro de Pesquisa Operacional - Porto de Galinhas, PE. SOBRAPO, 2015. p. 3539-3539.

  - SEGATTO, E. A.; BOERES, M. C. S.; RANGEL, M. C.; KAMPKE, E. H. Algoritmos GRASP e VNS para o Problema de Tabela-Horário para Universidades. In: Anais do XLVIII SBPO - Simpósio Brasileiro de Pesquisa Operacional - Vitória, ES. SOBRAPO, 2016. p. 3361-3361.

1.5 Organização do trabalho

Este trabalho está organizado em 6 capítulos, sendo este o primeiro.

No Capítulo 2, o PTHU da formulação CB-CTT proposta no ITC-2007 é descrito em detalhes. As restrições fortes e fracas são apresentadas juntamente com a função objetivo. Nesse capítulo, arquivos texto de uma instância e sua solução são apresentados como exemplo.

O Capítulo 3 apresenta vários trabalhos que também abordam o PTHU da formulação CB-CTT. Aqueles que discorrem sobre modelos matemáticos e métodos para obtenção de limitantes inferiores, e que foram usados como referência nesse trabalho, são descritos e apresentados detalhadamente. Outros trabalhos, que propõem métodos heurísticos de solução do problema, ou seja, para obtenção de limitantes superiores (solução viável), também são apresentados.

Os métodos *U-partition*, *C-partition*, *C-partition-null* e *C-partition-org* para obtenção de limitantes inferiores para o PTHU abordado são descritos detalhadamente no Capítulo 4. Neste capítulo também são apresentadas previamente a definição dos grafos G_U, usado no método *U-partition*, e G_C, usado nos métodos *C-partition*, *C-partition-null* e *C-partition-org*, e a biblioteca *METIS*, usada para particionar os grafos. Ao final, um fluxograma dos métodos *U-partition*, *C-partition*, *C-partition-null* e *C-partition-org* também é exibido.

O Capítulo 5 apresenta os experimentos computacionais, como eles foram avaliados, bem como os resultados obtidos e suas análises. Comparações com resultados da literatura, a fim de avaliar a qualidade dos limitantes inferiores encontrados, também são realizadas.

Por último, no Capítulo 6 são apresentadas as conclusões do trabalho e sugestões de trabalhos futuros.

2 Descrição do problema

O PTHU pode ser definido como a tarefa de alocar as aulas de um conjunto de disciplinas em um número limitado de horários e salas, de acordo com as restrições definidas pela formulação (LEWIS, 2008). Neste trabalho o foco é o PTHU da formulação CB-CTT proposta no ITC-2007, ou seja, o PTHU baseado em currículos de cursos, sendo que um currículo é definido como um conjunto de disciplinas que tem alunos em comum.

Portanto, o PTHU da formulação CB-CTT consiste na alocação de aulas para várias disciplinas em um horário semanal, considerando um número de salas e de períodos de aulas, em que conflitos entre as disciplinas são determinados de acordo com o currículo, e não pela solicitação de matrícula realizada pelos alunos.

A formulação CB-CTT do ITC-2007 é descrita em detalhes por Gaspero, McCollum e Schaerf (2007). As instâncias utilizadas na competição são baseadas em casos reais da Universidade de Udine, Itália.

Resumidamente, cada instância é composta pelos seguintes conjuntos de informações:

- **Dias, Períodos e Horários**: Cada dia possui um número fixo de horários de aula, que é o mesmo para todos os dias. Um período é um par composto por um dia e um horário. O número total de períodos disponíveis para alocação é obtido pela multiplicação do número de dias pelo número de horários em um dia.
- **Disciplinas e Professores**: Cada disciplina possui um número fixo de aulas que devem ser alocadas em períodos distintos, um professor que a leciona e o número de alunos que a assistem. Para cada disciplina há uma quantidade mínima de dias distintos em que suas aulas devem ser alocadas.
- **Salas**: Cada sala possui uma capacidade definida pelo número de assentos disponíveis.
- **Currículos**: Consiste em um conjunto de disciplinas que possuem alunos em comum.
- **Indisponibilidades**: Períodos em que uma determinada disciplina não pode ser lecionada.

A partir dessas informações são definidas as restrições fortes e fracas do PTHU da formulação CB-CTT. É importante lembrar que as restrições fortes devem ser sempre respeitadas. Qualquer violação de uma restrição forte gera uma tabela-horário inviável, que na prática não pode ser utilizada. Por outro lado, as restrições fracas devem ser satisfeitas o máximo possível e, quanto menos violações, melhor é a tabela-horário. A formulação

CB-CTT do ITC-2007 define oito restrições no total, sendo quatro restrições fortes e quatro restrições fracas. As restrições são descritas a seguir:

- Restrições Fortes:
  - **H1**-Aulas: Todas as aulas de uma disciplina devem ser alocadas, e em períodos diferentes. Uma violação ocorre se uma aula não é alocada, ou se duas aulas da mesma disciplina são alocadas no mesmo período.
  - **H2**-Ocupação de Sala: Duas aulas não podem ser alocadas em uma mesma sala e no mesmo período. Cada aula extra em uma mesma sala e no mesmo período conta como uma violação.
  - **H3**-Conflitos: Aulas de disciplinas de um mesmo currículo, ou ministradas pelo mesmo professor devem ser alocadas em períodos diferentes. Duas aulas conflitantes no mesmo período representa uma violação.
  - **H4**-Indisponibilidade: Se o professor de uma disciplina não está disponível para lecioná-la em determinado período, nenhuma aula dessa disciplina pode ser alocada nesse período. Cada aula alocada em um período não disponível para a disciplina conta como uma violação.
- Restrições Fracas:
  - **S1**-Capacidade de Sala: Para cada disciplina, o número de alunos que está matriculado na disciplina deve ser menor ou igual ao número de assentos disponíveis em todas as salas que ocorrem aulas dessa disciplina. Cada aluno acima da capacidade conta como uma violação.
  - **S2**-Número Mínimo de Dias de Aula: As aulas de uma disciplina devem ser espalhadas em um número mínimo de dias. Cada dia abaixo do número mínimo de dias conta como uma violação.
  - **S3**-Aulas Isoladas: Aulas de disciplinas de um mesmo currículo devem ser adjacentes uma à outra. Para cada currículo, uma violação é contada quando há uma aula não adjacente à nenhuma outra aula do mesmo currículo no mesmo dia.
  - **S4**-Estabilidade de Sala: Todas as aulas de uma disciplina devem acontecer na mesma sala. Cada sala distinta usada para aulas dessa disciplina, além da primeira, contam como uma violação.

A qualidade de uma solução viável, ou seja, uma solução que não viola nenhuma restrição forte (**H1**-**H4**), depende da quantidade de violações das restrições fracas (**S1**-**S4**), sendo que cada restrição fraca possui um peso associado.

Portanto, o objetivo do PTHU da formulação CB-CTT é encontrar uma tabela-horário viável que minimiza a soma ponderada das violações das restrições fracas. Considerando Ω o conjunto de todas as soluções viáveis, para cada $\chi \in \Omega$, o valor da função objetivo f é definido por:

$$f(\chi) = \sum_{i=1}^{4} \alpha_i \cdot \beta_i(\chi)$$

sendo que $(\alpha_1,\alpha_2,\alpha_3,\alpha_4) = (1,5,2,1)$ representa os pesos atribuídos, respectivamente, a cada uma das restrições fracas **S1**-**S4**. Os valores dos pesos foram definidos pelos organizadores do ITC-2007 (GASPERO; MCCOLLUM; SCHAERF, 2007). $\beta_i(\chi)$ representa o número total de violações de cada restrição fraca i na solução viável $\chi \in \Omega$.

Assim, pode-se dizer então, que formalmente o objetivo do PTHU da formulação CB-CTT é encontrar uma solução viável $\chi^* \in \Omega$, de tal forma, que para todo $\chi \in \Omega$, $f(\chi^*) \leq f(\chi)$.

A partir disso, o ITC-2007 definiu um conjunto de 21 instâncias reais (comp01-comp21), de diferentes tamanhos, para permitir a comparação de diferentes métodos de solução do PTHU da formulação CB-CTT. Esse conjunto foi proposto por Gaspero, McCollum e Schaerf (2007) e está disponível para *download* no *website* da competição (http://www.cs.qub.ac.uk/itc2007).

Gaspero, McCollum e Schaerf (2007) também apresentam informações importantes sobre cada instância, como o número de disciplinas (30-131), o número de salas (5-20), o número de currículos (13-150) e também o número total de aulas a serem alocadas (138-434). Apesar dessas informações ajudarem a medir a dificuldade para obter uma solução viável em uma instância, os autores também apresentam outras medidas, que relacionam essas informações e proporcionam assim uma análise mais precisa sobre o tamanho de uma instância e a respectiva dificuldade para obter uma solução viável. Dentre essas medidas destaca-se a média de conflitos entre aulas e a média da quantidade de aulas por currículo por dia.

O padrão do arquivo, definido pelo ITC-2007, que representa uma instância da formulação CB-CTT, é ilustrado na Figura 1, fornecido pela própria organização do campeonato.

O arquivo pode ser dividido em cinco partes: cabeçalho, disciplinas, salas, currículos e indisponibilidades. O cabeçalho é composto por sete linhas, cada uma contendo, respectivamente, o nome da instância, o número de disciplinas, salas, dias, períodos por dia, currículos e indisponibilidades. Na parte das disciplinas, cada linha contém o nome de uma disciplina, nome do professor que a leciona, quantidade de aulas na semana, número mínimo de dias semanais de aula e quantidade de alunos. A terceira parte apresenta informações a

```
Name: Toy
Courses: 4
Rooms: 3
Days: 5
Periods_per_day: 4
Curricula: 2
Constraints: 8

COURSES:
SceCosC Ocra 3 3 30
ArcTec Indaco 4 2 42
TecCos Rosa 3 4 40
GeoTec Scarlatti 3 4 18

ROOMS:
rA 32
rB 50
rC 40

CURRICULA:
Cur1 3 SceCosC ArcTec TecCos
Cur2 2 TecCos GeoTec

UNAVAILABILITY_CONSTRAINTS:
TecCos 2 0
TecCos 2 1
TecCos 3 2
TecCos 3 3
ArcTec 4 0
ArcTec 4 1
ArcTec 4 2
ArcTec 4 3

END.
```

Figura 1 – Arquivo de exemplo - Instância *Toy*.

respeito das salas. Cada linha representa uma sala e possui o seu nome seguido de sua capacidade (número de assentos). As informações sobre os currículos compõem a penúltima parte. Cada currículo é representado por uma linha contendo o seu nome, o número de disciplinas presentes nele seguido dos nomes de cada disciplina. Por último, são listadas as indisponibilidades das disciplinas, uma por linha, contendo o nome da disciplina, o dia e horário em que a disciplina não pode ser lecionada. Dessa forma, na Figura 1 a primeira linha dessa última parte indica que a disciplina *TecCos* não poderá ter aula alocada no primeiro horário (*horário 0*) do terceiro dia (*dia 2*).

A Figura 2 mostra um exemplo de um arquivo de resposta, ou seja, contendo uma

possível solução da instância *Toy* que foi apresentada na Figura 1. O arquivo deve conter, para cada aula alocada, uma linha contendo o nome da disciplina, o nome da sala, o dia e o horário do dia em que está alocada. Por exemplo, a primeira linha do arquivo apresentado na Figura 2 representa que uma aula da disciplina *GeoTec* foi alocada na sala *rA* no segundo horário (*horário 1*) do segundo dia (*dia 1*).

```
GeoTec rA 1 1
GeoTec rA 2 3
ArcTec rB 2 1
ArcTec rB 1 1
GeoTec rA 0 0
TecCos rB 2 2
ArcTec rB 1 3
SceCosC rB 0 0
ArcTec rB 3 0
SceCosC rB 1 2
TecCos rB 0 1
TecCos rB 1 0
SceCosC rB 3 1
```

Figura 2 – Arquivo de resposta - Instância *Toy*.

3 Trabalhos Correlatos

O PTHU tem sido amplamente estudado pela comunidade acadêmica devido a sua importância teórica e prática. É possível encontrar diversos trabalhos na literatura com diferentes abordagens para o problema. No entanto, essas abordagens tem recaído basicamente em dois grupos: métodos de programação matemática e métodos heurísticos (BETTINELLI et al., 2015).

Os métodos de programação matemática tem como principal objetivo encontrar a solução ótima do PTHU a partir de um modelo matemático. Esses métodos também são denominados *exatos* (BAZARAA; JARVIS; SHERALI, 2009). Devido à complexidade do PTHU, grande parte dos métodos exatos precisa de uma grande quantidade de tempo para encontrar a solução ótima. Assim, alguns desses métodos relaxam (ignoram) algumas restrições para encontrar uma solução do problema mais rapidamente (MAURI, 2008). O valor da solução obtida nesses casos representa um limitante inferior (superior) do problema de minimização (maximização). No caso da formulação CB-CTT do PTHU, por ser um problema de minimização, o valor de um limitante inferior (*Lower Bound* - LB), para uma instância, indica que uma solução viável daquela instância não possui valor de função objetivo menor que LB (GOLDBARG; LUNA, 2005).

Em contrapartida, o objetivo dos métodos heurísticos é encontrar uma solução viável e de qualidade para o PTHU de forma eficiente, ou seja, com pouco tempo computacional. Nos métodos heurísticos não é possível assegurar que a solução encontrada é a ótima (MAURI, 2008). No caso da formulação CB-CTT do PTHU, por ser um problema de minimização, o valor da função objetivo de uma solução viável, obtida por um método heurístico, representa um limitante superior (*Upper Bound* - UB). Dessa maneira, quanto menor for a distância entre LB e UB, melhor será a qualidade da solução obtida pelo método heurístico. Se os valores forem iguais, isso representa que a otimalidade foi provada, ou seja, que a solução obtida pelo método heurístico é a ótima (GOLDBARG; LUNA, 2005).

Neste capítulo é apresentada apenas uma parte dos trabalhos que abordam o PTHU, em especial aqueles que abordam a formulação CB-CTT do ITC-2007. Assim, a Seção 3.1 é dedicada a trabalhos que apresentam modelos matemáticos e métodos para obtenção de limitantes inferiores, principalmente os trabalhos cujos resultados são usados como referência. Já a Seção 3.2 apresenta trabalhos que propõem métodos heurísticos de solução para o problema.

É importante destacar que neste trabalho são citados alguns conceitos da *Teoria dos Grafos*. Todos esses conceitos podem ser encontrados em Diestel (2016). A seguir são

apresentadas apenas as definições dos principais conceitos citados neste trabalho.

Definição 1 (*Grafo Bipartido*): O grafo $G = (V, A)$ é denominado bipartido se o conjunto de vértices V pode ser particionado em dois subconjuntos X e Y, de tal forma que $V = X \cup Y$, $X \cap Y = \emptyset$ e para toda aresta $\{a, b\} \in A$, $a \in X$ e $b \in Y$.

Definição 2 (*Acoplamento*): Um subconjunto de arestas M do grafo $G = (V, A)$, $M \subseteq A$, representa um acoplamento em G, se, e somente se, existe o subconjunto X de vértices, $X \subseteq V$, de forma que todo vértice em X é incidente a uma, e apenas uma, aresta de M.

Definição 3 (*Acoplamento completo em um grafo bipartido*): Um acoplamento M do grafo bipartido $G = (X \cup Y, A)$ é denominado *completo* se todos os vértices de X, considerando $|X| \leq |Y|$, sejam incidentes a todas as arestas de M.

Definição 4 (*Acoplamento Maximal*): Considerando M um acoplamento no grafo $G = (V, A)$, M é maximal se não existe aresta $a \in A$, $a \notin M$, tal que $M + \{a\}$ também seja um acoplamento.

Definição 5 (*Grafo Conexo*): O grafo $G = (V, A)$ é denominado conexo se todo par de vértices $a \in V$ e $b \in V$ são conectados por um caminho em G.

Definição 6 (*Componente Conexa*): Seja $G = (V, A)$ um grafo não vazio. Um subgrafo é maximal com respeito à conexidade se não existe outro subgrafo em G no qual ele esteja contido e que seja conexo.

Definição 7 (*Clique*): Um *clique* no grafo $G = (V, A)$ é um conjunto V' de vértices, tal que $V' \subseteq V$, $V' \neq \emptyset$, e para todo par v'_i e v'_j de vértices distintos em V' tem se $\{v'_i, v'_j\} \in A$.

3.1 Modelos matemáticos e limitantes

Existem diversos trabalhos na literatura que apresentam modelos matemáticos para a formulação CB-CTT. Em alguns deles são apresentados métodos para obtenção de limitantes inferiores. Nesta seção são apresentados apenas os dois principais modelos matemáticos para a formulação CB-CTT. A escolha desses modelos ocorreu pelo fato dos demais modelos propostos, que não são apresentados neste trabalho, serem baseados nesses dois.

Os valores de limitantes inferiores obtidos com base nesses modelos matemáticos ajudam a avaliar a qualidade de soluções de métodos heurísticos. No caso de problemas muito complexos, ou seja, de difícil solução por métodos exatos, como o PTHU, a im-

portância de encontrar bons valores de limitantes, e até mesmo provar a otimalidade de algumas instâncias, é ainda maior.

Os primeiros trabalhos que abordam métodos de programação matemática, para o PTHU da formulação CB-CTT, apresentam modelos matemáticos com muitas variáveis e restrições (BURKE et al., 2010), além de métodos exatos clássicos como *Branch-and-cut* (BURKE et al., 2012). Posteriormente, novos modelos, com menos variáveis e restrições, são apresentados (LACH; LÜBBECKE, 2012). Métodos que relaxam restrições e dividem o problema também são apresentados na literatura (HAO; BENLIC, 2011). Alguns trabalhos mais recentes usam abordagens pouco conhecidas até então, como exemplo a representação do problema com dois modelos matemáticos que são conectados por uma rede de fluxo (BAGGER et al., 2019).

A seguir são apresentados em ordem cronológica alguns trabalhos na literatura que propõem modelos matemáticos e métodos de obtenção de limitantes inferiores para a formulação CB-CTT do PTHU. Nesses trabalhos são apresentados os resultados obtidos quando o método proposto em cada um deles foi executado no conjunto de instâncias do ITC-2007. A escolha desses trabalhos levou em consideração a contribuição de cada um deles, seja provando a otimalidade de algumas instâncias ou melhorando os valores de limitantes inferiores conhecidos até então.

3.1.1 Modelo compacto de Burke et al. (2010) e limitantes inferiores

No trabalho de Burke et al. (2010) é apresentado um modelo matemático para a formulação CB-CTT do ITC-2007. O modelo apresentado é denominado Monolítico pois representa na íntegra a formulação CB-CTT, sendo portanto usado para obtenção de limitantes inferiores e superiores para o problema. O modelo Monolítico representa um modelo de PLI, podendo assim ser resolvido por um método de programação matemática para obtenção da solução ótima do problema.

Neste modelo, cinco conjuntos de variáveis de decisão são definidos:

- $x_{p,r,c} \in \mathbb{B}$ é igual a 1 se a disciplina $c \in C$ possuir uma aula alocada no período $p \in P$ e na sala $r \in R$ e 0 caso contrário;
- $v_{d,c} \in \mathbb{B}$ é igual a 1 se a disciplina $c \in C$ possuir pelo menos uma aula alocada no dia $d \in D$ e 0 caso contrário;
- $q_c \in \mathbb{Z}$ representa a diferença entre o número mínimo de dias de aula da disciplina $c \in C$ (mld_c) e o número de dias distintos que a disciplina c possui aulas alocadas, caso a diferença seja positiva e 0 caso contrário;
- $z_{u,p} \in \mathbb{B}$ é igual a 1 se há uma aula isolada do currículo $u \in U$ alocada no período $p \in P$ e 0 caso contrário;

- $y_{r,c} \in \mathbb{B}$ é igual a 1 se há pelo menos uma aula da disciplina $c \in C$ alocada na sala $r \in R$ e 0 caso contrário.

O modelo Monolítico proposto por Burke et al. (2010) é apresentado a seguir:

$$min\ W^{RC} \sum_{r \in R} \sum_{p \in P} \sum_{\substack{c \in C \\ st_c > cap_r}} x_{p,r,c}(st_c - cap_r) + W^{MD} \sum_{c \in C} q_c$$
$$+ W^{CC} \sum_{u \in U} \sum_{p \in P} z_{u,p} + W^{RS} \sum_{c \in C} ((\sum_{r \in R} y_{r,c}) - 1)$$

sujeito a

$$\sum_{p \in P} \sum_{r \in R} x_{p,r,c} = lct_c \qquad \forall c \in C \qquad (3.1)$$
$$\sum_{c \in C} x_{p,r,c} \leqslant 1 \qquad \forall p \in P,\ r \in R \qquad (3.2)$$
$$\sum_{r \in R} x_{p,r,c} \leqslant 1 \qquad \forall p \in P,\ c \in C \qquad (3.3)$$
$$\sum_{r \in R} \sum_{c \in C_t} x_{p,r,c} \leqslant 1 \qquad \forall p \in P,\ t \in T \qquad (3.4)$$
$$\sum_{r \in R} \sum_{c \in C_u} x_{p,r,c} \leqslant 1 \qquad \forall p \in P,\ u \in U \qquad (3.5)$$
$$\sum_{r \in R} x_{p,r,c} = 0 \qquad \forall (c,p) \in \overline{N} \qquad (3.6)$$
$$\sum_{r \in R} x_{p,r,c} \leqslant v_{d_p,c} \qquad \forall c \in C,\ p \in P \qquad (3.7)$$
$$\sum_{r \in R} \sum_{p \in P_d} x_{p,r,c} \geqslant v_{d,c} \qquad \forall c \in C,\ d \in D \qquad (3.8)$$
$$\sum_{d \in D} v_{d,c} \geqslant mld_c - q_c \qquad \forall c \in C \qquad (3.9)$$
$$\sum_{c \in C_u} \sum_{r \in R} (x_{p,r,c} - x_{p-1,r,c} - x_{p+1,r,c}) \leqslant z_{u,p} \qquad \forall u \in U,\ p \in P \qquad (3.10)$$
$$x_{p,r,c} \leqslant y_{r,c} \qquad \forall p \in P,\ r \in R,\ c \in C \qquad (3.11)$$
$$\sum_{p \in P} x_{p,r,c} \geqslant y_{r,c} \qquad \forall r \in R,\ c \in C \qquad (3.12)$$
$$x_{p,r,c} \in \mathbb{B} \qquad \forall p \in p,\ r \in R,\ c \in C \qquad (3.13)$$
$$v_{d,c} \in \mathbb{B} \qquad \forall d \in D,\ c \in C \qquad (3.14)$$
$$q_c \in \mathbb{Z} \qquad \forall c \in C \qquad (3.15)$$
$$z_{u,p} \in \mathbb{B} \qquad \forall u \in U,\ p \in P \qquad (3.16)$$
$$y_{r,c} \in \mathbb{B} \qquad \forall r \in R,\ c \in C \qquad (3.17)$$

A função objetivo visa minimizar a violação das restrições fracas, uma vez que as restrições fortes são modeladas nos conjuntos de restrições (3.1)−(3.6), e garantem que todas as aulas da disciplina $c \in C$ sejam alocadas (3.1), que não exista mais de

uma aula alocada na mesma sala e no mesmo período (3.2), que uma disciplina tenha no máximo uma aula alocada por período (3.3), que disciplinas lecionadas pelo mesmo professor (3.4) ou do mesmo currículo (3.5) não sejam alocadas no mesmo período, e por fim, que uma disciplina não tenha uma das suas aulas alocadas em um período que possui indisponibilidade (3.6).

Os conjuntos de restrições (3.7)−(3.12) são usados para representar as restrições fracas. Dessa forma, (3.7) e (3.8) garantem que as variáveis $v_{d,c}$ assumem valores iguais a 1 se, e somente se, a disciplina $c \in C$ possuir uma aula alocada no dia $d \in D$. Os valores das variáveis q_c são definidos a partir dos valores das variáveis $v_{d,c}$ e isso é garantido em (3.9). O conjunto de restrições (3.10) modela a restrição fraca **S3** (aulas isoladas), mas vale destacar que caso o período $p \in P$ seja o primeiro ou último do dia $d \in D$, então as variáveis $x_{p-1,r,c}$ e $x_{p+1,r,c}$, respectivamente, terão valor 0, uma vez que não existem. A restrição fraca **S4** (estabilidade de sala) é modelada pelos conjuntos de restrições (3.11) e (3.12). Por último, os conjuntos de restrições (3.13)−(3.17) definem o domínio das variáveis de decisão.

Burke et al. (2010) apresentam os resultados da execução do modelo Monolítico pelo otimizador CPLEX (IBM ILOG, 2017) com o tempo de execução limitado em 31200 segundos (40 × 780), sendo 780 segundos o valor calculado pelo programa *benchmark* no computador usado para os experimentos computacionais. O programa *benchmark*, disponibilizado pela organização do ITC-2007, analisa as configurações do computador com o objetivo de informar a quantidade de tempo que seria equivalente ao dos computadores do campeonato. O valor do *multiplicador* 40 foi definido pelos autores sem apresentar uma justificativa para essa escolha. A maior parte dos demais trabalhos apresentados nesta seção também usam esse critério para definir o tempo limite dos seus métodos.

Dessa forma, Burke et al. (2010) encontraram, no tempo estipulado (31200 segundos), valores de limitantes inferiores e superiores para as 14 instâncias do problema disponibilizadas pela organização do ITC-2007 até aquele momento. Em apenas uma instância os valores de limitantes (inferior e superior) foram iguais, ou seja, foi possível provar a otimalidade apenas dessa instância.

Nesse mesmo trabalho, Burke et al. (2010) propõem a alteração do modelo Monolítico de duas maneiras diferentes, através de relaxações, derivando assim os modelos simplificados *Surface1* e *Surface2*. Esses novos modelos são denominados assim pois, diferentemente do modelo Monolítico, encontram apenas valores de limitantes inferiores.

O modelo *Surface1* ignora as penalidades das violações das restrições **S1** (capacidade de sala) e **S4** (estabilidade de sala), ou seja, $W^{RC} = 0$ e $W^{RS} = 0$, além de adicionar restrições adicionais para limitar o número de salas usadas em cada período. Essas relaxações foram definidas dessa maneira pois assim o número de variáveis e restrições nesse novo modelo é muito menor do que no modelo Monolítico. Já no modelo *Surface2* a

idéia é usar o conceito de multi-salas, ou seja, considerar que todas as $|R|$ salas possuem a capacidade igual à capacidade da maior sala disponível inicialmente. Posteriormente, foi considerada a existência de duas multi-salas, sendo que em uma multi-sala foram agrupadas as salas com capacidade maior que um limite e na outra multi-sala, as demais. Em ambos os casos, as quatro penalidades da função objetivo são consideradas e o número de variáveis e restrições permanece menor que no modelo Monolítico.

Os resultados apresentados por Burke et al. (2010) para 14 instâncias mostram que o modelo *Surface2*, também com tempo de execução limitado em 31200 segundos, apresentou, em média, melhores valores de limitantes inferiores do que os modelos Monolítico e *Surface1*, tendo provado a otimalidade de 6 instâncias, enquanto o Monolítico e *Surface1* provaram a otimalidade de 1 e 4 instâncias, respectivamente.

3.1.2 *Branch-and-cut* de Burke et al. (2012)

No trabalho de Burke et al. (2012) é proposto um algoritmo *branch-and-cut* para o problema. Inicialmente os autores sugerem uma relaxação no modelo Monolítico, mais precisamente no conjunto de restrições (3.10), que modelam a restrição fraca **S3** (aulas isoladas). Esse conjunto de restrições foi escolhido para ser relaxado pois o número de restrições, bem como a sua dificuldade, está diretamente relacionado ao número de currículos e a quantidade de disciplinas que cada um deles possui, sendo portanto um conjunto de restrições que dificulta a alocação das aulas.

Ao novo modelo, denominado Monolítico Relaxado, é adicionado um conjunto de restrições. Esse conjunto de restrições representa um *corte* (GOLDBARG; LUNA, 2005). No trabalho de Burke et al. (2012) esse *corte* no modelo Monolítico Relaxado foi denominado *Tipo 1*. Um segundo corte (*Tipo 2*) também é proposto e possui três conjuntos de restrições a serem adicionadas.

Os valores de limitantes inferiores obtidos pelo algoritmo *branch-and-cut*, proposto por Burke et al. (2012) e aplicado ao modelo Monolítico Relaxado com os dois tipos de corte, foi pior do que os resultados obtidos pelos modelos Monolítico, *Surface1* e *Surface2* (BURKE et al., 2010). Os autores também apresentam os resultados obtidos ao resolverem o modelo *Surface1* com a adição de um dos três conjuntos de restrições propostas no corte *Tipo 2*. Esses resultados são, em média, melhores que os resultados do *branch-and-cut* aplicado ao modelo Monolítico Relaxado com os dois tipos de corte, porém também são piores do que os valores obtidos pelos modelos Monolítico, *Surface1* e *Surface2* (BURKE et al., 2010).

Apesar dos resultados apresentados por Burke et al. (2012) terem sido, em média, piores que os apresentados por Burke et al. (2010), em três instâncias o melhor valor de limitante inferior, conhecido até aquele momento, foi atualizado.

3.1.3 Modelo de duas fases de Lach e Lübbecke (2012)

Lach e Lübbecke (2012) também propuseram um modelo matemático de PLI para resolver a formulação CB-CTT. O trabalho que apresenta o modelo proposto foi publicado em 2012, mas teve sua primeira versão disponibilizada *online* em 2010.

O modelo de Lach e Lübbecke (2012) possui menos variáveis e restrições que o Monolítico proposto por Burke et al. (2010). Além disso, é decomposto em dois modelos menores de PLI, cada um representando uma fase da resolução do problema. Na primeira fase as aulas das disciplinas são alocadas em um período de tempo sem levar em consideração as salas. Existe apenas um conjunto de restrições que limita o número de aulas alocadas em cada período $p \in P$ à quantidade de salas disponíveis $|R|$. Com esse conjunto de restrições é possível calcular a quantidade de violações da restrição fraca **S1** (capacidade de sala). Dessa forma, na primeira fase, além da restrição fraca **S1**, a resolução do modelo também tem por objetivo minimizar as restrições fracas **S2** (número mínimo de dias de aula) e **S3** (aulas isoladas). Na segunda fase, com a solução obtida na resolução da primeira fase, ou seja, de acordo com a quantidade de violações da restrição fraca **S1** (capacidade de sala), as aulas são então alocadas nas salas, considerando o objetivo de minimizar a restrição fraca **S4** (estabilidade de sala).

No modelo de PLI da primeira fase, além dos conjuntos de variáveis $v_{d,c}$, q_c e $z_{u,p}$ já definidas anteriormente, também são usados outros três conjuntos:

- $x_{p,c} \in \mathbb{B}$ é igual a 1 se a disciplina $c \in C$ possuir uma aula alocada no período $p \in P$ e 0 caso contrário;
- $\omega_{p,\psi,c} \in \mathbb{B}$ é igual a 1 se a disciplina $c \in C$ possuir uma aula alocada no período $p \in P$ em uma sala com capacidade menor que $\psi \in \Psi$ e 0 caso contrário;
- $\varphi_{u,p} \in \mathbb{B}$ é igual a 1 se há uma aula isolada do currículo $u \in U$ alocada no período $p \in P$ e 0 caso contrário;

É necessário destacar que as variáveis $x_{p,c}$ e $\omega_{p,\psi,c}$ só são consideradas no modelo se a disciplina $c \in C$ não possuir indisponibilidade no período $p \in P$, ou seja, se $(c, p) \notin \overline{N}$. Além disso, a variável $\omega_{p,\psi,c}$ também só é considerada se a quantidade de alunos matriculados na disciplina $c \in C$ for maior ou igual a ψ ($st_c \geq \psi$).

O modelo matemático de PLI da primeira fase proposto por Lach e Lübbecke (2012) é apresentado a seguir:

$$min\ W^{RC} \sum_{\psi \in \Psi} \sum_{p \in P} \sum_{\substack{c \in C_{\geq \psi}, \\ c \in \overline{N}_p}} \omega_{p,\psi,c} \cdot \theta_{p,\psi,c} + W^{MD} \sum_{c \in C} q_c + W^{CC} \sum_{u \in U} \sum_{p \in P} z_{u,p}$$

sujeito a

$$\sum_{p \in N_c} x_{p,c} = lct_c \qquad \forall c \in C \qquad (3.18)$$

$$\sum_{c \in N_p} x_{p,c} \leqslant |R| \qquad \forall p \in P \qquad (3.19)$$

$$x_{p,c} \geqslant \omega_{p,\psi,c} \qquad \forall \psi \in \Psi,\ c \in C_{\geq \psi},\ (c,p) \in N \qquad (3.20)$$

$$\sum_{\substack{c \in C_{\geq \psi}, \\ c \in N_p}} (x_{p,c} - \omega_{p,\psi,c}) \leqslant |R_{\geq \psi}| \qquad \forall p \in P,\ \psi \in \Psi \qquad (3.21)$$

$$\sum_{\substack{p \in P_d, \\ p \in N_c}} x_{p,c} \geqslant v_{d,c} \qquad \forall d \in D,\ c \in C \qquad (3.22)$$

$$\sum_{d \in D} v_{d,c} \geqslant mld_c - q_c \qquad \forall c \in C \qquad (3.23)$$

$$\sum_{\substack{c \in C_u, \\ c \in N_p}} x_{p,c} = \varphi_{u,p} \qquad \forall u \in U,\ p \in P \qquad (3.24)$$

$$-\varphi_{u,p-1} + \varphi_{u,p} - \varphi_{u,p+1} \leqslant z_{u,p} \qquad \forall u \in U,\ p \in P \qquad (3.25)$$

$$\sum_{\substack{c \in C_t, \\ c \in N_p}} x_{p,c} \leqslant 1 \qquad \forall p \in P,\ t \in T \qquad (3.26)$$

$$x_{p,c} \in \mathbb{B} \qquad \forall (c,p) \in N \qquad (3.27)$$

$$\omega_{p,\psi,c} \in \mathbb{B} \qquad \forall \psi \in \Psi,\ c \in C_{\geq \psi},\ (c,p) \in N \qquad (3.28)$$

$$v_{d,c} \in \mathbb{B} \qquad \forall d \in D,\ c \in C \qquad (3.29)$$

$$q_c \in \mathbb{Z} \qquad \forall c \in C \qquad (3.30)$$

$$z_{u,p} \in \mathbb{B} \qquad \forall u \in U,\ p \in P \qquad (3.31)$$

$$\varphi_{u,p} \in \mathbb{B} \qquad \forall u \in U,\ p \in P \qquad (3.32)$$

A função objetivo do modelo possui três termos, representando respectivamente a violação das restrições fracas **S1**, **S2** e **S3**, uma vez que a restrição **S4** é ignorada. Vale destacar que o valor de $\theta_{p,\psi,c}$, apresentado pela primeira vez na função objetivo do modelo proposto, é definido por:

$$\theta_{p,\psi_i,c} = min\{st_c - \psi_i, \psi_{i+1} - \psi_i\} \qquad (3.33)$$

O valor de $\theta_{p,\psi_i,c}$ (3.33) é definido dessa maneira para que a aula da disciplina $c \in C$ lecionada no período $p \in P$, caso não possa ser alocada em uma sala com capacidade maior que o número de alunos matriculados, seja alocada preferencialmente em uma sala cuja capacidade é inferior, porém o mais próximo possível do número de alunos matriculados.

O conjunto de restrições (3.18) garante que todas as aulas da disciplina $c \in C$ sejam alocadas em algum período e o conjunto de restrições (3.19) assegura que no máximo $|R|$ aulas podem ser alocadas no período $p \in P$. O correto relacionamento entre

as variáveis de decisão $x_{p,c}$ e $\omega_{p,\psi,c}$ é garantido pelos conjuntos de restrições (3.20) e (3.21). Esse correto relacionamento permite que os valores das variáveis de decisão $\omega_{p,\psi,c}$ sejam definidos de acordo com os valores das variáveis $x_{p,c}$. O valor das variáveis de decisão $v_{d,c}$ é definido no conjunto de restrições (3.22), sendo que essas variáveis são usadas no conjunto de restrições (3.23) para definir a penalidade q_c de **S2** (número mínimo de dias de aula) para cada disciplina $c \in C$. O conjunto de restrições (3.24) garante o correto relacionamento entre as variáveis de decisão $x_{p,c}$ e $\varphi_{u,p}$, e assim a penalidade $z_{u,p}$ de **S3** (aulas isoladas) é definida no conjunto de restrições (3.25). Como as disciplinas lecionadas pelo mesmo professor não podem ter suas aulas alocadas no mesmo período, o conjunto de restrições (3.26) garante esta exigência. Por fim, os conjuntos de restrições (3.27)−(3.32) definem o domínio das variáveis de decisão.

O modelo matemático de PLI da segunda fase representa a formulação padrão do problema de acoplamento completo de custo mínimo em um grafo bipartido $G = (X \cup Y, A)$ (AHUJA; MAGNANTI; ORLIN, 1993).

No grafo $G = (X \cup Y, A)$, cada variável $x_{p,c}$ que, após a resolução do modelo da primeira fase tiver valor igual a 1, será representada por um vértice do conjunto X. O conjunto de vértices Y possui $|P \times R|$ vértices representando todas as combinações de salas e períodos. A aresta $\{\mu_{p,c},\upsilon_{p,r}\}$, sendo $\mu_{p,c} \in X$ e $\upsilon_{p,r} \in Y$, pertence ao conjunto A se $\omega_{p,\psi,c}$ é igual a 1, ou se $\omega_{p,\psi,c}$ é igual a 0 e a capacidade da sala r é maior ou igual ao número de alunos matriculados em c. Nesse modelo é adicionado um conjunto de restrições que modela a restrição fraca **S4** (estabilidade de sala).

Lach e Lübbecke (2012) resolveram o modelo da primeira fase no otimizador CPLEX com tempo limite de 13000 segundos (40×325), sendo 325 segundos o valor informado pelo programa *benchmark*, disponibilizado pela organização do ITC-2007, no computador usado para os experimentos computacionais. Os valores obtidos são limitantes inferiores válidos e mesmo obtendo, em média, resultados que não superam os valores obtidos na resolução do modelo *Surface2* por Burke et al. (2012), a otimalidade foi provada em 4 das 14 instâncias usadas nos testes, e em outras duas o valor do limitante inferior obtido foi melhor do que o conhecido até aquele momento.

3.1.4 Abordagem *dividir para conquistar* de Hao e Benlic (2011)

O trabalho de Lach e Lübbecke (2012), por ter sido disponibilizado *online* em 2010, motivou o trabalho de Hao e Benlic (2011). Nesse trabalho é proposto um método baseado na ideia de particionar o problema original em subproblemas e resolver cada um deles separadamente, seguindo assim o príncípio de *dividir para conquistar*. A proposta é que um subconjunto selecionado dos conjuntos de restrições (3.18)−(3.26) do modelo de PLI da primeira fase é eliminado ou relaxado, de tal forma que o problema possa ser dividido em subproblemas independentes. Cada um dos subproblemas é resolvido pelo otimizador

COIN-OR (COIN-OR FOUNDATION, 2011) com tempo limite de 25200 segundos (7 horas) por subproblema. Os autores não detalham a motivação que os levou a definir esse tempo limite. Um limitante inferior válido para o problema original é então obtido pela soma dos valores obtidos na resolução dos subproblemas. É importante frisar que apenas o modelo da primeira fase, proposto por Lach e Lübbecke (2012), é considerado no trabalho de Hao e Benlic (2011).

O método proposto por Hao e Benlic (2011) pode ser resumido em três passos:

1. Particionar o conjunto C de disciplinas.
2. Resolver cada subproblema obtido em cada classe da partição, com um otimizador.
3. Somar os valores encontrados na etapa anterior para calcular o limitante inferior do problema original.

Para realizar a partição descrita no passo 1, Hao e Benlic (2011) executaram um algoritmo *Iterated Tabu Search*. Esse algoritmo define a partição do problema original em k (parâmetro de entrada) classes através do grafo $G = (V, A)$, definido da seguinte maneira: o conjunto de vértices V possui um vértice para cada disciplina $c \in C$, e uma aresta $\{c_1,c_2\}$ pertence a A se existir pelo menos um currículo $u \in U$ que tenha ambas as disciplinas, ou seja, $c_1, c_2 \in C_u$ para algum $u \in U$.

Durante a execução do algoritmo *Iterated Tabu Search* o tamanho de cada classe (subconjunto de vértices) é limitado em $(1, 2 \times |C|)/k$. Assim, evita-se que as classes da partição tenham tamanhos muito diferentes entre si. Após a execução do algoritmo, caso os vértices c_1 e c_2 pertençam a diferentes classes, sendo que c_1 e c_2 são adjacentes entre si ($c_1, c_2 \in C_u$ para algum $u \in U$), então todas as arestas que representam o currículo u, inclusive a aresta $\{c_1, c_2\}$, são removidas de G.

Dessa forma, no modelo proposto por Lach e Lübbecke (2012) para a primeira fase, Hao e Benlic (2011) consideram que um subconjunto dos conjuntos de restrições (3.24) e (3.25), que correspondem às arestas eliminadas, podem ser desconsideradas no problema e assim não pertencerem a nenhum dos subproblemas. Nos conjuntos de restrições (3.19), (3.21) e (3.26), aquelas que relacionarem disciplinas de classes diferentes serão relaxadas, permitindo que o problema original possa ser dividido em subproblemas independentes.

Hao e Benlic (2011) executaram o algoritmo *Iterated Tabu Search* variando o valor de k entre 2 e 6 para cada uma das 21 instâncias da formulação CB-CTT proposta no ITC-2007. Em todas as execuções foi encontrada uma partição do problema em menos de 600 segundos. Os resultados apresentados por Hao e Benlic (2011) são apenas do melhor valor de limitante inferior e o respectivo número de classes (k). A partir dos resultados é possível notar que o método proposto foi capaz de encontrar limitantes inferiores, em

média, melhores que os obtidos na resolução do modelo *Surface2* de Burke et al. (2012), sendo que a otimalidade de 7 das 14 instâncias iniciais foi provada, além de ter encontrado o melhor valor de limitante inferior, conhecido até aquele momento, para outras quatro instâncias.

O trabalho de Hao e Benlic (2011) e os demais trabalhos descritos a seguir, nesta seção, executaram seus métodos após a divulgação, pela organização do ITC-2007, das últimas sete instâncias. Assim, esses trabalhos encontraram limitantes inferiores para todas as 21 instâncias da formulação CB-CTT.

3.1.5 Geração de colunas de Cacchiani et al. (2013)

No trabalho de Cacchiani et al. (2013) foram apresentados novos e diferentes modelos matemáticos para o problema. Dentre esses modelos destaca-se o denominado por *Two Weekly Schedule Types* (2WST), pois ao ser resolvido pelo otimizador CPLEX apresentou resultados melhores quando comparados com os demais. No entanto, é importante destacar que o modelo 2WST considera um conjunto maior de variáveis que o modelo Monolítico de Burke et al. (2010) e o modelo de duas fases de Lach e Lübbecke (2012).

O modelo 2WST é baseado no modelo Monolítico, apresentado na Seção 3.1.1, cuja função objetivo visa minimizar as violações das quatro restrições fracas (**S1**-**S4**) da formulação CB-CTT. Inicialmente, Cacchiani et al. (2013) dividiram a função objetivo do modelo Monolítico em duas partes, sendo que na primeira parte são consideradas apenas as restrições **S1** (capacidade de sala) e **S4** (estabilidade de sala). Já na segunda parte são consideradas as outras restrições: **S2** (número mínimo de dias de aula) e **S3** (aulas isoladas). Com essa divisão, dois modelos matemáticos independentes, denominados *M1* e *M2*, foram obtidos, sendo que a resolução do modelo *M1* encontra uma solução que minimiza a violação das restrições **S1** e **S4**, enquanto que a resolução do *M2* encontra uma solução que minimiza as violações das demais restrições (**S2** e **S3**).

A partir disso, o modelo 2WST foi proposto como a combinação linear de soluções dos modelos *M1* e *M2*. Em outras palavras, o 2WST tem por objetivo minimizar a soma das soluções obtidas pelos dois modelos, com restrições que garantam que apenas uma solução, para cada um dos modelos, será considerada.

Cacchiani et al. (2013) apresentam então um método para encontrar limitantes inferiores. Esse método consiste em resolver separadamente os modelos *M1* e *M2*, pois a soma dos valores encontrados representa um limitante inferior. Para resolver o modelo *M1* foi utilizado o otimizador CPLEX, enquanto o modelo *M2* foi resolvido pelo método de *Geração de Colunas* com tempo de execução limitado em 22800 segundos (40×570), sendo 570 segundos o valor informado pelo programa *benchmark*, disponibilizado pela organização do ITC-2007, no computador usado para os experimentos computacionais.

O método proposto por Cacchiani et al. (2013) provou a otimalidade de 6 instâncias e em outras 6 instâncias os valores obtidos foram melhores que os resultados encontrados até aquele momento, superando assim, em média, os resultados obtidos por Hao e Benlic (2011).

3.1.6 Codificações SAT de Achá e Nieuwenhuis (2014)

O trabalho de Achá e Nieuwenhuis (2014) apresenta diferentes formas de codificar a formulação CB-CTT para o problema de satisfatibilidade booleana, também denominado SAT (GAREY; JOHNSON, 1979). As codificações diferem entre si pelos subconjuntos de restrições fortes e fracas consideradas e pela maneira como são definidas. Mesmo com essas diferenças, todas as codificações propostas podem encontrar limitantes inferiores e superiores ao problema, ou seja, em alguns casos a otimalidade também pode ser provada.

Vale ressaltar que no problema SAT as restrições fracas também são definidas como restrições fortes, ou seja, se um otimizador para o problema SAT encontra uma solução, obrigatoriamente essa solução terá custo zero.

Dentre as codificações propostas, a que apresentou os melhores limitantes inferiores foi a codificação que inicialmente desconsidera todas as restrições fracas, mas que depois vai adicionando cláusulas ao problema SAT. Essas cláusulas modelam as restrições fracas e a quantidade de cláusulas adicionadas é proporcional ao valor das penalidades. Assim, W^{RC} cláusulas são adicionadas para levar em conta a restrição **S1** (capacidade de sala), W^{MD} para levar em conta **S2** (número mínimo de dias de aula), e assim por diante. As codificações propostas foram resolvidas com um otimizador para o problema SAT, denominado Barcelogic (BOFILL et al., 2008). O tempo limite de execução foi de 10^5 segundos, sendo que os autores não detalham a motivação que os levou a definir esse tempo limite.

Apesar da média dos limitantes inferiores encontrados por Achá e Nieuwenhuis (2014) ter sido pior do que a encontrada por Cacchiani et al. (2013), a codificação provou a otimalidade de 11 das 21 instâncias, sendo portanto o método que até aquele momento provou a otimalidade no maior número de instâncias.

3.1.7 Novas abordagens de Bagger et al. (2019)

Três trabalhos (BAGGER; DESAULNIERS; DESROSIERS, 2019; BAGGER et al., 2019; BAGGER; SøRENSEN; STIDSEN, 2019) foram recentemente publicados apresentando novos valores de limitantes inferiores para a formulação CB-CTT do PTHU. Esses trabalhos possuem em comum o autor Niels-Christian Fink Bagger da Universidade Técnica da Dinamarca e o ano da publicação: 2019. No entanto, em cada um deles é apresentada uma abordagem diferente para obtenção de limitantes inferiores para a formulação

CB-CTT do PTHU.

Além disso, apesar do ano de publicação dos trabalhos ser 2019, cada um deles, Bagger, Desaulniers e Desrosiers (2019), Bagger et al. (2019) e Bagger, Sørensen e Stidsen (2019), teve uma primeira versão disponibilizada *online*, respectivamente, em 2016, 2017 e 2018.

No trabalho disponibilizado *online* em 2016, Bagger, Desaulniers e Desrosiers (2019) apresentam um modelo matemático com relaxações. Nesse modelo matemático apenas as restrições fracas **S2** (número mínimo de dias de aula) e **S3** (aulas isoladas) são consideradas na função objetivo. Leva-se em consideração também o que os autores denominam de padrões de alocações, sendo que um padrão representa a possibilidade de alocação de aulas de uma disciplina em um subconjunto dos períodos de um dia. Um conjunto de restrições garante que apenas um padrão por disciplina e por dia é escolhido. Outro conjunto de restrições garante que no máximo $|R|$ aulas serão alocadas em um mesmo período.

O número de padrões de um dia d é $2^{|P_d|}$, ou seja, se a tabela-horário tiver, por exemplo, 4 períodos por dia, então haverá 16 padrões de alocação de aulas de cada disciplina $c \in C$ no dia d. Considerando que esse valor aumenta consideravelmente o número de variáveis e restrições do modelo, os autores propõem uma etapa de pré-processamento, para reduzir o número de variáveis e restrições. Em seguida, os autores apresentam e provam algumas desigualdades que tornam a relaxação mais forte se inseridas como restrições no modelo.

Considerando que o valor informado pelo programa *benchmark*, disponibilizado pela organização do ITC-2007, no computador usado para os experimentos computacionais foi de 208 segundos, Bagger, Desaulniers e Desrosiers (2019) resolvem o modelo com o otimizador Gurobi (GUROBI OPTIMIZATION, 2016), limitando o tempo de execução em 100 vezes o valor retornado pelo programa *benchmark*, ou seja, 20800 segundos (100 $\times$ 208). Os valores de limitantes inferiores obtidos provaram a otimalidade em 13 das 21 instâncias, sendo que atualmente é o método da literatura que provou a otimalidade no maior número de instâncias. Em outras 4 instâncias os valores obtidos foram melhores que os resultados encontrados anteriormente na literatura, superando assim, em média, os resultados obtidos por Cacchiani et al. (2013).

No segundo trabalho, disponibilizado *online* em 2017, Bagger et al. (2019) apresentam a divisão do modelo matemático reproduzido na Seção 3.1.1 em dois modelos independentes. No primeiro modelo é considerada apenas a alocação das aulas em períodos, ou seja, desconsidera-se a existência de salas e consequentemente os conjuntos de restrições que modelam **S1** (capacidade de sala) e **S4** (estabilidade de sala). Em contrapartida, o segundo modelo considera apenas a alocação das aulas em salas, desconsiderando a existência de períodos e levando em conta apenas o conjunto de restrições que modelam **S4** (estabilidade de sala).

Bagger et al. (2019) afirmam que a motivação do desenvolvimento do trabalho vem de Lach e Lübbecke (2012), pois estes também dividiram o modelo matemático em dois modelos menores. No entanto, Lach e Lübbecke (2012) propõem a resolução dos dois modelos em sequência, enquanto que Bagger et al. (2019), apesar de também dividirem o modelo matemático em dois, propõem que os modelos sejam agrupados novamente em um único modelo. A conexão entre os dois modelos é feita com técnicas de formulação de fluxo, baseadas no problema de fluxo de custo mínimo em uma rede (AHUJA; MAGNANTI; ORLIN, 1993). Nessa conexão, leva-se em consideração a restrição fraca **S1** (capacidade de sala).

Portanto, os autores propõem a divisão do modelo proposto por Burke et al. (2010) em dois modelos independentes que depois são agrupados novamente em um único modelo através de técnicas de formulação de fluxo. O novo modelo é então resolvido com o otimizador Gurobi, cujo tempo de execução foi limitado em 8320 segundos (40×208), sendo 208 segundos o valor informado pelo programa *benchmark*, disponibilizado pela organização do ITC-2007, no computador usado para os experimentos computacionais. Apesar dos valores de limitantes inferiores obtidos, em média, serem piores que os obtidos no trabalho anterior (BAGGER; DESAULNIERS; DESROSIERS, 2019), eles foram, também em média, melhores que os valores obtidos por Lach e Lübbecke (2012), que motivaram esse trabalho, e os valores obtidos por Hao e Benlic (2011), que também foram motivados por Lach e Lübbecke (2012). A otimalidade foi provada em apenas 7 das 21 instâncias.

Por fim, no terceiro trabalho, disponibilizado *online* em 2018, Bagger, Sørensen e Stidsen (2019) propuseram um método de *Geração de Colunas* aplicado ao modelo apresentado por Bagger, Desaulniers e Desrosiers (2019). O método foi resolvido com o otimizador Gurobi, cujo tempo limite de execução não foi definido. No entanto, Bagger, Sørensen e Stidsen (2019) reportam o tempo gasto pelo método em cada instância, sendo que foram necessários 87471 segundos para se obter os valores de limitantes inferiores para todas as instâncias.

Os valores de limitantes inferiores obtidos com o método *Geração de Colunas* proposto por Bagger, Sørensen e Stidsen (2019) foi, em média, melhor que os demais resultados encontrados na literatura até o momento. No entanto, a otimalidade foi provada em apenas 7 das 21 instâncias. A partir dos resultados, é importante observar também que em outras 4 instâncias o valor de limitante inferior foi o melhor valor obtido até então, sendo que para uma dessas instâncias o resultado obtido superou em 41,71% o valor considerado o melhor até aquele momento. Dessa forma, os valores de limitantes inferiores obtidos nessas 4 instâncias foram os responsáveis para o método *Geração de Colunas* de Bagger, Sørensen e Stidsen (2019) ter superado, em média, os valores obtidos por Bagger, Desaulniers e Desrosiers (2019).

3.1.8 Observações finais

Nesta seção foram apresentados, de maneira resumida, diversos trabalhos da literatura em que são propostos modelos matemáticos para o PTHU da formulação CB-CTT. Dentre esses, destacam-se dois: o modelo Monolítico proposto por Burke et al. (2010) e o modelo de duas fases proposto por Lach e Lübbecke (2012). O Monolítico é um modelo completo, ou seja, considera todas as restrições fracas (**S1**-**S4**). No entanto, possui muitas variáveis e restrições, o que dificulta sua resolução. Em contrapartida, o modelo de duas fases de Lach e Lübbecke (2012) possui a desvantagem de não considerar todas as restrições fracas em uma única formulação, mas sua resolução é mais simples, já que apresenta menos variáveis e restrições. Os demais modelos matemáticos propostos para a formulação CB-CTT apresentam mais variáveis e restrições que o Monolítico, ou da mesma forma que o modelo de duas fases de Lach e Lübbecke (2012), também não são completos.

Além dos trabalhos de Burke et al. (2010) e Lach e Lübbecke (2012), nesta seção também são apresentados outros que propõem métodos de obtenção de limitantes inferiores para o PTHU da formulação CB-CTT. A variedade de abordagens adotadas nesses trabalhos é grande, incluindo métodos clássicos como *Branch-and-cut* e *Geração de colunas*.

Apesar disso, nenhuma das abordagens propostas conseguiu provar a otimalidade de todas as 21 instâncias usadas na formulação CB-CTT do ITC-2007. Em cinco dessas instâncias a otimalidade ainda não foi provada.

Ao ordenar cronologicamente os trabalhos apresentados nesta seção, observa-se no entanto que cada um deles contribuiu de maneira significativa no estudo do problema, das quais destaca-se a melhora dos valores de limitantes inferiores encontrados anteriormente, provando a otimalidade em uma quantidade maior de instâncias, ou ainda, encontrando melhor valor médio de limitantes inferiores.

Dessa maneira, diversos métodos para obtenção de limitantes inferiores foram apresentados nesta seção, cada um com características específicas, o que dificulta realizar uma comparação entre eles. Entretanto, grande parte desses métodos possui em comum a estratégia de diminuir a complexidade do problema original, seja dividindo o modelo matemático em subproblemas ou relaxando algumas restrições.

Portanto, os trabalhos apresentados nesta seção contribuiram na compreensão das melhores estratégias a serem usadas para obtenção de limitantes inferiores para o PTHU da formulação CB-CTT. Com isso, os métodos apresentados nesta tese (Seções 4.3 e 4.4) seguem as estratégias de dividir o modelo matemático em subproblemas e também relaxar algumas restrições.

3.2 Métodos heurísticos

Nesta seção são apresentados os principais algoritmos heurísticos propostos para a solução da formulação CB-CTT do ITC-2007.

Inicialmente, na Subseção 3.2.1 são apresentados os algoritmos que se destacaram na competição, alcançando a primeira e a segunda colocação. Em seguida, na Subseção 3.2.2 são apresentados alguns trabalhos posteriores que obtiveram resultados de boa qualidade, sendo que os resultados obtidos por alguns desses algoritmos superam aqueles obtidos pelos competidores do ITC-2007.

É importante observar, que na Subseção 3.2.2 são apresentados apenas uma parte dos trabalhos que propõem métodos de solução heurísticos para a formulação CB-CTT do ITC-2007. A escolha desses trabalhos foi feita entre os que apresentaram os melhores valores médios de solução viável até o momento ou que foram desenvolvidos pelo grupo de pesquisa do Laboratório de Otimização e Modelagem Computacional (Labotim) da UFES, que estuda o PTHU da formulação CB-CTT proposta no ITC-2007.

3.2.1 Vencedores do ITC-2007

O ITC-2007 se distinguiu das demais edições do campeonato por representar o PTHU em três formulações distintas, conforme apresentado na Seção 1.1. A formulação CB-CTT teve 17 competidores, sendo que todos os algoritmos enviados foram executados 10 vezes em cada uma das 14 instâncias disponibilizadas antes do prazo final de envio. Os cinco competidores que obtiveram, em média, os melhores resultados nessa primeira fase foram classificados para a fase seguinte, que consistia na execução dos respectivos algoritmos, novamente 10 vezes, em outras sete instâncias, que só foram disponibilizadas após a divulgação do resultado final da competição.

Ao observar os resultados gerais da competição é possível perceber que os dois primeiros colocados tiveram, em média, resultados muito próximos, mas que se distanciam dos resultados dos demais competidores. Na primeira fase da competição, a média do primeiro colocado (MÜLLER, 2009) foi apenas 1,07% menor que a do segundo colocado (LÜ; HAO, 2010). Já na segunda fase a diferença entre eles foi de 2,45%, enquanto que a diferença entre Müller (2009) e o terceiro colocado foi de 13,61%. Além disso, Müller (2009) encontrou em 10 das 21 instâncias o melhor resultado da competição, enquanto que Lü e Hao (2010) encontrou o melhor resultado em outras 7 instâncias. Em três das quatro instâncias restantes foi registrado um empate entre os melhores resultados obtidos, ou seja, em apenas uma instância o melhor resultado da competição não foi obtido por Müller (2009) ou Lü e Hao (2010). Dessa forma, nesta seção é apresentado apenas as contribuições dos dois primeiros colocados na formulação CB-CTT proposta no ITC-2007.

Müller (2009) participou do ITC-2007, na formulação CB-CTT, com um algoritmo

que possui três fases, sendo que na primeira uma solução viável é obtida usando o algoritmo *Iterative Forward Search* (MÜLLER, 2005), que por sua vez faz uso de *Conflict-based Statistics* para prevenir ciclos durante as iterações de construção da solução. Na fase seguinte, a partir da solução contruída, uma solução ótima local é obtida usando o método *Hill Climbing* (RUSSELL; NORVIG, 1995), que possui movimentos específicos ao problema, relacionados às restrições fracas, para obter uma solução vizinha. Por fim, o método *Great Deluge* (DUECK, 1993) é usado, de tal forma que um limite é colocado no valor da solução, e esse limite é iterativamente reduzido durante a busca. Nessa fase, o método *Simulated Annealing* (KIRKPATRICK; GELATT; VECCHI, 1983) é aplicado quando o valor limite é reduzido.

O algoritmo *Adaptive Tabu Search* proposto por Lü e Hao (2010) é composto por duas fases: inicialização e integração (intensificação e diversificação). Na fase de inicialização um algoritmo guloso constrói uma solução viável para o problema, ou seja, a partir de uma tabela-horário vazia uma solução é construída iterativamente selecionando e alocando, em cada iteração, uma aula de uma disciplina. A escolha da aula a ser alocada é feita a partir de uma lista ordenada. A ordenação dessa lista é definida pelas disciplinas que possuem poucos períodos disponíveis e grande número de aulas ainda não alocadas.

A solução construída é então submetida à fase seguinte, que congrega formas de intensificação e diversificação na busca de soluções. A intensificação é realizada pelo algoritmo *Tabu Search*, que gradualmente aumenta a profundidade de busca, enquanto que a diversificação é realizada com um procedimento de *pertubação* na melhor solução encontrada até o momento.

3.2.2 Métodos heurísticos posteriores ao ITC-2007

Após o ITC-2007 outros métodos de solução heurísticos são propostos na literatura para a formulação CB-CTT. Dentre esses trabalhos, Rocha (2013) usa os trabalhos de Müller (2009) e Lü e Hao (2010) como motivação, e implementa a meta-heurística GRASP (FEO; RESENDE, 1989) para o problema. Nesse caso, o GRASP proposto por Rocha (2013) constrói uma solução viável em cada iteração. A solução construída é submetida a uma etapa de busca local, sendo que após a busca local a estratégia de intensificação *Path-Relinking* (GLOVER, 1997) também é aplicada. Foram testados dois métodos na etapa de busca local: *Hill Climbing* e *Simulated Annealing*. Para obter uma solução vizinha, os métodos aplicam um dos movimentos *move* e *swap*. A escolha do movimento é feita de forma aleatória. Os resultados obtidos com *Simulated Annealing* foram melhores que os resultados obtidos com *Hill Climbing*. Em comparação com os resultados obtidos durante o ITC-2007, os resultados do GRASP de Rocha (2013) são inferiores aos resultados dos três primeiros colocados na competição, ou seja, caso tivesse participado do ITC-2007 teria ocupado a 4ª colocação.

Segatto et al. (2015) deu continuidade ao trabalho de Rocha (2013) implementando o movimento conhecido como *Cadeia de Kempe* (MORGENSTERN, 1989) para obter uma solução vizinha na etapa de busca local. No entanto, melhorias foram obtidas em apenas algumas instâncias e, em média, os resultados foram inferiores aos obtidos por Rocha (2013). Caso essa nova versão do algoritmo GRASP tivesse participado do ITC-2007 teria obtido a 6ª colocação.

Posteriormente, Segatto (2017) implementou melhorias na codificação dos seus algoritmos, deixando-os mais eficientes e consequentemente obtendo melhores resultados. Essa nova versão do GRASP, denominada GRASPK, encontrou resultados que, em média, são melhores do que os obtidos por Müller (2009). Dessa forma, caso o algoritmo GRASPK tivesse participado do ITC-2007, ao final da competição teria ocupado a primeira colocação.

O trabalho de Bellio et al. (2016) também representa um marco importante dentre as propostas de métodos heurísticos de solução da formulação CB-CTT, pois apresenta uma dupla contribuição. Primeiramente os autores propuseram um efetivo e robusto método *Simulated Annealing* de fase única para resolver o problema. Em seguida, projetaram e aplicaram uma extensa análise baseada em princípios estatísticos para o procedimento de calibragem de parâmetros. O resultado dessa análise foi uma metodologia que permite a definição dos parâmetros a partir das características de cada instância, como exemplo: a quantidade de disciplinas, a quantidade de salas e a quantidade de currículos. Os resultados obtidos pelo algoritmo *Simulated Annealing* de Bellio et al. (2016) são melhores do que os resultados do algoritmo GRASPK de Segatto (2017), que por sua vez são melhores do que os apresentados por Müller (2009), vencedor do ITC-2007 na formulação CB-CTT.

Kiefer, Hartl e Schnell (2017) apresentam um modelo matemático e também um algoritmo *Adaptive Large Neighborhood Search* para a solução da formulação CB-CTT. O algoritmo proposto apresenta diversos operadores de destruição e reconstrução de uma solução. A escolha dos operadores é feita de forma aleatória, mas de acordo com uma probabilidade definida a partir da qualidade das soluções obtidas nas iterações anteriores. A média dos resultados obtidos foi melhor que a obtida por Bellio et al. (2016), sendo este trabalho o que apresenta as melhores soluções atualmente.

Por fim, o trabalho de Kampke et al. (2017) amplia a análise feita por Lü, Hao e Glover (2011) sobre os movimentos aplicados na busca local dos métodos heurísticos propostos para a formulação CB-CTT, incluindo nessa análise os movimentos propostos por Müller (2009). Posteriormente os autores propuseram um novo método de construção de uma solução inicial para a meta-heurística GRASP apresentada por Kampke et al. (2015), dessa vez propondo a utilização de uma modelagem do problema em uma rede de fluxo. Na fase de busca local é proposta a utilização do movimento *Lecture Move* que obteve o melhor desempenho na análise realizada anteriormente. Com isso, Kampke et al. (2019) apresentou avanços significativos nos resultados obtidos pela meta-heurística

GRASP e, apesar de não ter superado os resultados de Kiefer, Hartl e Schnell (2017), Bellio et al. (2016) e Segatto (2017), os resultados obtidos são melhores do que os apresentados por Müller (2009), vencedor do ITC-2007 na formulação CB-CTT.

3.2.3 Observações finais

Nesta seção foram apresentados, de maneira resumida, os principais algoritmos heurísticos propostos para o PTHU da formulação CB-CTT.

Ao apresentar esses algoritmos foi possível notar uma grande variedade de métodos, que por sua vez usam diferentes estratégias para obtenção de soluções viáveis (limitantes superiores). Assim, esses trabalhos, em especial os desenvolvidos no Labotim/UFES, contribuíram para o entendimento da dificuldade em encontrar uma solução viável em cada instância da formulação CB-CTT do ITC-2007, além de perceber características específicas dessas instâncias.

Por fim, é importante destacar que a análise dos diferentes métodos de construção de uma solução inicial e as análises realizadas sobre os movimentos aplicados na busca local, também contribuíram para o entendimento do PTHU abordado. Todas essas contribuições foram importantes na proposta dos métodos apresentados nesta tese (Seções 4.3 e 4.4) e por consequência nas suas implementações.

4 Metodologia

Em diversos trabalhos na literatura são propostos métodos para encontrar limitantes para o PTHU, conforme destaque no Capítulo 3. A importância desses métodos está no fato de permitir a avaliação da qualidade das soluções obtidas por outros métodos e até mesmo provar a otimalidade das soluções.

Neste capítulo são apresentadas novas estratégias para encontrar limitantes inferiores para a formulação CB-CTT do ITC-2007. Essas estratégias adotam o princípio de *dividir para conquistar*, também adotado por Hao e Benlic (2011), que particiona o problema original em problemas menores e em seguida resolve cada um deles com um otimizador.

Ao particionar o problema, algumas restrições do modelo matemático são relaxadas, possibilitando encontrar valores de limitantes inferiores válidos. Assim, na Seção 4.1 são apresentadas duas formas de representar a formulação CB-CTT como um grafo $G = (V, A)$. Na primeira forma, denominada G_U, cada currículo $u \in U$ é representado por um vértice de G e na outra, denominada G_C, cada disciplina $c \in C$ representa um vértice de G. Em G_U, a aresta $\{u_1, u_2\} \in A$, se, e somente se, os currículos u_1 e u_2 tiverem pelo menos uma disciplina em comum, ou seja, se $C_{u_1} \cap C_{u_2} \neq \emptyset$. Já no grafo G_C, a aresta $\{c_1, c_2\} \in A$, se, e somente se, as disciplinas c_1 e c_2 pertencerem simultaneamente a um mesmo currículo, ou seja, se $U_{c_1} \cap U_{c_2} \neq \emptyset$.

A Seção 4.2 apresenta resumidamente a biblioteca *METIS*, proposta por Karypis e Kumar (1998), que implementa heurísticas para o particionamento de grafos. Os dois principais parâmetros dessas heurísticas são o grafo G a ser particionado e o número de classes k. Considerando que durante o particionamento de G algumas arestas serão cortadas e que essas arestas representam as restrições do modelo matemático que serão relaxadas, pode-se afirmar que o particionamento ideal para os métodos propostos nesse trabalho é aquele que minimiza o número de arestas cortadas, e consequentemente o número de restrições relaxadas.

Nesse momento é importante salientar, conforme apresentado no Capítulo 3, que Hao e Benlic (2011) implementaram a heurística *Busca Tabu Iterativa* para particionar o grafo G_C. Essa heurística também tem como objetivo encontrar um particionamento que minimiza o número de arestas cortadas.

Na Seção 4.3 são apresentados os métodos *U-partition* e *C-partition* que particionam, respectivamente, os grafos G_U e G_C. Em outras palavras, esses métodos particionam o problema original em subproblemas e, em seguida, relaxando algumas restrições do modelo matemático, cada subproblema é resolvido pelo otimizador CPLEX.

Na Seção 4.4 são apresentados os métodos *C-partition-null* e *C-partition-org*. A diferença desses métodos para o método *C-partition* está na forma como as restrições são relaxadas, uma vez que usam variáveis auxiliares, denominadas variáveis de cópia, para obter uma relaxação mais forte. A diferença entre os métodos *C-partition-null* e *C-partition-org* está apenas no valor dos coeficientes das variáveis de cópia nas funções objetivo dos subproblemas.

Por fim, na Seção 4.5 é apresentado um fluxograma para os quatro métodos propostos. Nesse fluxograma são representadas todas as etapas executadas pelos métodos propostos nas Seções 4.3 e 4.4.

4.1 Representação do PTHU usando grafos

Neste trabalho são propostas duas formas distintas para representar via grafos, o PTHU segundo a formulação CB-CTT. Nas Subseções 4.1.1 e 4.1.2 são apresentadas cada uma delas.

4.1.1 Grafo G_U

Considerando $U = \{u_1, u_2, \ldots, u_n\}$ o conjunto com n currículos e C_{u_i} o conjunto de disciplinas do currículo u_i, o PTHU da formulação CB-CTT é então representado pelo grafo valorado $G_U = (V, A)$, sendo que G_U consiste em um conjunto V de n vértices e um conjunto A de arestas entre esses vértices. Cada vértice $u_i \in V$, $i = 1, 2, \ldots, n$, corresponde a um currículo do conjunto U. Para cada par de currículos u_i e u_j em U, $\{u_i, u_j\} \in A$ se $C_{u_i} \cap C_{u_j} \neq \emptyset$ e $i \neq j$. O valor (ou peso) de cada aresta $\{u_i, u_j\} \in A$ é igual a $|C_{u_i} \cap C_{u_j}|$, que representa o número de disciplinas que os currículos u_i e u_j possuem em comum.

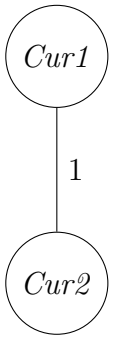

Figura 3 – Grafo G_U da instância *Toy*.

Na Figura 3 é exibido o grafo G_U da instância *Toy* apresentada na Figura 1. Observe que nesse caso o grafo possui dois vértices que representam os currículos *Cur1* e *Cur2*. A aresta entre os currículos possui peso igual a 1, pois os currrículos possuem apenas a disciplina *TecCos* em comum.

4.1.2 Grafo G_C

Neste trabalho também é apresentada uma segunda forma de representar o problema com grafos, que é semelhante a que foi proposta por Hao e Benlic (2011). Assim, considerando o conjunto $C = \{c_1, c_2, \ldots, c_m\}$ com m disciplinas, o grafo valorado $G_C = (V, A)$ corresponde ao conjunto V com m vértices e ao conjunto A de arestas entre esses vértices. Cada vértice $c_i \in V$, $i = 1, 2, \ldots, m$, corresponde a uma disciplina do conjunto C. Para cada par de disciplinas c_i e c_j em C, $\{c_i, c_j\} \in A$ se existe pelo menos um currículo u tal que $u \in U_{c_i}$ e $u \in U_{c_j}$, ou seja, $U_{c_i} \cap U_{c_j} \neq \emptyset$ e $i \neq j$.

Da mesma forma que o grafo G_U, mas diferentemente do que foi proposto por Hao e Benlic (2011), o grafo G_C é valorado e o peso de cada aresta $\{c_i, c_j\} \in A$ é igual a $|U_{c_i} \cap U_{c_j}|$, que representa a quantidade de currículos nos quais as disciplinas c_i e c_j pertencem simultaneamente.

A partir disso, pode-se afirmar que todo par de disciplinas c_i e c_j, com $i \neq j$, que pertencem simultaneamente a um currículo $u \in U$, são adjacentes entre si, ou seja, todos os vértices que representam as disciplinas $c \in C_u$ formam um *clique* em G_C.

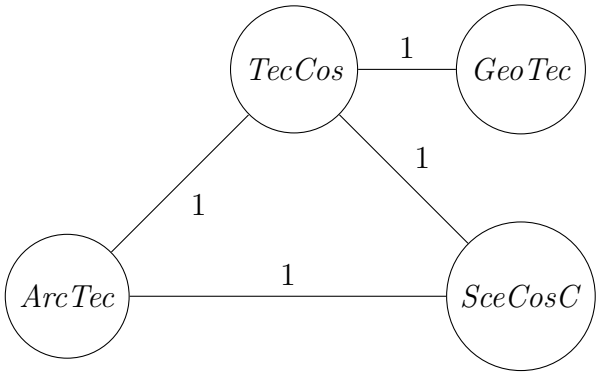

Figura 4 – Grafo G_C da instância *Toy*.

A Figura 4 exibe o grafo G_C da instância *Toy*. Cada disciplina da instância é representada por um vértice. As arestas $\{TecCos, ArcTec\}$, $\{SceCosC, ArcTec\}$ e $\{TecCos, SceCosC\}$ formam um *clique* de tamanho 3 que representa o currículo *Cur1*, enquanto que a aresta $\{TecCos, GeoTec\}$ forma um *clique* de tamanho 2 representando o currículo *Cur2*. Nesse exemplo, todas as arestas possuem peso igual a 1, pois as disciplinas adjacentes pertencem simultaneamente apenas a um currículo.

4.2 *METIS*

O problema de particionar o grafo valorado e não direcionado $G = (V, A)$ consiste em dividir o conjunto V de vértices em k subconjuntos $V_1, V_2, \ldots, V_k$ de tal maneira que $V_i \cap V_j = \emptyset$ para $i \neq j$, $V_1 \cup V_2 \cup \cdots \cup V_k = V$, a soma dos pesos das arestas de A que são incidentes a vértices que pertencem a diferentes subconjuntos de V é minimizado e o

tamanho desses subconjuntos seja balanceado ($|V_i| \approx |V|/k$) (KERNIGHAN; LIN, 1970). Cada um dos subconjuntos de V é denominado classe ou subdomínio (CARVALHO, 2002).

A solução do problema, que representa um particionamento do grafo, pode ser facilmente representada por um vetor Υ de tamanho $|V|$ e para cada vértice $v \in V$, $\Upsilon[v]$ é um número inteiro entre 1 e k que indica a classe que o vértice v pertence. Assim, dado um particionamento Υ, as arestas incidentes a vértices que pertencem a classes diferentes são denominadas *arestas cortadas* e formam um subconjunto de A denominado A^-.

O conceito de *Acoplamento Maximal*, apresentado no Capítulo 3, é usado na definição do paradigma *multinível* de resolução do problema de particionar um grafo G em k classes. Karypis e Kumar (1998) apresentam heurísticas que implementam o paradigma *multinível*. Nesse paradigma, três fases são empregadas para particionar o grafo. Na primeira fase uma sequência de grafos $G_0, G_1, \ldots, G_\eta$ é gerada, sendo que o grafo G_0 é o grafo original e para cada i e j, tal que $0 \leq i \leq j \leq \eta$, o grafo G_i possui mais vértices que o grafo G_j. Em outras palavras, o grafo original G_0 é iterativamente reduzido ao grafo G_η. Na iteração i, $i = 0, 1, \ldots, \eta - 1$, é obtido um acoplamento maximal M_i em G_i. Os vértices incidentes a uma aresta $a \in M_i$ são agrupados em um único vértice e o peso das demais arestas incidentes aos vértices é incrementado. Dessa forma, o acoplamento maximal M_i é usado para gerar o grafo G_{i+1}, que possui menos vértices que G_i. Karypis e Kumar (1998) apresentam três heurísticas para obter M_i e todas elas possuem complexidade $O(|A|)$.

A segunda fase da abordagem *multinível* consiste então em particionar o grafo G_η em k classes, obtendo assim o particionamento Υ_η. Em outro trabalho, Karypis e Kumar (1999) apresentam quatro heurísticas para este objetivo. A terceira fase consiste basicamente no inverso da primeira, sendo que na iteração j, $j = \eta, \eta - 1, \ldots, 0$, é aplicada uma heurística de busca local no particionamento Υ_j, com o objetivo de minimizar a soma dos pesos das arestas de A^-, atendendo à restrição que o tamanho das classes seja balanceado. Karypis e Kumar (1998) apresentam duas heurísticas para este propósito e consequentemente para obter o particionamento Υ_0 do grafo original G_0.

Karypis e Kumar (1998) mostram que o particionamento de um grafo $G = (V, A)$ em k classes pode ser obtido de forma rápida, mesmo para grafos com valores elevados de $|V|$ e k, uma vez que o algoritmo que implementa heurísticas para as três fases do paradigma *multinível* possui complexidade final $O(|A|)$.

Na biblioteca *METIS*, desenvolvida no Departamento de Ciência da Computação e Engenharia da Universidade de Minnesota, são implementados algoritmos baseados no paradigma *multinível* para particionar grafos. A biblioteca é distribuida gratuitamente e está disponível para *download* em http://glaros.dtc.umn.edu/gkhome/views/metis. Vale destacar que a biblioteca possui uma *Application Programming Interface* (API) que pode ser invocada em algoritmos desenvolvidos nas linguagens *C*, *C++* ou *Fortran*. Essa API possui dois métodos, denominados *METIS_PartGraphRecursive* e *METIS_PartGraphKway*, para

particionar o grafo $G = (V, A)$ em k classes. Assim, os dois principais parâmetros desses métodos são G e k. Os dois métodos retornam o particionamento Υ. A diferença entre eles está no fato que o *METIS_PartGraphKway* possui capacidades adicionais, como exemplo, permite classes contíguas, ou seja, classes cuja diferença de tamanho entre si seja maior que uma unidade. Neste trabalho, o objetivo é obter um particionamento em que a diferença de tamanho entre as classes seja no máximo uma unidade, para que as classes tenham praticamente a mesma quantidade de vértices e, consequentemente, os subproblemas representados por essas classes tenham complexidade de resolução pelo CPLEX similares. Dessa forma, evita-se que haja subproblemas com poucos vértices e resolvidos facilmente, enquanto outros, com grande quantidade de vértices e de difícil resolução pelo CPLEX. Portanto, nesse trabalho é usado o método *METIS_PartGraphRecursive*.

4.3 Métodos sem variáveis de cópia

Nesta seção são apresentados os métodos *U-partition* e *C-partition* que, com a API da biblioteca *METIS*, particionam o problema original, representado respectivamente pelo grafo G_U ou G_C, em problemas menores.

4.3.1 *U-partition*

No método *U-partition* o grafo $G_U = (V, A)$ é particionado em k classes com a função *METIS_PartGraphRecursive*, invocada através da API da biblioteca *METIS*. Nesse caso, cada classe terá aproximadamente $|V|/k$ vértices e o particionamento encontrado minimiza a soma dos pesos das arestas cortadas que formam A^-.

Dessa forma, $\Upsilon[u]$ representa em qual classe (subproblema) o currículo $u \in U$ está presente. Portanto, a ideia principal do método *U-partition* consiste em dividir o problema original em subproblemas, que consequentemente possuem menos currículos que o problema original, mas com aproximadamente a mesma quantidade de currículos entre si.

Assim, ao particionar G_U, cada currículo $u \in U$ pertence a um, e apenas um, subproblema SP_i, $i = 1, \ldots, k$. No entanto, podem existir disciplinas que pertencem a mais de um currículo, que por sua vez pertencem aos subproblemas SP_i e SP_j, $i \neq j$. O conjunto de arestas cortadas A^- representam essas disciplinas. Portanto, se uma disciplina $c \in C$ não é representada por alguma aresta em A^-, isso indica que todos os currículos $u \in U_c$ pertencem ao mesmo subproblema SP_i. Com isso, a disciplina c será considerada na função objetivo e nas restrições apenas de SP_i.

No caso de uma disciplina $c \in C$, representada por uma ou mais arestas em A^-, o método *U-partition* adota uma estratégia gulosa para definir o subproblema em que c será considerada na função objetivo. A estratégia consiste em verificar o currículo

$u \in U_c$ que possua a maior quantidade de disciplinas. Em outras palavras, a disciplina $c \in C$, representada por uma ou mais arestas em A^-, é considerada na função objetivo do subproblema no qual o currículo u pertence, desde que u tenha o maior valor de $|C_u|$ dentre os currículos de U_c. Se houver empate a escolha é feita de forma aleatória. É importante salientar que a disciplina c é considerada nas restrições de todos os subproblemas que tenham pelo menos um currículo $u \in U_c$.

Nesse momento é importante recordar, conforme apresentado nas Seções 3.1.1 e 3.1.3, que a variável $\omega_{p,\psi,c}$ terá valor igual a 1 se a disciplina $c \in C$ possuir uma aula alocada no período $p \in P$ em uma sala com capacidade menor que $\psi \in \Psi$ e 0 caso contrário. Já a variável q_c representa a diferença entre o número mínimo de dias de aula (mld_c) definido para disciplina $c \in C$ e o número de dias distintos que a mesma disciplina possui aulas alocadas. Se a diferença for negativa, então q_c terá valor igual a 0. Por fim, a variável $z_{u,p}$ terá valor igual a 1 se há uma aula isolada do currículo $u \in U$ alocada no período $p \in P$ e 0 caso contrário. Dessa forma, as variáveis $\omega_{p,\psi,c}$ e q_c são definidas para cada disciplina $c \in C$ e estão relacionadas, respectivamente, às restrições fracas **S1** (capacidade de sala) e **S2** (número mínimo de dias de aula), enquanto que a variável $z_{u,p}$ é definida para cada currículo $u \in U$ e está relacionada à restrição fraca **S3** (aulas isoladas). Essas variáveis de decisão são as únicas presentes na função objetivo do modelo matemático da primeira fase proposto por Lach e Lübbecke (2012), já que a restrição fraca **S4** (estabilidade de sala) é considerada apenas no modelo matemático da segunda fase.

Portanto, a estratégia de considerar a disciplina $c \in C$, representada por uma ou mais arestas em A^-, na função objetivo de apenas um subproblema, é usada para evitar que essa disciplina contribua com o aumento do valor da função objetivo em dois, ou mais, subproblemas. Como exemplo, suponha que a disciplina $c \in C$ viole a restrição fraca **S2** (número mínimo de dias de aula) em dois dos subproblemas em que é considerada nas restrições. Caso essa disciplina não fosse considerada na função objetivo de apenas um subproblema, a mesma violação incrementaria a função objetivo dos dois subproblemas, ou seja, seria contabilizada duas vezes.

Após particionar o grafo G_U e definir em qual subproblema a disciplina $c \in C$ será considerada na função objetivo, o método *U-partition* gera cada subproblema, usando para isso o modelo matemático da primeira fase proposto por Lach e Lübbecke (2012) e apresentado na Seção 3.1.3. Assim, a função objetivo do subproblema SP_i terá apenas as variáveis $z_{u,p}$ em que $\Upsilon[u] = i$, além das variáveis q_c e $\omega_{p,\psi,c}$, caso a disciplina c seja considerada na função objetivo de SP_i.

De forma análoga à função objetivo, os conjuntos de restrições (3.24) e (3.25) pertencem ao subproblema SP_i, se $\Upsilon[u] = i$, uma vez que esses conjuntos de restrições modelam a violação de **S3** (aulas isoladas) para cada currículo $u \in U$. Já os conjuntos de restrições (3.18)−(3.23) e (3.26) pertencem a SP_i apenas se a disciplina $c \in C_u$ e $\Upsilon[u] = i$.

Os conjuntos de restrições (3.19), (3.21) e (3.26) também fazem parte do conjunto de restrições relaxadas do método *U-partition*, uma vez que em cada uma das restrições desse conjunto, apenas as variáveis $x_{p,c}$ e $\omega_{p,\psi,c}$ em que a disciplina $c \in C_u$ e $\Upsilon[u] = i$ são consideradas. Ou seja, cada uma dessas restrições possui tamanho reduzido em relação à mesma restrição no problema original.

Na Figura 5 é exibido o grafo G_U da instância *Toy* apresentada anteriormente. Observe que o grafo G_U foi particionado em 2 subproblemas ($k = 2$), SP_1 e SP_2. O currículo *Cur1* pertence a SP_1, enquanto o currículo *Cur2* pertence a SP_2. Dessa forma, os conjuntos de restrições (3.24) e (3.25) referentes ao currículo *Cur1* fazem parte apenas de SP_1 e as do *Cur2*, apenas de SP_2. Além disso, faz parte de SP_1 todas as restrições referentes a disciplina $c \in C_{Cur1}$, sendo $C_{Cur1} = \{SceCosC, ArcTec, TecCos\}$, ou seja, as três disciplinas do currículo *Cur1*. De forma semelhante, em SP_2 também são consideradas as restrições referentes à disciplina $c \in \{TecCos, GeoTec\}$, uma vez que *TecCos* e *GeoTec* são as duas disciplinas que fazem parte do currículo *Cur2*.

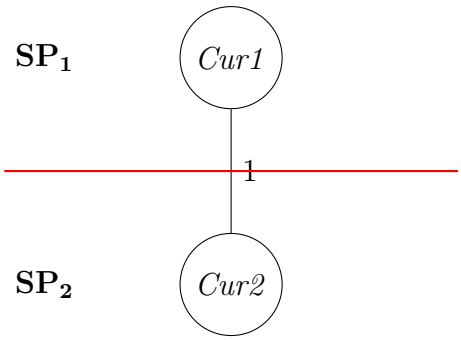

Figura 5 – Grafo G_U da instância *Toy* particionado em 2 subproblemas ($k = 2$).

Nesse mesmo exemplo é possível perceber que a partição obtida teve apenas uma aresta cortada. Essa aresta, que tem peso 1, representa a disciplina *TecCos*, já que essa disciplina pertence a ambos os currículos: *Cur1* e *Cur2*. Assim, apesar dos conjuntos de restrições (3.18)−(3.23) e (3.26) referentes à disciplina *TecCos* pertencerem aos subproblemas SP_1 e SP_2, as variáveis q_c e $\omega_{p,\psi,c}$, em que $c \in \{TecCos\}$, são consideradas apenas na função objetivo de SP_1, já que $|C_{Cur1}| > |C_{Cur2}|$. Portanto, fazem parte da função objetivo de SP_1 as variáveis $z_{u,p}$, q_c e $\omega_{p,\psi,c}$ em que $u \in \{Cur1\}$ e $c \in \{SceCosC, ArcTec, TecCos\}$ e de SP_2, as variáveis em que $u \in \{Cur2\}$ e $c \in \{GeoTec\}$.

4.3.2 *C-partition*

O método *C-partition* segue o mesmo princípio do método *U-partition*, ou seja, particiona o problema original em subproblemas independentes. Para isso, ambos os métodos relaxam algumas restrições presentes no problema original.

Apesar de seguirem o mesmo princípio, os métodos possuem algumas diferenças, que estão diretamente relacionadas ao grafo usado para representar o problema original,

uma vez que no método *U-partition* é usado o grafo G_U, e no método *C-partition* o grafo G_C. Assim, as restrições relaxadas, e consequentemente a forma que currículos e disciplinas são representados nos subproblemas, também diferenciam os dois métodos.

No método *C-partition*, da mesma forma que no método *U-partition*, o particionamento Υ é obtido com a função *METIS_PartGraphRecursive*, invocada através da API da biblioteca *METIS*. Nesse caso, já que o grafo G_C foi particionado, $\Upsilon[c]$ representa em qual classe (subproblema) a disciplina $c \in C$ pertence.

Portanto, no método *C-partition*, após particionar o grafo G_C, cada disciplina $c \in C$ pertence a um, e apenas um subproblema SP_i, $i = 1, \ldots, k$. Nesse caso, o conjunto de arestas cortadas A^- representa os currículos que possuem disciplinas que pertencem aos subproblemas SP_i e SP_j, $i \neq j$. Assim sendo, se um currículo $u \in U$ não é representado por alguma aresta em A^-, isso indica que todas as disciplinas $c \in C_u$ pertencem ao mesmo subproblema SP_i. Com isso, o currículo u será considerado na função objetivo e nas restrições apenas de SP_i. No entanto, se um currículo $u \in U$ é representado por uma ou mais arestas de A^-, o currículo u não é considerado na função objetivo e nas restrições de nenhum dos subproblemas.

O método *C-partition*, da mesma forma que o método *U-partition*, também gera cada subproblema com o modelo matemático da primeira fase proposto por Lach e Lübbecke (2012). A função objetivo de SP_i terá apenas as variáveis q_c e $\omega_{p,\psi,c}$ em que $\Upsilon[c] = i$, e somente as variáveis $z_{u,p}$ se toda disciplina $c \in C_u$ tiver $\Upsilon[c] = i$.

Da mesma forma que as variáveis $z_{u,p}$ na função objetivo, os conjuntos de restrições (3.24) e (3.25) pertencem a SP_i, se no currículo $u \in U$ toda disciplina $c \in C_u$ tiver $\Upsilon[c] = i$. Os conjuntos de restrições (3.18)−(3.23) e (3.26) pertencem a SP_i se $\Upsilon[c] = i$ para a disciplina $c \in C$.

Os conjuntos de restrições (3.19), (3.21) e (3.26) também fazem parte do conjunto de restrições relaxadas do método *C-partition*, uma vez que em cada uma das restrições desse conjunto no SP_i, apenas as variáveis $x_{p,c}$ e $\omega_{p,\psi,c}$ em que $\Upsilon[c] = i$ para a disciplina $c \in C$ são consideradas, ou seja, possuem tamanho reduzido em relação à mesma restrição no problema original.

Na Figura 6 é exibido o grafo G_C da instância *Toy* apresentada anteriormente. Observe que o grafo G_C foi particionado em 2 subproblemas ($k = 2$), SP_1 e SP_2. As disciplinas *ArcTec* e *SceCosC* pertencem a SP_1, enquanto as disciplinas *TecCos* e *GeoTec* pertencem a SP_2. Dessa forma, os conjuntos de restrições (3.18)−(3.23) e (3.26) referentes às disciplinas *ArcTec* e *SceCosC* fazem parte apenas de SP_1 e às disciplinas *TecCos* e *GeoTec*, apenas de SP_2. Em relação aos conjuntos de restrições (3.24) e (3.25), em SP_1 não haverá nenhuma restrição desses conjuntos, uma vez que não há nenhum currículo $u \in U$ em que todas as disciplinas $c \in C_u$ tenham $\Upsilon[c] = 1$. Já em SP_2 os conjuntos de

restrições terão apenas as restrições referentes ao currículo *Cur2*, pois todas as disciplinas $c \in C_{Cur2}$ possuem $\Upsilon[c] = 2$.

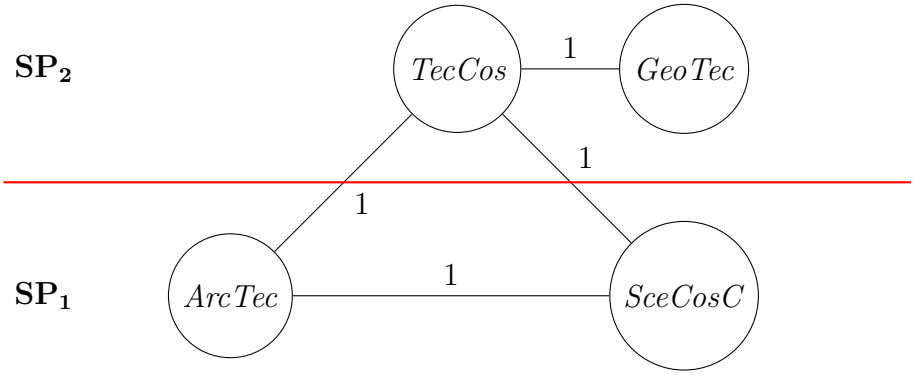

Figura 6 – Grafo G_C da instância *Toy* particionado em 2 subproblemas ($k = 2$).

Na Figura 7 é apresentado o grafo G_C após o particionamento e a criação dos subproblemas. Como no particionamento apresentado na Figura 6 duas arestas representando o currículo *Cur1* foram cortadas, ao gerar os subproblemas esse currículo não é considerado em nenhum deles. Por isso, o clique de tamanho 3 que representa o currículo *Cur1* é apresentado na Figura 7 com arestas tracejadas.

Por fim, é importante ressaltar que fazem parte da função objetivo de SP_2 as variáveis $z_{u,p}$, q_c e $\omega_{p,\psi,c}$ em que $u \in \{Cur2\}$ e $c \in \{TecCos, GeoTec\}$, e de SP_1, as variáveis q_c e $\omega_{p,\psi,c}$ em que $c \in \{SceCosC, ArcTec\}$.

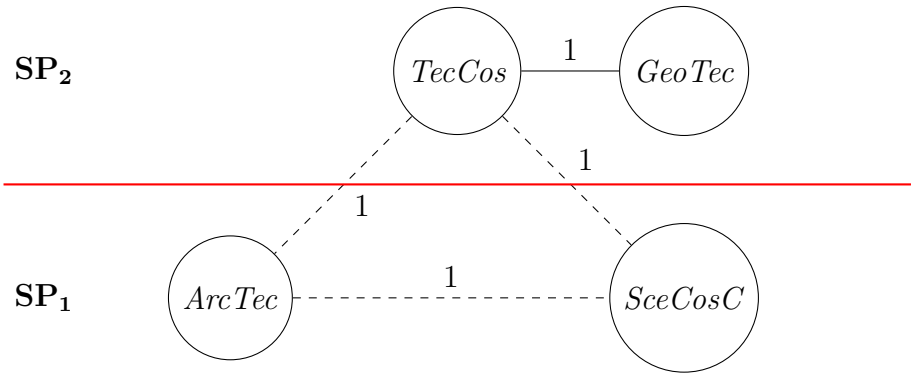

Figura 7 – Grafo G_C da instância *Toy* – Arestas tracejadas representam o currículo *Cur1*.

4.4 Métodos com variáveis de cópia

Nesta seção são apresentados mais dois métodos, denominados *C-partition-null* e *C-partition-org*, baseados no método *C-partition* apresentado na Seção 4.3.2, para encontrar valores de limitantes para a formulação CB-CTT do ITC-2007. Os métodos também incorporam alguns conceitos do método *U-partition* apresentado na Seção 4.3.1.

Conforme apresentado na Seção 3.1.4, o trabalho de Hao e Benlic (2011) também

encontrou valores de limitantes para a formulação CB-CTT do ITC-2007 usando o príncípio *dividir para conquistar*. Da mesma forma que no método *C-partition*, Hao e Benlic (2011) particionam o grafo G_C e os currículos representados por uma ou mais arestas cortadas durante o particionamento são desconsiderados nos subproblemas. Apesar das similaridades, o método *C-partition* se diferencia do método apresentado por Hao e Benlic (2011) por usar a biblioteca *METIS* para particionar o grafo G_C, enquanto Hao e Benlic (2011) implementaram a heurística *Busca Tabu Iterativa* para esse propósito. Assim, os métodos apresentados nesta seção dão continuidade ao método *C-partition* e ao método de Hao e Benlic (2011), usando variáveis de cópia para que os currículos representados por uma ou mais arestas cortadas também sejam considerados nos subproblemas.

Os métodos *C-partition-null* e *C-partition-org* são descritos conjuntamente a seguir, pois possuem apenas uma pequena diferença, que é o valor dos coeficientes das variáveis de cópia na função objetivo dos subproblemas. Dessa forma, os métodos *C-partition-null* e *C-partition-org* particionam o grafo $G_C = (V, A)$ em k classes com a função *METIS_PartGraphRecursive*, invocada através da API da biblioteca *METIS*, sendo que cada classe tem aproximadamente $|V|/k$ vértices e o particionamento encontrado minimiza a soma dos pesos das arestas cortadas que formam A^-. Além disso, os métodos geram cada subproblema SP_i, $i = 1, \ldots, k$. Para gerar os subproblemas, da mesma forma que nos métodos *U-partition* e *C-partition*, é usado o modelo matemático da primeira fase proposto por Lach e Lübbecke (2012).

Semelhantemente ao método *C-partition*, $\Upsilon[c]$ representa em qual classe (subproblema) a disciplina $c \in C$ pertence e o conjunto de arestas cortadas A^- representa os currículos que possuem disciplinas em mais de um subproblema. Assim, se o currículo $u \in U$ não é representado por alguma aresta em A^-, isso indica que todas as disciplinas $c \in C_u$ pertencem ao mesmo subproblema e o currículo u é considerado na função objetivo e nas restrições desse subproblema.

A principal diferença dos métodos *C-partition-null* e *C-partition-org* para o método *C-partition* é em relação aos currículos representados por uma ou mais arestas de A^-. No método *C-partition*, se um currículo $u \in U$ é representado por uma ou mais arestas de A^-, os conjuntos de restrições (3.24) e (3.25) referentes ao currículo u são desconsiderados em todos os subproblemas. Já nos métodos apresentados nesta seção, se um currículo $u \in U$ é representado por uma ou mais arestas de A^-, o currículo u é considerado no subproblema SP_i que tiver mais disciplinas $c \in C_u$ com $\Upsilon[c] = i$. Assim, os conjuntos de restrições (3.24) e (3.25) referentes ao currículo u são representados com as variáveis $z_{u,p}$ e $\varphi_{u,p}$ no subproblema SP_i. Consequentemente, as variáveis $z_{u,p}$ são também consideradas na função objetivo do subproblema SP_i.

No entanto, para considerar o currículo $u \in U$ representado por uma ou mais arestas de A^- no subproblema SP_i, é necessário que todas as disciplinas $c \in C_u$ também sejam

consideradas em SP_i. Para isso, os métodos *C-partition-null* e *C-partition-org* também consideram as variáveis $\overline{x^i}_{p,c}$, $\overline{\omega^i}_{p,\psi,c}$, $\overline{v^i}_{d,c}$ e $\overline{q^i}_c$, referentes à disciplina $c \in C_u$ em que $\Upsilon[c] = j$, $i \neq j$, nas restrições e função objetivo do subproblema SP_i. Nesse caso, as variáveis $\overline{x^i}_{p,c}$, $\overline{\omega^i}_{p,\psi,c}$, $\overline{v^i}_{d,c}$ e $\overline{q^i}_c$ são denominadas variáveis de cópia, pois as variáveis $x_{p,c}$, $\omega_{p,\psi,c}$, $v_{d,c}$ e q_c são consideradas no subproblema SP_j.

Diante disso, os métodos *C-partition-null* e *C-partition-org* apresentam uma relaxação mais forte do que o método *C-partition*, uma vez que os conjuntos de restrições (3.24) e (3.25) referentes a cada currículo $u \in U$ são considerados em um subproblema. Além disso, os conjuntos de restrições (3.19), (3.21) e (3.26) continuam possuindo tamanho reduzido em relação à mesma restrição no problema original, mas tamanho superior do que apresentado no método *C-partition*.

Em relação as variáveis $z_{u,p}$ é importante destacar que cada uma delas é definida para um currículo $u \in U$, e cada currículo é considerado em apenas um subproblema, que neste caso é o subproblema que mais tem disciplinas daquele currículo. Dessa maneira, cada variável $z_{u,p}$ também é considerada em apenas um subproblema e isso indica que as variáveis $z_{u,p}$ não possuem variáveis de cópia, ou seja, os seus coeficientes na função objetivo dos subproblemas serão sempre iguais ao do problema original, que é W^{CC}.

As variáveis $\omega_{p,\psi,c}$ e q_c são consideradas na função objetivo do subproblema SP_i, com coeficientes W^{RC} e W^{MD}, respectivamente, se $\Upsilon[c] = i$. Caso contrário, se $\Upsilon[c] \neq i$, mas $c \in C_u$ e o subproblema SP_i é o subproblema que mais tem disciplinas do currículo u, então as variáveis de cópia $\overline{\omega^i}_{p,\psi,c}$ e $\overline{q^i}_c$ referentes à disciplina c são consideradas na função objetivo do subproblema SP_i.

A diferença entre os métodos *C-partition-null* e *C-partition-org* é a definição dos coeficientes das variáveis de cópia, $\overline{\omega^i}_{p,\psi,c}$ e $\overline{q^i}_c$, na função objetivo de cada subproblema SP_i em que a disciplina c não pertence ($\Upsilon[c] \neq i$), mas é considerada. No método *C-partition-null* o coeficiente das variáveis de cópia $\overline{\omega^i}_{p,\psi,c}$ e $\overline{q^i}_c$ é igual a 0 (zero). Já no método *C-partition-org* o coeficiente das variáveis de cópia $\overline{\omega^i}_{p,\psi,c}$ e $\overline{q^i}_c$ é igual a W^{RC} e W^{MD}, respectivamente.

Portanto, no método *C-partition-org*, após a solução de cada subproblema SP_i, $i = 1, \ldots, k$, pelo otimizador CPLEX, o valor obtido da função objetivo do subproblema SP_i é decrementado em $W^{RC} \cdot \theta_{p,\psi,c} \cdot \overline{\omega^i}_{p,\psi,c}$ e $W^{MD} \cdot \overline{q^i}_c$ unidades, respectivamente, para cada variável de cópia $\overline{\omega^i}_{p,\psi,c}$ e $\overline{q^i}_c$. Dessa maneira, o método *C-partition-org* possui uma relaxação mais forte que o método *C-partition-null*, pois cada subproblema é mais parecido com o problema original.

Como apresentado na Figura 6 e descrito na Seção 4.3.2, ao particionar o grafo G_C da instância *Toy* em 2 subproblemas ($k = 2$), as arestas $\{TecCos, ArcTec\}$ e $\{SceCosC, TecCos\}$ pertencem ao conjunto A^- de arestas cortadas. Na Figura 8 é exibida

a representação do grafo G_C da instância *Toy* após o particionamento em 2 subproblemas ($k = 2$) com um dos métodos apresentados nesta seção.

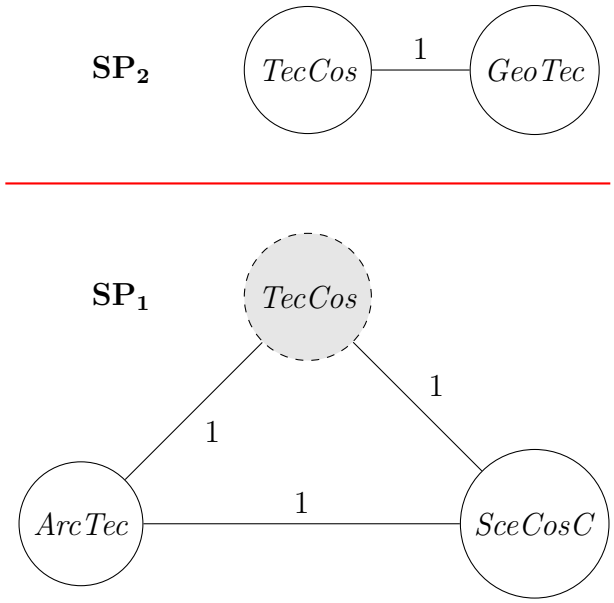

Figura 8 – Grafo G_C da instância *Toy* − Vértice tracejado representa as variáveis de cópia da disciplina *TecCos*.

Ao observar a Figura 8 é possível notar que há dois vértices representando a disciplina *TecCos*. Isso ocorre pois essa disciplina pertence ao subproblema SP_2, mas também pertence ao currículo *Cur1* que possui duas outras disciplinas (*ArcTec* e *SceCosC*) no subproblema SP_1. Assim, o currículo *Cur1* é considerado completamente no SP_1. As variáveis $x_{p,c}$, $\omega_{p,\psi,c}$, $v_{d,c}$ e q_c referentes à disciplina *TecCos* serão consideradas em SP_2, e as variáveis de cópia $\overline{x^1}_{p,c}$, $\overline{\omega^1}_{p,\psi,c}$, $\overline{v^1}_{d,c}$ e $\overline{q^1}_c$, também referentes à disciplina *TecCos*, são consideradas em SP_1.

4.5 Fluxograma dos métodos propostos

Na Figura 9 é exibido um fluxograma para os métodos *U-partition*, *C-partition*, *C-partition-null* e *C-partition-org*. Nesse fluxograma, cada etapa de execução dos métodos propostos é representada.

A primeira etapa da execução consiste na entrada dos dados da instância e do número de subproblemas k. O tempo limite de execução do otimizador CPLEX também pode ser um dado de entrada. Caso esse valor não seja fornecido, então o CPLEX não terá tempo limite de execução. Após a entrada dos dados, o problema é então representado a partir do grafo G_U (*U-partition*) ou G_C (*C-partition*, *C-partition-null* e *C-partition-org*).

Em seguida, o grafo é particionado em k subproblemas. Nessa etapa é usada a biblioteca *METIS*, apresentada na Seção 4.2, para particionar o grafo. O particionamento

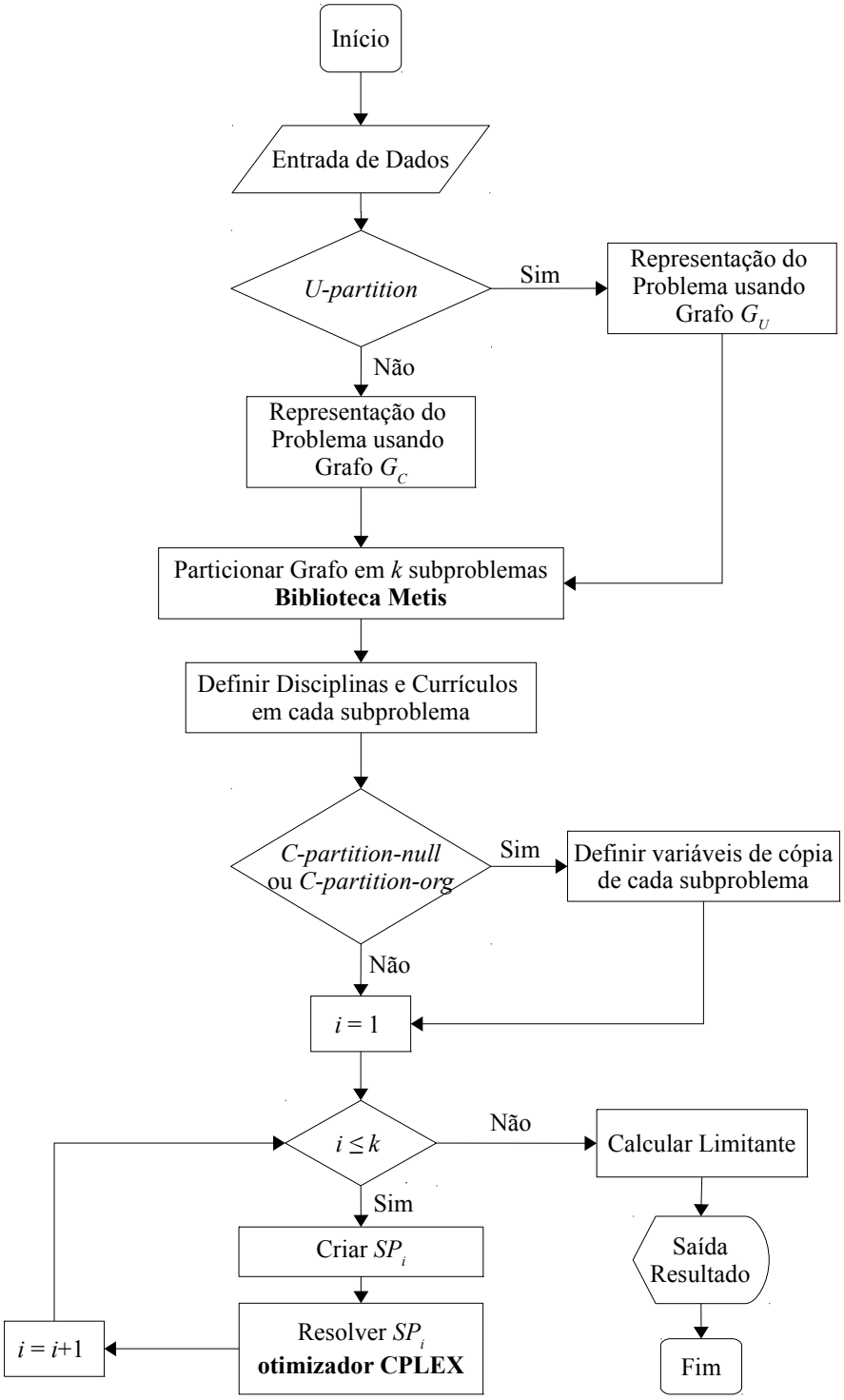

Figura 9 – Fluxograma dos métodos propostos.

obtido (Υ) é usado para definir em qual subproblema cada disciplina e currículo será representado, seguindo os critérios do método que está sendo executado. É importante destacar que essa etapa é executada nos quatro métodos propostos. No entanto, os métodos *C-partition-null* e *C-partition-org*, que usam variáveis de cópia, possuem uma

etapa adicional, que consiste em definir quais disciplinas serão representadas por tais variáveis e em quais subproblemas.

Por fim, iterativamente os subproblemas são criados e, em seguida, resolvidos pelo otimizador CPLEX, cujo tempo limite de execução pode ser definido de acordo com parâmetro na entrada de dados. Após a resolução de todos os subproblemas, o valor do limitante inferior é calculado (soma das soluções dos subproblemas obtidas pelo CPLEX) e exibido, terminando assim a execução do método.

5 Experimentos e Resultados

Os métodos *U-partition* e *C-partition*, descritos respectivamente nas Seções 4.3.1 e 4.3.2, além dos métodos *C-partition-null* e *C-partition-org*, descritos na Seção 4.4, foram submetidos a experimentos computacionais executados nas 21 instâncias (comp01-comp21) do ITC-2007 para a formulação CB-CTT. As instâncias estão disponíveis para *download* no *website* da competição (http://www.cs.qub.ac.uk/itc2007).

Neste capítulo são apresentadas inicialmente algumas características importantes das instâncias usadas nos experimentos computacionais (Seção 5.1). Na Seção 5.2 são apresentados os detalhes de implementação dos quatro métodos propostos. O ambiente computacional usado para realizar os experimentos computacionais e os parâmetros necessários para execução dos métodos também são descritos nessa seção. Por fim, na Seção 5.3 são especificados os experimentos computacionais realizados, além de apresentar, analisar e comparar os resultados obtidos com os resultados encontrados nos trabalhos descritos na Seção 3.1.

5.1 O conjunto de instâncias comp01-comp21 (CB-CTT/ITC-2007)

Desde o fim do ITC-2007 as instâncias da formulação CB-CTT são usadas como referência (*benchmark*) por trabalhos que abordam o PTHU. Alguns desses trabalhos também se dedicam em entender ou descobrir características dessas instâncias, que por sua vez podem ajudar no processo de calibração de parâmetros dos seus métodos. Os organizadores do ITC-2007 (GASPERO; MCCOLLUM; SCHAERF, 2007) foram os primeiros a apresentar as principais características dessas instâncias, conforme descrito no Capítulo 2. Posteriormente, um trabalho que se destacou nessa iniciativa foi o de Bellio et al. (2016) (descrito na Seção 3.2), que após uma extensa análise baseada em princípios estatísticos identificou que as principais características que definem a dificuldade em se encontrar uma solução viável para uma determinada instância são a quantidade de disciplinas e currículos.

Neste trabalho as principais características das instâncias foram estudadas antes e após a realização dos experimentos computacionais. O objetivo desses estudos foi mensurar a dificuldade do otimizador CPLEX em encontrar uma solução para os subproblemas e, de certa forma, identificar características que impactam o desempenho dos métodos propostos.

A partir disso, usando como base os trabalhos de Gaspero, McCollum e Schaerf (2007) e Bellio et al. (2016), foi possível perceber que as instâncias comp01 e comp11 são as mais fáceis de serem resolvidas, pois apresentam menos disciplinas e currículos que as

demais. Em contrapartida, as instâncias comp02, comp05 e comp12 foram consideradas as mais difíceis, já que possuem muitos currículos e disciplinas, além de uma alta taxa de conflitos e grande quantidade de indisponibilidades de alocação das disciplinas. Esse fato pode ser comprovado ao somar a quantidade de currículos e disciplinas de cada instância. A instância comp11 possui o menor valor dentre as 21 instâncias: 43 (30 disciplinas e 13 currículos). Já a instância comp12 é a que apresenta o maior valor dentre as instâncias: 238 (88 disciplinas e 150 currículos).

Por fim, a média de currículos por disciplinas também foi levada em consideração nos estudos realizados. Essa medida indicou, por exemplo, que apenas três instâncias (comp05, comp12 e comp18) possuem mais currículos que disciplinas. Dessa maneira, é esperado que os métodos *C-partition*, *C-partition-null* e *C-partition-org*, que particionam o grafo G_C, e o método *U-partition*, que particiona o grafo G_U, tenham comportamento diferente nessas instâncias.

5.2 Detalhes de implementação e aplicação dos métodos propostos

Os métodos *U-partition*, *C-partition*, *C-partition-null* e *C-partition-org* foram implementados com a linguagem de programação C++.

Na implementação dos métodos foram utilizados quatro vetores para representar respectivamente as informações das disciplinas, currículos, professores e salas. Além disso, duas matrizes auxiliares, cada uma com duas dimensões, também são criadas. A primeira matriz, denominada *matResDisPer*, possui dimensão $|C| \times |P|$ e $matResDisPer[c][p]$ é igual a 1 se a disciplina $c \in C$ é indisponível no período $p \in P$, e 0 caso contrário. Já a segunda matriz, denominada *matDisTur*, possui dimensão $|C| \times |U|$. Da mesma forma que a matriz anterior, $matDisTur[c][u]$ é igual a 1 se a disciplina $c \in C_u$, e 0 caso contrário. Essas matrizes facilitam o processo de criação dos grafos G_C e G_U e também dos subproblemas.

A partir das informações do problema, armazenadas em variáveis ou nas estruturas de dados já citadas, se define o vetor *vetTamSal* de dimensão *numSalDif*, que armazena os diferentes tamanhos de salas ordenados crescentemente. O parâmetro *numSalDif* representa a quantidade de salas com tamanhos distintos. O vetor *vetTamSal* e o valor *numSalDif* também facilitam o processo de criação dos subproblemas, pois são usados na definição dos valores $\theta_{p,\psi,c}$, presentes nas funções objetivo dos subproblemas, conforme apresentado na Seção 3.1.3.

Os grafos G_C e G_U são particionados, ou seja, divididos em subproblemas, com a API da biblioteca *METIS*, versão 5.1.0 64 bits. Os modelos matemáticos dos subproblemas foram resolvidos com o otimizador CPLEX 12.8 64 bits para Windows, sendo que as configurações padrão do otimizador CPLEX foram usadas nos quatro métodos propostos.

Os experimentos computacionais foram realizados em um computador Dell Precision T5810 com processador Intel Xeon® E5-1650 v3 (6C - 3.5 GHz) e memória RAM de 64 GB.

Os quatro métodos propostos neste trabalho possuem três parâmetros, sendo que dois deles são obrigatórios: a instância e o número de subproblemas k. O terceiro parâmetro é opcional e consiste no tempo limite de execução do CPLEX para resolver cada um dos subproblemas. Esse parâmetro é opcional já que se não for informado o CPLEX não terá o seu tempo de execução limitado.

5.3 Experimentos computacionais e análise dos resultados

Os primeiros experimentos computacionais realizados nesta tese tiveram como objetivo definir o modelo matemático a ser usado nos métodos propostos, e também verificar a qualidade do particionamento obtido pela biblioteca *METIS* quando comparado com o particionamento obtido pela heurística *Busca Tabu Iterativa*, proposta por Hao e Benlic (2011).

Inicialmente, para definir o modelo matemático a ser usado para criar os subproblemas nos métodos propostos, o modelo Monolítico de Burke et al. (2010) (ver Seção 3.1.1) e o modelo da primeira fase de Lach e Lübbecke (2012) (ver Seção 3.1.3) foram escolhidos para serem usados nesse experimento. Essa escolha levou em consideração, além das diferenças que esses modelos apresentam entre si, o fato deles terem sido os dois primeiros modelos propostos para o PTHU da formulação CB-CTT. Além disso, os demais modelos, propostos posteriormente, são baseados neles.

É importante destacar mais uma vez as diferenças entre o modelo Monolítico de Burke et al. (2010) e o modelo da primeira fase de Lach e Lübbecke (2012). O modelo Monolítico de Burke et al. (2010) é um modelo completo, pois considera as quatro restrições fracas (**S1**-**S4**) da formulação CB-CTT, enquanto que o modelo da primeira fase de Lach e Lübbecke (2012) considera apenas as três primeiras restrições fracas (**S1**-**S3**). Entretanto, o modelo Monolítico possui mais variáveis e restrições que o modelo proposto por Lach e Lübbecke (2012), e isso aumenta a sua dificuldade de resolução.

Dessa forma, nesse primeiro experimento, cada instância da formulação CB-CTT foi representada com o modelo Monolítico de Burke et al. (2010), sendo que nesse caso o problema não foi particionado, ou seja, o problema original de cada instância foi representado com o modelo Monolítico. Em seguida, cada instância foi resolvida com o otimizador CPLEX, cujo tempo limite de execução foi definido em 40τ. O valor τ é definido de acordo com as regras do ITC-2007, ou seja, é definido pelo programa *benchmark*, disponível no *website* da competição e apresentado na Seção 1.1. O programa *benchmark* analisa as configurações do computador com o objetivo de informar a quantidade de

tempo que seria equivalente ao dos computadores do campeonato. Ao executar o programa *benchmark*, no computador utilizado para os experimentos computacionais deste trabalho, foi estipulado o valor τ igual a 247 segundos. Assim, o tempo de execução do CPLEX foi limitado em 9880 segundos (40×247).

Após essa execução, o experimento foi novamente realizado, mas dessa vez usando o modelo da primeira fase de Lach e Lübbecke (2012) para representar o problema original de cada instância.

Ao final de cada execução, desse experimento, o *gap* de otimalidade ($\%gap$) foi calculado pela fórmula $(ub - lb)/ub \times 100$, sendo que ub e lb são, respectivamente, o melhor valor de limitante superior (solução viável) atualmente conhecido e o valor de limitante inferior obtido.

É importante citar que o melhor valor de limitante superior (ub), atualmente conhecido para cada instância, é apresentado no trabalho de Kiefer, Hartl e Schnell (2017). Esses valores também são usados na comparação de resultados realizada por Bagger, Sørensen e Stidsen (2019), sendo esse o trabalho mais recente da literatura que apresenta valores de limitantes inferiores para o PTHU da formulação CB-CTT.

Na Tabela 1 são apresentados os resultados, valores de limitantes inferiores (lb), obtidos ao resolver o problema original de cada instância, representado com o modelo Monolítico de Burke et al. (2010) e com o modelo da primeira fase de Lach e Lübbecke (2012). O melhor valor de limitante inferior encontrado para cada instância é destacado em **negrito**. Nessa tabela é apresentado o melhor valor de limitante superior (ub) atualmente conhecido para cada uma das 21 instâncias. Ao lado direito de cada valor de limitante inferior, encontra-se o valor do *gap* de otimalidade ($\%gap$) entre ub e o respectivo limitante inferior. Por fim, nessa tabela também é apresentado o tempo de resolução (T($seg.$)) em segundos, para cada instância, com cada um dos modelos.

Ao analisar os resultados apresentados na Tabela 1 é possível notar que em média os resultados obtidos com o modelo da primeira fase de Lach e Lübbecke (2012) foram 30,36% melhores do que os encontrados com o modelo Monolítico de Burke et al. (2010). Além disso, em apenas uma instância (comp01) o resultado do modelo Monolítico foi melhor.

O modelo da primeira fase de Lach e Lübbecke (2012) também se mostrou mais eficiente ao encontrar, para todas instâncias, o respectivo valor de limitante inferior com tempo menor ou igual ao obtido pelo modelo Monolítico de Burke et al. (2010). Com isso, o modelo de Lach e Lübbecke (2012) encontrou os valores de limitantes inferiores, em média, 23,45% mais rápido que o modelo Monolítico.

Dessa maneira, o modelo da primeira fase de Lach e Lübbecke (2012) foi escolhido para representar os subproblemas nos métodos *U-partition*, *C-partition*, *C-partition-null* e

C-partition-org, que são propostos nesta tese.

Tabela 1 – Comparação de limitantes inferiores obtidos pelos modelos matemáticos propostos por Burke et al. (2010) e Lach e Lübbecke (2012).

Instâncias	ub[1]	BMPR10[2]			LL12[3]		
		lb	*%gap*	T(*seg.*)	*lb*	*%gap*	T(*seg.*)
comp01	5	**5**	0,00	222	4	20,00	2
comp02	24	-20	183,33	9880	**15**	37,50	9880
comp03	64	5	92,19	9880	**49**	23,44	9880
comp04	35	4	88,57	9880	**35**	0,00	986
comp05	284	134	52,82	9880	**209**	26,41	9880
comp06	27	-30	211,11	9880	**23**	14,81	9880
comp07	6	-66	1200,00	9880	**6**	0,00	9880
comp08	37	2	94,59	9880	**37**	0,00	9880
comp09	96	22	77,08	9880	**96**	0,00	4947
comp10	4	-40	1100,00	9880	**4**	0,00	9880
comp11	0	**0**	0,00	524	**0**	0,00	1
comp12	294	59	79,93	9880	**128**	56,46	9880
comp13	59	18	69,49	9880	**59**	0,00	5827
comp14	51	8	84,31	9880	**51**	0,00	275
comp15	62	5	91,94	9880	**49**	20,97	9880
comp16	18	-49	372,22	9880	**18**	0,00	3741
comp17	56	-5	108,93	9880	**47**	16,07	9880
comp18	61	8	86,89	9880	**44**	27,87	9880
comp19	57	25	56,14	9880	**57**	0,00	44
comp20	4	-55	1475,00	9880	**4**	0,00	9880
comp21	74	2	97,30	9880	**66**	10,81	9880
Média		1,52	267,71	8974,57	**47,67**	12,11	6869,66

[*] Melhores valores de limitantes inferiores (*lb*) destacados em **negrito**
[1] Melhores valores de soluções viáveis (limitantes superiores) - Kiefer, Hartl e Schnell (2017)
[2] BMPR10 - Burke et al. (2010) - Modelo Monolítico.
[3] LL12 - Lach e Lübbecke (2012) - Modelo matemático da primeira fase.

O segundo experimento computacional teve como objetivo verificar a qualidade do particionamento obtido pela biblioteca *METIS* quando comparado com o particionamento obtido pela heurística *Busca Tabu Iterativa*, proposta por Hao e Benlic (2011). Para isso, o método *C-partition*, apresentado na Seção 4.3.2, foi executado em todas as 21 instâncias da formulação CB-CTT, com o valor de k em que a abordagem *dividir para conquistar* de Hao e Benlic (2011) (ver Seção 3.1.4) apresentou os melhores resultados.

Esse experimento foi realizado dessa maneira pois o método proposto por Hao e Benlic (2011) e o *C-partition* são muito similares. Ambos os métodos particionam o problema original, representado pelo grafo G_C, em subproblemas independentes que são então representados com o modelo matemático da primeira fase proposto por Lach e Lübbecke (2012). Além disso, nos dois métodos as restrições representadas pelas arestas

cortadas durante o particionamento são relaxadas. A soma dos valores encontrados ao resolver os subproblemas com um otimizador representa, em ambos os métodos, um valor de limitante inferior para o problema original.

No entanto, os métodos apresentam algumas diferenças entre si, sendo a principal delas a maneira como o partionamento de G_C é obtido, já que no *C-partition* o particionamento é obtido com a biblioteca *METIS* e no método de Hao e Benlic (2011) é usado a heurística *Busca Tabu Iterativa*.

Além dessa diferença, é importante citar também que o grafo G_C no método *C-partition* é valorado e o tamanho de duas classes do particionamento difere em no máximo uma unidade, enquanto que no método proposto por Hao e Benlic (2011) o grafo não é valorado e a diferença de tamanho entre duas classes pode ser maior que uma unidade. Por fim, outra diferença é que o método proposto por Hao e Benlic (2011) limitou o tempo de execução do otimizador COIN-OR em 25200 segundos (7 horas), por subproblema, sem apresentar uma justificativa para a escolha desse valor, enquanto que no método *C-partition* o tempo limite de execução do otimizador CPLEX foi de 40τ (9800 segundos) por subproblema.

Assim, apesar dessas diferenças, e dos métodos terem sido executados com diferentes otimizadores e em computadores com diferentes configurações, as similaridades entre eles permitem verificar a qualidade do particionamento através da comparação dos resultados obtidos.

Tabela 2 – Comparação de limitantes inferiores obtidos pelo método *C-partition* com método proposto por Hao e Benlic (2011).

Instâncias	k	HB11	*C-partition*	Instâncias	k	HB11	*C-partition*
comp01	2	**4**	**4**	comp12	5	109	**120**
comp02	6	**12**	**12**	comp13	4	**59**	**59**
comp03	6	38	**49**	comp14	3	**51**	**51**
comp04	5	**35**	**35**	comp15	5	38	**50**
comp05	4	**183**	152	comp16	4	**16**	**16**
comp06	5	**22**	20	comp17	5	48	**49**
comp07	4	**6**	**6**	comp18	4	24	**37**
comp08	3	**37**	**37**	comp19	3	**56**	**56**
comp09	6	72	**95**	comp20	5	**2**	**2**
comp10	6	**4**	**4**	comp21	5	61	**64**
comp11	4	**0**	**0**	Média		41,76	**43,71**

Na coluna HB11 da Tabela 2 são apresentados os resultados do método proposto por Hao e Benlic (2011) em comparação com os resultados obtidos pelo método *C-partition*. Na coluna k da Tabela 2 são apresentados os valores de k em que o método de Hao e

Benlic (2011) obteve o melhor valor de limitante inferior para cada instância. Esse valor de k também foi usado no método *C-partition*. Os valores apresentados em destaque são os melhores valores obtidos na comparação entre os métodos. No caso de empate, ambos os valores são apresentados em **negrito**.

Dessa maneira, analisando os valores apresentados na Tabela 2, percebe-se que em 12 instâncias o valor obtido pelo método *C-partition* foi igual ao resultado do método proposto por Hao e Benlic (2011), sendo que em outras sete instâncias o resultado do *C-partition* foi melhor. Com isso, o método *C-partition* encontrou resultados, em média, 4,68% melhores que os obtidos pela abordagem *dividir para conquistar* de Hao e Benlic (2011).

Portanto, se verifica dessa forma a qualidade do particionamento obtido pela biblioteca *METIS*, já que ambos os métodos particionam, para cada instância, o grafo G_C em k subproblemas e os resultados obtidos pelo método *C-partition*, apresentados na Tabela 2, são em média melhores que os obtidos por Hao e Benlic (2011).

Após a realização desses dois experimentos computacionais e o término da implementação dos métodos *U-partition*, *C-partition-null* e *C-partition-org*, novos experimentos computacionais foram realizados com os quatro métodos propostos nesta tese. Esses experimentos podem ser divididos em duas etapas. O objetivo da primeira etapa é definir o melhor valor de limitante inferior e o respectivo número de subproblemas k, para cada instância e cada método aplicado. Para isso, cada instância e método foram executados com k variando de 2 até δ, sendo que o valor δ é o primeiro valor de k em que o tempo de execução do otimizador CPLEX foi menor que 40τ para cada um dos subproblemas.

Etapa 1:

A primeira etapa de testes foi realizada limitando o tempo de execução do CPLEX em 40τ, pois nos métodos apresentados por Burke et al. (2010), Lach e Lübbecke (2012), Cacchiani et al. (2013) e Bagger et al. (2019), o tempo de execução do otimizador também foi limitado em 40τ.

Em todos os experimentos computacionais realizados, ou seja, em cada execução dos métodos propostos, foi anotado o valor do limitante inferior obtido (que corresponde à soma dos valores obtidos pelo CPLEX ao resolver cada subproblema), a quantidade de tempo para particionar o grafo, o tempo necessário para criar os subproblemas e o tempo de resolução pelo CPLEX de todos os subproblemas.

Após o término da primeira etapa de testes, foram desprezadas as informações relativas à quantidade de tempo necessária para particionar os grafos G_U (*U-partition*) ou G_C (*C-partition*, *C-partition-null* e *C-partition-org*), e para criar os subproblemas, uma vez que em todas as execuções realizadas (na execução de cada método para cada instância e para cada valor de k ($k = 2, ..., \delta$)), os tempos obtidos foram inferiores a um segundo.

Destaca-se com isso a eficiência das heurísticas de particionamento de grafos implementadas na biblioteca *METIS*, para o particionamento do problema abordado, quando comparado ao tempo da heurística *Busca Tabu Iterativa*, proposta por Hao e Benlic (2011) para particionar o grafo G_C, que em algumas instâncias foi próximo de 600 segundos.

Portanto, considerando que os objetivos das heurísticas de particionamento de grafos implementadas na biblioteca *METIS* e da *Busca Tabu Iterativa* de Hao e Benlic (2011) são os mesmos — encontrar um particionamento do grafo G_C que minimize o número de arestas cortadas — pode-se afirmar que a utilização da biblioteca *METIS* é vantajosa.

Com respeito ao valor de k: Cada um dos quatro métodos foi executado inicialmente com $k = 2$ para cada instância. Nos casos em que a execução do CPLEX, para algum subproblema e instância, foi interrompida por atingir o tempo máximo de 9880 segundos (40τ), a mesma foi repetida, mas com $k = 3$. Esse procedimento foi iterativamente realizado, incrementando o valor de k, até que a execução do CPLEX não tenha sido interrompida. O último valor de k foi definido como δ. A partir daí identifica-se o valor de k, dentre as $\delta - 1$ execuções, para o qual foi obtido o maior limitante inferior para o problema. Esse valor é denominado γ, $2 \leq \gamma \leq \delta$, e pode ser usado na Etapa 2.

O tempo total de execução de cada método, para cada instância, é calculado como a soma do tempo gasto em cada uma das $\delta - 1$ execuções.

Na Tabela 3 são apresentados os valores de γ e δ, para cada instância, para os métodos *U-partition*, *C-partition*, *C-partition-null* e *C-partition-org*. Nessa tabela, o valor de δ é apresentado entre parêntesis, mas apenas para os casos em que $\gamma \neq \delta$. Para os demais casos ($\gamma = \delta$) o valor apresentado representa γ e δ.

Ao observar os valores de γ e δ apresentados na Tabela 3, nota-se que em 18 das 21 instâncias, para todos os quatro métodos propostos, os valores de γ e δ coincidem e são iguais a 2 ($\gamma = \delta = 2$). Ou seja, particionando o problema original em apenas dois subproblemas foi possível encontrar o valor do limitante inferior dentro do prazo estipulado de 9880 segundos (40τ) para cada subproblema. Já nas outras três intâncias (comp02, comp05 e comp12), que são as instâncias consideradas mais difíceis (ver Seção 5.1), nem sempre o melhor valor de limitante foi obtido com $\gamma = 2$, sendo que em alguns desses casos γ difere de δ.

De acordo com a Tabela 3, na primeira etapa de testes foram realizadas 106 execuções dos métodos propostos, sendo que 79,25% dessas execuções foi particionando o problema original em dois subproblemas ($k = 2$). Isso sugere que a estratégia de dividir o PTHU da formulação CB-CTT inicialmente em dois subproblemas foi eficiente, pois na maioria das vezes isso foi suficiente para diminuir a complexidade do problema original e

Tabela 3 – Número de subproblemas γ que obteve o melhor valor de limitante inferior e o valor máximo de k (δ).

Instâncias	*U-partition*	*C-partition*	*C-partition-null*	*C-partition-org*
	$\gamma(\delta)$	$\gamma(\delta)$	$\gamma(\delta)$	$\gamma(\delta)$
comp01	2	2	2	2
comp02	2	3(4)	3(4)	3(4)
comp03	2	2	2	2
comp04	2	2	2	2
comp05	3(4)	2	3	3(4)
comp06	2	2	2	2
comp07	2	2	2	2
comp08	2	2	2	2
comp09	2	2	2	2
comp10	2	2	2	2
comp11	2	2	2	2
comp12	3	3(5)	4(5)	6
comp13	2	2	2	2
comp14	2	2	2	2
comp15	2	2	2	2
comp16	2	2	2	2
comp17	2	2	2	2
comp18	2	2	2	2
comp19	2	2	2	2
comp20	2	2	2	2
comp21	2	2	2	2

obter bons valores de limitantes inferiores dentro do prazo estipulado.

Conforme apresentado na Seção 5.1, as três instâncias da formulação CB-CTT que se destacam por serem de difícil resolução são: comp02, comp05 e comp12. Além de possuírem muitos currículos e disciplinas, essas instâncias se caracterizam por uma alta taxa de conflitos e grande quantidade de indisponibilidades de alocação das disciplinas (GASPERO; MCCOLLUM; SCHAERF, 2007), o que pode justificar a não obtenção de valores de limitantes inferiores, para essas instâncias, em alguns métodos, dentro do prazo estipulado e com $k = 2$ (ver Tabela 3).

Por fim, considerando ainda na Tabela 3 apenas as instâncias comp02, comp05 e comp12, é possível observar que em dez ocasiões o melhor valor de limitante não foi obtido particionando o problema original em dois subproblemas ($k = 2$), sendo que em sete desses casos o valor γ difere de δ. Isso indica que, para essas instâncias, quando não foi possível encontrar o valor de limitante inferior dentro do prazo estipulado, na maioria das vezes, não foi vantajoso dividir o problema original em mais subproblemas, pois apesar do tempo gasto para encontrar o valor de limitante inferior ter diminuído, a quantidade de restrições

relaxadas aumentou, impactando a qualidade do limitante.

Etapa 2:

Na segunda etapa dos experimentos computacionais os quatro métodos foram executados novamente, mas dessa vez apenas com os respectivos valores de $k = \gamma$ e com as instâncias com execução do CPLEX interrompida na Etapa 1, para algum subproblema, por ter atingido o tempo limite definido de 40τ. Portanto, a Etapa 2 se restringiu às instâncias comp02, comp05 e comp12 e à execução dos métodos em que $\gamma \neq \delta$, sem limitar o tempo de execução do CPLEX.

Na Seção 5.3.1 são apresentados os resultados dos experimentos computacionais realizados com os quatro métodos propostos. Os melhores resultados obtidos são então comparados com resultados de outros trabalhos da literatura (Seção 5.3.2).

5.3.1 Análise dos resultados obtidos

Na Tabela 3 são apresentados, para cada instância e método proposto, o valor de k (δ) para o qual o CPLEX encontrou a solução de cada subproblema sem extrapolar o tempo limite e também o valor de k no qual, dentre as execuções realizadas, foi obtido o melhor valor de limitante (γ, $2 \leq \gamma \leq \delta$). Já na Tabela 4 são apresentados os melhores valores de limitantes inferiores (lb) obtidos, para cada instância, pelos métodos *U-partition*, *C-partition*, *C-partition-null* e *C-partition-org*, limitando o tempo de execução do otimizador CPLEX para cada subproblema em 9880 segundos (40τ). Além disso, é apresentado também os valores de limitantes obtidos ao representar o problema original de cada instância, sem particionar (*No-partition*), com o modelo da primeira fase de Lach e Lübbecke (2012). Esses valores de limitantes já foram apresentados na Tabela 1, mas são apresentados novamente na Tabela 4 para facilitar a comparação. Os valores apresentados em destaque são os melhores valores obtidos na comparação entre os quatro métodos. No caso de empate, todos os valores do empate são apresentados em **negrito**.

A Tabela 4 também apresenta o tempo de resolução (T(*seg.*)) em segundos para cada instância. No caso dos quatro métodos propostos neste trabalho, esse valor representa a soma do tempo total gasto pelo CPLEX para resolver todos os subproblemas nas $\delta - 1$ execuções ($2 \leq k \leq \delta$), uma vez que o tempo para particionar o grafo (G_U ou G_C) e criar os subproblemas pode ser desconsiderado (em todos os casos foi menor que um segundo).

A partir dos resultados apresentados na Tabela 4 nota-se que em 12 das 21 instâncias o valor de limitante inferior obtido na resolução do problema original, sem particionar (*No-partition*), foi igual ao melhor valor obtido entre os quatro métodos propostos nesta tese. Entretanto, é importante observar que nessas instâncias os métodos propostos foram, em média, mais eficientes que a estratégia *No-partition*. Em apenas uma instância (comp18) o valor de limitante inferior obtido pela estratégia *No-partition* foi melhor do que o obtido

Tabela 4 – Limitantes inferiores obtidos pelos métodos propostos na Etapa 1.

Instâncias	*No-partition*		*U-partition*		*C-partition*		*C-partition-null*		*C-partition-org*	
	lb	T(*seg.*)	*lb*	T(*seg.*)	*lb*	T(*seg.*)	*lb*	T(*seg.*)	*lb*	T(*seg.*)
comp01	**4**	2	**4**	1	**4**	1	**4**	1	**4**	1
comp02	15	9880	16	1274	17	19825	17	19831	**18**	19852
comp03	49	9880	58	3073	**61**	2160	**61**	2182	**61**	2173
comp04	**35**	986	**35**	3	**35**	3	**35**	4	**35**	3
comp05	209	9880	139	20904	152	1981	198	17298	**229**	35408
comp06	23	9880	**27**	445	**27**	602	**27**	549	**27**	918
comp07	**6**	9880	**6**	8	**6**	5	**6**	7	**6**	7
comp08	**37**	9880	**37**	3	**37**	2	**37**	3	**37**	3
comp09	**96**	4947	**96**	4386	**96**	1267	**96**	3582	**96**	3582
comp10	**4**	9880	**4**	4	**4**	5	**4**	5	**4**	4
comp11	**0**	1	**0**	1	**0**	1	**0**	1	**0**	1
comp12	128	9880	135	28222	154	33987	173	44573	**182**	54807
comp13	**59**	5827	**59**	18	**59**	15	**59**	16	**59**	16
comp14	**51**	275	**51**	29	**51**	29	**51**	24	**51**	35
comp15	49	9880	58	3045	**61**	2195	**61**	2175	**61**	2158
comp16	**18**	3741	16	26	16	33	16	13	**18**	33
comp17	47	9880	**56**	556	**56**	542	**56**	753	**56**	628
comp18	**44**	9880	14	30	37	286	37	293	40	594
comp19	**57**	44	40	5	**57**	17	**57**	17	**57**	17
comp20	**4**	9880	**4**	382	**4**	221	**4**	455	**4**	307
comp21	66	9880	**74**	1822	**74**	1835	**74**	1866	**74**	1781
Média	47,67	6869,66	44,24	3058,90	48,00	3095,81	51,10	4459,43	**53,29**	5825,14

* Melhores valores de limitantes inferiores (*lb*) destacados em **negrito**

pelos quatro métodos propostos. Portanto, esses resultados indicam que a estratégia de particionar o problema original em subproblemas foi eficaz e eficiente em 20 das 21 instâncias.

Na Tabela 4 se observa, como exemplo, que os métodos *C-partition* e *C-partition-org* apresentaram valores de limitantes inferiores, em média, 0,69% e 11,79%, respectivamente, melhores do que os valores obtidos sem particionar o problema. Esses métodos além de apresentarem, em média, melhores resultados do que a estratégia *No-partition*, também encontraram os valores de limitantes inferiores, em média, 54,94% (*C-partition*) e 15,20% (*C-partition-org*) com menos tempo. A estratégia de não particionar o problema apresentou, em média, resultados melhores apenas quando comparado com o método *U-partition*. Entretanto, é importante frisar que o *U-partition* encontrou os valores de limitantes inferiores, em média, com menos da metade do tempo gasto pela estratégia *No-partition*. Além disso, em 17 das 21 instâncias os resultados do método *U-partition* foram melhores ou iguais ao *No-partition*.

Ao comparar os resultados dos quatro métodos propostos, que são apresentados na Tabela 4, nota-se que os limitantes inferiores obtidos pelo método *C-partition* foram melhores do que os obtidos pelo método *U-partition*. Apesar disso, a diferença percentual entre as médias dos resultados é de apenas 8,50%. Nessa tabela também é possível perceber

que os métodos *U-partition* e *C-partition* obtiveram o mesmo valor de limitante inferior em 14 das 21 instâncias. No entanto, nas outras 7 instâncias, o método *C-partition* obteve melhor valor de limitante inferior, com destaque para a instância comp18, na qual o valor de limitante inferior obtido pelo método *U-partition* foi 14, enquanto que no método *C-partition* foi 37. Em termos percentuais essa diferença é de 164,29%.

O desempenho superior do método *C-partition* se justifica pelo fato de ao particionar o grafo G_C, algumas arestas cortadas do conjunto A^- representam um mesmo currículo $u \in U$. Isso ocorre pois no grafo G_C cada currículo $u \in U$ é representado com um clique de tamanho $|C_u|$. Portanto, se uma disciplina $c \in C_u$ pertence a uma partição diferente das demais disciplinas de C_u, isso significa que todas as arestas incidentes ao vértice que representa a disciplina c e ao vértice que representa alguma outra disciplina de C_u, foram cortadas durante o particionamento. Esse fato pode ser observado no grafo G_C da instância *Toy*, apresentado na Figura 6, em que duas arestas são cortadas, mas ambas representam o currículo *Cur1*. Assim, como a biblioteca *METIS* tenta minimizar o número de arestas cortadas ao particionar o grafo G_C e várias arestas cortadas representam o mesmo currículo, pode-se afirmar que a quantidade de currículos desconsiderados também será minimizada, representando assim uma relaxação mais forte. É importante lembrar que quanto menos (mais) restrições são relaxadas, o problema fica mais (menos) parecido com o problema original, sendo considerada assim uma relaxação forte (fraca).

Além disso, como cada disciplina $c \in C$ pode pertencer a mais de um currículo $u \in U$, ao particionar o grafo G_U no método *U-partition*, uma disciplina c representada por uma ou mais arestas de A^- terá mais horários e salas disponíveis para alocar suas aulas, uma vez que o número de disciplinas que concorrem pelos mesmos horários e salas será menor. Em outras palavras, se todos os currículos $u \in U_c$ pertencerem a um subproblema diferente, isso significa que a disciplina c terá em cada subproblema menos disciplinas que concorrem pelos mesmos horários e salas, ou seja, mais horários e salas disponíveis representam menos chance das restrições **S1** (capacidade de sala) e **S2** (número mínimo de dias de aula) serem violadas. Em contrapartida, se todos os currículos $u \in U_c$ pertencerem ao mesmo subproblema, isso significará que todas as disciplinas que pertencem a um currículo $u \in U_c$ concorrem com a disciplina c pelos mesmos horários e salas, aumentando assim a chance de que ocorra violação das restrições anteriormente citadas. Por fim, cada aresta cortada no método *U-partition* representa uma ou mais disciplinas, e apesar da biblioteca *METIS* tentar minimizar o número de arestas cortadas, a quantidade de disciplinas que se encaixam na situação descrita acima pode ser grande, o que sugere que o método *U-partition* apresenta uma relaxação mais fraca.

Os resultados dos métodos *C-partition-null* e *C-partition-org*, que usam variáveis de cópia, foram em média melhores do que os resultados do método *C-partition*, que não usa tais variáveis. Para comparação, os métodos *C-partition-null* e *C-partition-org*

apresentaram, em média, resultados 6,46% e 11,02%, respectivamente, superiores ao obtido pelo método *C-partition*. Em relação ao número de instâncias, o método *C-partition-null* encontrou resultados superiores ao método *C-partition* em duas instâncias e o mesmo resultado nas outras 19 instâncias, enquanto que o método *C-partition-org* encontrou resultados superiores em cinco instâncias e o mesmo resultado nas outras 16 instâncias. Esse comportamento nos resultados dos métodos *C-partition-null* e *C-partition-org* já era esperado, pois conforme apresentado na Seção 4.4, eles usam variáveis de cópias para representar as disciplinas que serão consideradas em mais de um subproblema por pertecerem a algum currículo representado por alguma aresta em A^-. A utilização dessas variáveis de cópia representa uma relaxação mais forte do que a do método *C-partition*, pois cada subproblema é mais parecido com o problema original.

Por fim, pode-se destacar que, de forma geral, conforme a Tabela 4, o método *C-partition-org* apresenta resultados melhores ou iguais, que os outros três métodos propostos, em todas as instâncias (destaques em **negrito**). Além disso, em 16 das 21 instâncias os resultados obtidos pelo método *C-partition-org* foram iguais aos obtidos pelo método *C-partition-null*. Os resultados das outras cinco instâncias contribuem para que a eficácia do método *C-partition-org* seja, em média, 4,29% superior que os resultados do método *C-partition-null*. Dentre essas instâncias, a maior diferença percentual é de 15,66% na instância comp05, já que o método *C-partition-null* obteve 198 como valor de limitante inferior e o método *C-partition-org*, 229.

Uma diferença importante entre os métodos *C-partition-null* e *C-partition-org* (Seção 4.4) é o conjunto de coeficientes das variáveis de cópia, $\overline{\omega^i}_{p,\psi,c}$ e $\overline{q^i}_c$, na função objetivo de cada subproblema SP_i em que a disciplina c não pertence ($\Upsilon[c] \neq i$), mas é considerada. No método *C-partition-org* os coeficientes das variáveis de cópia $\overline{\omega^i}_{p,\psi,c}$ e $\overline{q^i}_c$ são iguai a W^{RC} e W^{MD}, respectivamente, enquanto no método *C-partition-null* é 0 (zero). Dessa maneira, o método *C-partition-org* possui uma relaxação mais forte que o método *C-partition-null*, pois cada subproblema é mais parecido com o problema original, uma vez que até os coeficientes das variáveis de decisão nas suas funções objetivo são iguais aos das respectivas variáveis na função objetivo do problema original.

A diferença nos conjuntos de coeficientes das variáveis de cópia dos métodos *C-partition-null* e *C-partition-org* faz com que os resultados deste último sejam melhores, já que nele a violação das restrições **S1** (capacidade de sala) e **S2** (número mínimo de dias de aula) é evitada para a disciplina c representada com variáveis de cópia no subproblema SP_i. Isso ocorre pois essa violação aumenta o valor da função objetivo de SP_i. No método *C-partition-null* uma violação dessas restrições, pela mesma disciplina c, não é evitada, pois não impacta o valor da função objetivo de SP_i.

Em relação ao tempo total gasto pelo CPLEX para resolver todos os k subproblemas, com k variando de 2 até δ, pode-se observar na Tabela 4 que o método *U-partition*, em

7 das 21 instâncias, encontrou o valor do limitante inferior em menos de 10 segundos, sendo que a instância comp12 foi a com execução mais demorada (28222 segundos) para encontrar o melhor valor do limitante inferior. O método *C-partition* gastou, em média, mais tempo para encontrar os limitantes inferiores que o método *U-partition*, sendo que na instância comp12 o valor foi obtido após 33987 segundos e o método gastou menos de 10 segundos em apenas 6 das 21 instâncias.

A diferença percentual da média de tempo gasto pelos métodos *U-partition* e *C-partition* foi de apenas 1,21%. Entretanto, os métodos *C-partition-null* e *C-partition-org* encontraram o melhor valor de limitante inferior gastando, respectivamente, em média, 44,05% e 88,16% mais tempo que o método *C-partition*. Em ambos os métodos (*C-partition-null* e *C-partition-org*) também foi possível encontrar o melhor valor de limitante inferior, em 6 intâncias, gastando menos de 10 segundos. A instância comp12 também foi a que os métodos *C-partition-null* e *C-partition-org* demoraram mais, 44573 e 54807 segundos, respectivamente, para encontrar o melhor valor do limitante inferior.

A Tabela 4 resume os resultados obtidos pelos quatro métodos propostos na Etapa 1. A segunda etapa de testes foi realizada apenas com as instâncias comp02, comp05 e comp12. Nessas instâncias, em alguns métodos, o melhor valor de limitante inferior, dentre as $\delta - 1$ execuções, foi obtido com $k = \gamma$ e $\gamma \neq \delta$. Nesses casos o melhor valor de limitante inferior foi obtido particionando o problema original em γ subproblemas, mas a execução do otimizador CPLEX foi interrompida por ter atingido o tempo limite de 9880 segundos (40τ) para algum subproblema.

Portanto, a Etapa 2 consistiu na execução dos métodos e instâncias nos quais $\gamma \neq \delta$, com $k = \gamma$ e sem limitar o tempo de execução do otimizador CPLEX. Os resultados dessa etapa são apresentados na Tabela 5, sendo que nela são representados com um *traço* (-) os casos das instâncias comp02, comp05 e comp12 em que os métodos não foram novamente executados na Etapa 2, já que na primeira etapa o melhor valor de limitante inferior foi obtido particionando o problema original em γ subproblemas e em nenhum deles a execução do CPLEX extrapolou o tempo limite de 9880 segundos (40τ), ou seja, os casos em que $\gamma = \delta$.

Tabela 5 – Limitantes inferiores obtidos pelos métodos propostos na Etapa 2.

Instâncias	*U-partition*		*C-partition*		*C-partition-null*		*C-partition-org*	
	lb	T(*seg.*)	*lb*	T(*seg.*)	*lb*	T(*seg.*)	*lb*	T(*seg.*)
comp02	-	-	22	30667	24	60048	24	58208
comp05	147	79507	-	-	-	-	230	20955
comp12	-	-	174	110189	195	106366	-	-

Analisando os valores apresentados na Tabela 5, observa-se que em todas as

execuções o valor do limitante inferior encontrado foi superior ao respectivo valor obtido na Etapa 1, sendo a maior diferença observada na instância comp02, com o método *C-partition-null*. Esse método obteve o valor 17, na Etapa 1, enquanto que na segunda etapa de testes o valor obtido foi 24, representando assim um aumento de 41,18%.

O valor médio dos limitantes inferiores, considerando os resultados da Etapa 2, é 44,62 e 49,19 para os métodos *U-partition* e *C-partition*, respectivamente. Esses valores representam um incremento de 0,86% e 2,48%, respectivamente, ao valor médio observado na primeira etapa de testes. Em relação aos métodos *C-partition-null* e *C-partition-org*, que usam variáveis de cópia, o valor médio dos limitantes inferiores, considerando os resultados da Etapa 2, são 2,70% e 0,63%, respectivamente, melhores do que o valor médio observado na primeira etapa. Portanto, o valor médio dos limitantes inferiores, considerando os resultados da segunda etapa de testes, é 52,48 e 53,62, respectivamente, para os métodos *C-partition-null* e *C-partition-org*.

Os registros de tempo na Tabela 5 representam apenas o tempo gasto na execução única dos métodos com $k = \gamma$, enquanto que os valores de tempo apresentados na Tabela 4 (Etapa 1), equivalem ao tempo gasto nas $\delta - 1$ execuções dos métodos. Diante disso, observa-se que na segunda etapa de testes o tempo necessário para obter o valor de limitante inferior com $k = \gamma$, e sem limitar o tempo de execução do CPLEX, foi maior do que a soma do tempo gasto nas $\delta - 1$ execuções da primeira etapa de testes. A única exceção é a instância comp05 com o método *C-partition-org*, que na primeira etapa de testes obteve o valor de limitante inferior 229 com $k = \gamma = 3$ e tempo total gasto nas três execuções ($\delta = 4$) de 35408 segundos. Já na segunda etapa de testes, com a execução única de $k = \gamma = 3$, o valor 230 de limitante inferior foi obtido após 20955 segundos.

Considerando os resultados das duas etapas de testes, é possível observar que o método *C-partition-org* obteve valores de limitantes inferiores melhores ou iguais que os demais métodos (*U-partition*, *C-partition* e *C-partition-null*) para as 21 instâncias, sendo assim, dentre os quatro métodos, o que teve melhor resultado médio. Por essa razão, os resultados do método *C-partition-org* foram escolhidos para compor a tabela de comparação com os resultados apresentados em outros trabalhos da literatura.

5.3.2 Análise comparativa com resultados da literatura

Nas Tabelas 6 e 7 são apresentados os resultados do método *C-partition-org* e de outros trabalhos da literatura. A Tabela 6 se restringe apenas as 14 instâncias iniciais (comp01 - comp14) da formulação CB-CTT do ITC-2007, pois os trabalhos de Burke et al. (2010), Burke et al. (2012) e Lach e Lübbecke (2012) apresentam apenas os resultados dessas instâncias. Nessa tabela é apresentado o melhor valor de limitante superior (ub) atualmente conhecido para cada uma das 14 instâncias. Além disso, ao lado direito de cada valor de limitante inferior, encontra-se o valor do *gap* de otimalidade ($\%gap$) entre ub

e o respectivo limitante inferior.

A Tabela 7 apresenta os resultados do método *C-partition-org* para todas as instâncias (comp01 - comp21) da formulação CB-CTT do ITC-2007. Nessa tabela também são apresentados os resultados obtidos nos trabalhos de Hao e Benlic (2011), Cacchiani et al. (2013), Achá e Nieuwenhuis (2014), Bagger, Desaulniers e Desrosiers (2019), Bagger et al. (2019) e Bagger, Sørensen e Stidsen (2019), que apresentam os valores dos limitantes inferiores obtidos para cada uma das 21 instâncias.

Ao final de ambas as tabelas é exibida uma legenda com as principais informações dos trabalhos usados para comparação. Nas Tabelas 6 e 7 também são apresentados entre parêntesis, para as instâncias comp02 e comp05, os valores de limitantes inferiores (lb) obtidos pelo método *C-partition-org* na Etapa 2 com os respectivos *gaps* de otimalidade. A média dos resultados e a média de $\%gap$, considerando os valores obtidos na segunda etapa de testes, também são apresentados entre parêntesis. Além disso, em ambas as tabelas, o melhor valor de limitante inferior encontrado para cada instância é destacado em **negrito**.

Com o intuito de facilitar a análise e comparação dos resultados apresentados nas Tabelas 6 e 7, a Tabela 8 resume as principais contribuições do método *C-partition-org* e dos demais trabalhos da literatura usados nas comparações. Na primeira coluna da Tabela 8 são exibidos, ordenados cronologicamente pela data em que foram publicados, os trabalhos usados na comparação e o método *C-partition-org*. Nas outras cinco colunas da tabela são apresentados, respectivamente, a quantidade de instâncias que a otimalidade foi provada (Opt), a quantidade de instâncias que a otimalidade foi provada pela primeira vez (Opt^{1st}), a quantidade de instâncias que houve uma melhoria no valor de limitante inferior encontrado em relação aos anteriormente obtidos por outros trabalhos (Imp), a média de limitantes inferiores (lb_m) e a média de $\%gap$ ($\%gap_m$).

Ainda em relação à Tabela 8 é importante frisar que os valores apresentados para os trabalhos de Burke et al. (2010), Burke et al. (2012) e Lach e Lübbecke (2012) consideram apenas as 14 instâncias iniciais, pois esses trabalhos apresentam apenas os resultados dessas instâncias. O trabalho de Burke et al. (2010), por ter sido o primeiro da literatura a apresentar valores de limitantes inferiores para a formulação CB-CTT, na quarta coluna (Imp) não é exibido nenhum valor, sendo representado assim com um *traço* (-). Já o trabalho de Hao e Benlic (2011), por ter sido o primeiro da literatura a apresentar valores de limitantes inferiores para todas as 21 instâncias da formulação CB-CTT, na quarta coluna (Imp) o valor exibido se refere apenas a comparação com as 14 instâncias iniciais.

Tabela 6 – Comparação de limitantes inferiores obtidos pelo método *C-partition-org* com outros métodos (14 instâncias).

Instâncias	ub[1]	BMPR10[2]						BMPR12[3]				LL12[4]		*C-partition-org*[5]	
		lb1	*%gap*	*lb2*	*%gap*	*lb3*	*%gap*	*lb1*	*%gap*	*lb2*	*%gap*	*lb*	*%gap*	*lb*	*%gap*
comp01	5	4	20,00	0	100,00	**5**	0,00	4	20,00	**5**	0,00	4	20,00	4	20,00
comp02	24	1	95,83	0	100,00	1	95,83	0	100,00	6	75,00	11	54,17	**18** (**24**)	25,00 (0,00)
comp03	64	25	60,94	25	60,94	33	48,44	0	100,00	43	32,81	25	60,94	**61**	4,69
comp04	35	8	77,14	**35**	0,00	**35**	0,00	0	100,00	2	94,29	28	20,00	**35**	0,00
comp05	284	111	60,92	119	58,10	114	59,86	95	66,55	183	35,56	108	61,97	**229** (**230**)	19,37 (19,01)
comp06	27	12	55,56	13	51,85	16	40,74	0	100,00	6	77,78	10	62,96	**27**	0,00
comp07	6	0	100,00	**6**	0,00	**6**	0,00	0	100,00	0	100,00	**6**	0,00	**6**	0,00
comp08	37	11	70,27	**37**	0,00	**37**	0,00	0	100,00	2	94,59	**37**	0,00	**37**	0,00
comp09	96	21	78,13	68	29,17	66	31,25	0	100,00	0	100,00	46	52,08	**96**	0,00
comp10	4	2	50,00	3	25,00	**4**	0,00	0	100,00	0	100,00	**4**	0,00	**4**	0,00
comp11	0	**0**	0,00	**0**	0,00	**0**	0,00	**0**	0,00	**0**	0,00	**0**	0,00	**0**	0,00
comp12	294	39	86,73	101	65,65	95	67,69	0	100,00	5	98,30	53	81,97	**182**	38,10
comp13	59	14	76,27	52	11,86	54	8,47	3	94,92	0	100,00	41	30,51	**59**	0,00
comp14	51	40	21,57	41	19,61	42	17,65	0	100,00	0	100,00	46	9,80	**51**	0,00
Média		20,57	60,95	35,71	37,30	36,29	26,42	7,29	84,39	18,00	72,02	29,93	32,46	**57,79** (**58,29**)	7,65 (5,84)

[*] Melhores valores de limitantes inferiores (*lb*) destacados em **negrito**
[1] Melhores valores de soluções viáveis (limitantes superiores) - Kiefer, Hartl e Schnell (2017)
[2] BMPR10 - Burke et al. (2010) - Monolítico (*lb1*), *Surface1* (*lb2*) e *Surface2* (*lb3*) - Tempo Limite de 31200 segundos (*lb1* e *lb3*) e 780 segundos (*lb2*).
[3] BMPR12 - Burke et al. (2012) - *Branch-and-cut*: Monolítico Relaxado + dois tipos (*Tipo 1* e *Tipo 2*) de corte (*lb1*) - *Surface1* + um dos três conjuntos de restrições propostos no corte *Tipo 2* (*lb2*) - Tempo Limite de 7200 segundos (*lb1*) e 1800 segundos (*lb2*).
[4] LL12 - Lach e Lübbecke (2012) - Modelo matemático da primeira fase - Tempo Limite de 13000 segundos.
[5] *C-partition-org* - Tempo Limite de 9880 segundos por subproblema.

Tabela 7 – Comparação de limitantes inferiores obtidos pelo método *C-partition-org* com outros métodos (21 instâncias).

Instâncias	ub[1]	HB11[2]		CCRT13[3]		AN14[4]		BDD16[5]		BKSS17[6]		BSS18[7]		*C-partition-org*[8]	
		lb	*%gap*	*lb*	*%gap*	*lb*	*%gap*	*lb*	*%gap*	*lb*	*%gap*	*lb*	*%gap*	*lb*	*%gap*
comp01	5	4	20,00	**5**	0,00	0	100,00	0	100,00	**5**	0,00	0	100,00	4	20,00
comp02	24	12	50,00	16	33,33	16	33,33	**24**	0,00	8	66,67	20	16,67	18 (**24**)	25,00 (0,00)
comp03	64	38	40,63	52	18,75	28	56,25	54	15,63	38	40,63	58	9,38	**61**	4,69
comp04	35	**35**	0,00	**35**	0,00	**35**	0,00	**35**	0,00	**35**	0,00	**35**	0,00	**35**	0,00
comp05	284	183	35,56	166	41,55	48	83,10	210	26,06	186	34,51	**247**	13,03	229 (230)	19,37 (19,01)
comp06	27	22	18,52	11	59,26	**27**	0,00	26	3,70	16	40,74	23	14,81	**27**	0,00
comp07	6	**6**	0,00	**6**	0,00	**6**	0,00	**6**	0,00	**6**	0,00	**6**	0,00	**6**	0,00
comp08	37	**37**	0,00	**37**	0,00	**37**	0,00	**37**	0,00	**37**	0,00	**37**	0,00	**37**	0,00
comp09	96	72	25,00	92	4,17	35	63,54	**96**	0,00	74	22,92	92	4,17	**96**	0,00
comp10	4	**4**	0,00	2	50,00	**4**	0,00	**4**	0,00	**4**	0,00	**4**	0,00	**4**	0,00
comp11	0	**0**	0,00	**0**	0,00	**0**	0,00	**0**	0,00	**0**	0,00	**0**	0,00	**0**	0,00
comp12	294	109	62,93	100	65,99	99	66,33	175	40,48	142	51,70	**248**	15,65	182	38,10
comp13	59	**59**	0,00	57	3,39	**59**	0,00	**59**	0,00	**59**	0,00	**59**	0,00	**59**	0,00
comp14	51	**51**	0,00	48	5,88	**51**	0,00	**51**	0,00	44	13,73	49	3,92	**51**	0,00
comp15	62	38	38,71	52	16,13	28	54,84	54	12,90	38	38,71	58	6,45	**61**	1,61
comp16	18	16	11,11	13	27,78	**18**	0,00	**18**	0,00	13	27,78	17	5,56	**18**	0,00
comp17	56	48	14,29	48	14,29	**56**	0,00	53	5,36	44	21,43	**56**	0,00	**56**	0,00
comp18	61	24	60,66	**52**	14,75	27	55,74	**52**	14,75	36	40,98	**52**	14,75	40	34,43
comp19	57	56	1,75	48	15,79	46	19,30	**57**	0,00	56	1,75	51	10,53	**57**	0,00
comp20	4	2	50,00	**4**	0,00	**4**	0,00	**4**	0,00	0	100,00	3	25,00	**4**	0,00
comp21	74	61	17,57	68	8,11	42	43,24	**74**	0,00	57	22,97	71	4,05	**74**	0,00
Média		41,76	21,27	43,43	18,06	31,71	27,41	51,86	10,42	42,76	24,98	**56,48**	11,62	53,29 (53,62)	6,82 (5,61)

[*] Melhores valores de limitantes inferiores (*lb*) destacados em **negrito**
[1] Melhores valores de soluções viáveis (limitantes superiores) - Kiefer, Hartl e Schnell (2017)
[2] HB11 - Hao e Benlic (2011) - Abordagem *dividir para conquistar* - Tempo Limite de 25200 segundos por subproblema.
[3] CCRT13 - Cacchiani et al. (2013) - *Geração de Colunas* - Tempo Limite de 22800 segundos.
[4] AN14 - Achá e Nieuwenhuis (2014) - Codificações SAT - Tempo Limite de 10^5 segundos.
[5] BDD16 - Bagger, Desaulniers e Desrosiers (2019) - Relaxações no modelo matemático de padrões - Tempo Limite de 20800 segundos.
[6] BKSS17 - Bagger et al. (2019) - Conexão de dois modelos independentes através de técnicas de formulação de fluxo - Tempo Limite de 8320 segundos.
[7] BSS18 - Bagger, Sørensen e Stidsen (2019) - *Geração de Colunas* - Tempo Ilimitado.
[8] *C-partition-org* - Tempo Limite de 9880 segundos por subproblema.

Tabela 8 – Resumo das principais contribuições do método *C-partition-org* e demais trabalhos da literatura.

	Opt	Opt^{1st}	Imp	lb_m	$\%gap_m$
Burke et al. (2010)	6	6	-	36,29	26,42
Burke et al. (2012)	2	0	3	18,00	72,02
Lach e Lübbecke (2012)	4	0	2	29,93	32,46
Hao e Benlic (2011)	7	2	6	41,76	21,27
Cacchiani et al. (2013)	6	1	7	43,43	18,06
Achá e Nieuwenhuis (2014)	11	3	3	31,71	27,41
Bagger, Desaulniers e Desrosiers (2019)	13	4	8	51,86	10,42
Bagger et al. (2019)	7	0	0	42,76	24,98
Bagger, Sørensen e Stidsen (2019)	7	0	4	56,48	11,62
C-partition-org	15	0	2	53,29	6,82

Nas análises comparativas (Tabelas 6 e 7) realizadas com o método *C-partition-org*, os resultados obtidos na Etapa 1 de testes são considerados. É importante recordar que, nessa etapa, o tempo de execução do CPLEX foi limitado em 40τ para proporcionar uma comparação justa com os resultados obtidos por Burke et al. (2010), Lach e Lübbecke (2012), Cacchiani et al. (2013) e Bagger et al. (2019), cujos tempos de execução dos seus métodos foram limitados usando essa mesma medida.

De acordo com a Tabela 6, o método *C-partition-org* encontrou, em média, melhores valores de limitantes inferiores do que os trabalhos que apresentaram valores de limitantes inferiores apenas para as 14 primeiras instâncias. O valor médio foi 57,79, o que equivale a uma melhoria de 59,23% em relação ao melhor valor médio (36,29) obtido dentre os métodos que foram executados apenas para esse conjunto de instâncias apresentados na Tabela 6. Esse resultado é justificado pelo fato do método *C-partition-org* considerar nos subproblemas as restrições que modelam **S3** (aulas isoladas), enquanto que nos demais trabalhos na literatura, cujos resultados são apresentados na Tabela 6, por serem de difícil resolução, essas restrições são completamente desconsideradas.

Na Tabela 7, que apresenta os resultados para todas as instâncias, a média dos valores obtidos pelo método *C-partition-org* foi 22,70% melhor que a média dos valores obtidos pelo método *Geração de Colunas* apresentado por Cacchiani et al. (2013). Neste caso, o método *C-partition-org* encontrou melhores valores de limitantes inferiores, quando comparado aos resultados de Cacchiani et al. (2013), em 14 instâncias, sendo que em outras 5 instâncias os valores obtidos por ambos os métodos, *C-partition-org* e *Geração de Colunas*, foram equivalentes.

Conforme descrito na Seção 3.1.6, o valor médio dos limitantes inferiores encontrados por Achá e Nieuwenhuis (2014) foi pior do que o apresentado por Cacchiani et al. (2013). No entanto, as codificações SAT propostas por Achá e Nieuwenhuis (2014) provaram a

otimalidade de 11 instâncias. Nessas instâncias o método *C-partition-org* também obteve o mesmo resultado, ou seja, provou a otimalidade. Já em outras três instâncias (comp09, comp19 e comp21), que as codificações SAT não provaram a otimalidade, o método *C-partition-org* obteve valor de limitante inferior igual ao valor do melhor limitante superior (*ub*) atualmente conhecido, provando assim a otimalidade dessas instâncias. As codificações SAT, propostas por Achá e Nieuwenhuis (2014), e o método *C-partition-org*, na primeira etapa de testes, não provaram a otimalidade da instância comp02. No entanto, o valor de limitante inferior obtido pelo método *C-partition-org*, para a instância comp02, na execução da segunda etapa de testes, é igual ao valor do melhor limitante superior (*ub*) atualmente conhecido para essa instância, provando assim também a sua otimalidade. Dessa maneira, o método *C-partition-org* provou a otimalidade em 15 das 21 instâncias.

É importante lembrar que o método *C-partition-org*, da mesma forma que o método proposto por Hao e Benlic (2011), também usa a abordagem *dividir para conquistar*, sendo que a principal diferença entre esses métodos está no fato que o *C-partition-org* usa a API da biblioteca *METIS* para particionar o grafo G_C, enquanto que o método proposto por Hao e Benlic (2011) usa a heurística *Busca Tabu Iterativa* para particionar o mesmo grafo. É importante destacar que tanto a biblioteca *METIS* quanto a heurística *Busca Tabu Iterativa* possuem o mesmo objetivo, ou seja, encontrar um particionamento do grafo G_C que minimize o número de arestas cortadas.

Além dos métodos usarem diferentes estratégias de particionamento, e apesar delas possuirem o mesmo objetivo, os métodos também se diferem pelo ambiente computacional em que os experimentos computacionais foram realizados e pelo otimizador usado, já que nesse trabalho foi usado o CPLEX, enquanto que no trabalho de Hao e Benlic (2011) foi usado o otimizador COIN-OR. Por fim, o método *C-partition-org* usa variáveis de cópia que garantem que todo currículo $u \in U$ seja considerado em um subproblema, enquanto que no método proposto por Hao e Benlic (2011) alguns currículos podem não ser considerados, pois tiveram uma ou mais arestas do seu clique cortadas durante o particionamento. Todas essas diferenças justificam o fato dos resultados desses métodos, que são apresentados na Tabela 7, serem diferentes.

Assim, ao comparar os resultados do método *C-partition-org* com os resultados reportados por Hao e Benlic (2011), conforme são apresentados na Tabela 7, verifica-se que o método *C-partition-org* obteve um resultado médio 27,61% melhor que o método de Hao e Benlic (2011). Em uma análise quantitativa, o valor de limitante inferior obtido pelo *C-partition-org* em 13 das 21 instâncias foi melhor do que o apresentado por Hao e Benlic (2011), sendo que houve um empate nos resultados das outras oito instâncias.

Das três abordagens apresentadas por Bagger, Desaulniers e Desrosiers (2019), Bagger et al. (2019) e Bagger, Sørensen e Stidsen (2019), em apenas uma delas o valor médio obtido pelo método *C-partition-org* foi inferior, sendo que o método *Geração de*

Colunas apresentado por Bagger, Sørensen e Stidsen (2019) é o que apresenta atualmente o melhor valor médio na literatura. Apesar disso, o método *Geração de Colunas* provou a otimalidade em apenas 7 das 21 instâncias. Com isso, observando os resultados obtidos pelo método *Geração de Colunas* de Bagger, Sørensen e Stidsen (2019), nota-se que nas instâncias comp05, comp12 e comp18 os valores de limitantes inferiores foram os melhores obtidos (comp05 e comp12) até o momento ou igual (comp18) ao valor obtido anteriormente em outros trabalhos na literatura. Nessas instâncias, os limitantes inferiores obtidos pelo método *Geração de Colunas* foram aproximadamente 7,86%, 36,26% e 30,00% melhores do que valores obtidos pelo método *C-partition-org*, respectivamente, para as instâncias, comp05, comp12 e comp18. Dessa forma, os valores obtidos por Bagger, Sørensen e Stidsen (2019) para essas instâncias contribuíram fortemente para que o valor médio do método *Geração de Colunas* fosse superior ao valor médio obtido pelo método *C-partition-org*. Caso os valores obtidos nessas três instâncias sejam desconsiderados, em ambos os métodos, o valor médio do método *C-partition-org* seria aproximadamente 4,53% superior ao valor médio do método *Geração de Colunas*.

Conforme descrito no Capítulo 2 e na Seção 5.1, o conjunto de instâncias da formulação CB-CTT do ITC-2007 foi definido por Gaspero, McCollum e Schaerf (2007), que indicaram também uma série de suas características. Ao analisar essas características, foi possível perceber que as instâncias comp05, comp12 e comp18 são as únicas em que a quantidade de currículos é maior que a quantidade de disciplinas ($|U| > |C|$). Essa informação é importante, pois indica que no grafo G_C dessas instâncias, o peso de uma grande parte das arestas tende a ser maior que uma unidade, além de ser um grafo denso, com muitas arestas. Assim, ao particionar esse grafo com a API da biblioteca *METIS*, mesmo que a soma do peso das arestas cortadas seja minimizado, a quantidade de arestas cortadas é grande, e cada uma delas pode representar vários currículos.

Dessa forma, após particionar o grafo G_C das instâncias comp05, comp12 e comp18, uma grande quantidade de disciplinas são representadas com variáveis de cópia, o que nesses casos acaba enfraquecendo a relaxação do método *C-partition-org*, e que por sua vez justifica o desempenho inferior nessas instâncias.

Em relação à resolução do modelo matemático de padrões de alocação, proposto por Bagger, Desaulniers e Desrosiers (2019), é importante destacar que o método *C-partition-org* superou o seu valor médio de limitantes inferiores obtido, mas a diferença percentual foi de apenas 2,76%. Ainda, Bagger, Desaulniers e Desrosiers (2019) provaram a otimalidade em 13 das 21 instâncias, sendo o trabalho com maior número de instâncias com ótimos provados disponível na literatura. Vale ressaltar que o método *C-partition-org* provou a otimalidade, considerando os resultados obtidos nas duas etapas de testes, em 15 instâncias.

Apesar do valor médio de limitantes inferiores encontrados pelo método *C-partition-*

org ter sido pior do que o obtido no trabalho de Bagger, Sørensen e Stidsen (2019), é importante destacar que o método *C-partition-org* obteve valor médio de *%gap* melhor que todos os demais trabalhos da literatura. Esse valor médio de *%gap* (6,82) foi 52,79% melhor do que o apresentado por Bagger, Desaulniers e Desrosiers (2019) (10,42), que é o melhor publicado até o momento na literatura. Esse resultado permite afirmar que o método *C-partition-org* obteve, em média, valores de limitantes inferiores 3,60% mais próximos dos respectivos valores de limitantes superiores.

Por fim, é necessário observar que a resolução de modelo matemático proposto por Bagger et al. (2019), que consiste na junção de dois modelos independentes através de técnicas de formulação de fluxo, obteve resultado médio inferior ao método *C-partition-org* e provou a otimalidade em apenas 7 das 21 instâncias. Esses resultados precisam ser observados, já que o trabalho de Bagger et al. (2019) teve como base o trabalho de Lach e Lübbecke (2012) e o método *C-partition-org* foi motivado pela abordagem *dividir para conquistar* de Hao e Benlic (2011), que por sua vez também teve como referência o modelo de duas fases proposto por Lach e Lübbecke (2012).

De forma geral, ao comparar os resultados apresentados nas Tabelas 6 e 7, pode-se destacar que o método *C-partition-org* atualizou o valor do melhor limitante inferior, atualmente conhecido, em duas instâncias (comp03 e comp15). Além disso, o método *C-partition-org* provou a otimalidade em outras 15 instâncias, mas obtendo como resultado valores iguais aos que já haviam sido obtidos por outros trabalhos na literatura.

Apesar dos resultados satisfatórios apresentados pelo método *C-partition-org*, em cinco instâncias, a otimalidade ainda não foi provada, o que serve de motivação para prosseguir com o estudo apresentado neste trabalho, e assim propor outros métodos para encontrar limitantes inferiores para a formulação CB-CTT do ITC-2007. No Capítulo 6 são abordadas algumas possibilidades para dar continuidade a este trabalho.

Após a análise dos resultados obtidos pelo método *C-partition-org* em comparação com outros trabalhos da literatura, os valores de limitantes inferiores apresentados na Tabela 7 foram submetidos a uma análise de significância estatística. Essa análise não tem como objetivo mostrar que um método é melhor ou pior que outro, já que nesse contexto a palavra *significância* não tem o sentido de *importância*, mas avaliar a validade (ou não) de uma afirmação feita sobre uma determinada característica em um conjunto de dados, ou seja, se há (ou não) uma diferença significativa entre o conjunto de dados observados e esperados que nos permita afirmar se tal característica existe ou não (PINHEIRO et al., 2009).

Dessa forma, antes de iniciar a análise de significância, é preciso definir a afirmação que se deseja validar com a análise. Essa afirmação é denominada de *Hipótese Nula* (H_0). Ao definir H_0 também é necessário definir uma outra hipótese, denominada *Hipótese Alternativa* (H_a), que é mutuamente excludente à hipótese H_0. Portanto, o objetivo da

análise é aceitar H_0 ou recusar H_0 (em favor de H_a) (DEVORE, 2006).

Em seguida, mas ainda antes da realização da análise, é definido o *Nível de Significância* (π). O valor π representa a probabilidade de rejeitar H_0 e geralmente é definida em 5% ou 1% (PINHEIRO et al., 2009). Assim, durante a análise de significância estatística é calculada uma probabilidade, também denominada $p - valor$, que representa a probabilidade de um valor observado em um experimento se distancie significativamente do valor esperado, supondo que H_0 seja verdadeira. Em outras palavras, se $p - valor \leq \pi$ então H_0 é rejeitada (em favor de H_a) e, caso contrário ($p - valor > \pi$), H_0 é aceita.

Na análise de significância estatística realizada nesta tese a hipótese H_0 é descrita como: *Não há diferença significativa de resultados entre os métodos* apresentados na comparação da Tabela 7. Por sua vez, a hipótese H_a é descrita como: *Há pelo menos um método cujos resultados se distanciam significativamente dos demais*. O *Nível de Significância* estabelecido foi de 5% ($\pi = 0,05$). A escolha desse valor foi motivada pelo fato desse ser o maior valor dentre os normalmente usados (5% e 1%), o que representa uma maior probabilidade de H_0 ser rejeitada (em favor de H_a), ou seja, aumenta a chance dessa análise encontrar um ou mais métodos cujos resultados se distanciam significativamente dos demais e se aproximam dos melhores valores de limitantes superiores (ub) atualmente conhecidos.

Após a definição de H_0, H_a e π, os valores de limitantes inferiores, apresentados na Tabela 7, e os melhores valores de limitantes superiores (ub) atualmente conhecidos, foram submetidos ao *Teste de Friedman* (FRIEDMAN, 1940) para o cálculo de $p - valor$. Esse cálculo foi realizado usando a versão 4.0.2 da linguagem e ambiente *R* para computações estatísticas (R FOUNDATION FOR STATISTICAL COMPUTING, 2018).

Na realização desse *Teste de Friedman* o $p - valor$ obtido foi igual a $2,927 \times 10^{-6}$. Como $p - valor \leq \pi$, então H_0 é rejeitada, ou seja, existe pelo menos um método, dentre aqueles cujos valores de limitantes inferiores são apresentados na Tabela 7, que se distancia dos demais métodos e dos valores de limitantes superiores (ub). Dessa maneira, os valores foram submetidos a um novo teste. Esse novo teste é chamado *Teste de Nemenyi* (NEMENYI, 1963) e consiste na aplicação do *Teste de Friedman* entre pares dos métodos cujos valores de limitantes inferiores são apresentados na Tabela 7, e também desses métodos com os valores de limitantes superiores (ub).

Portanto, o *Teste de Nemenyi* retorna uma tabela com os valores de $p - valor$ obtidos ao executar o *Teste de Friedman* entre cada par de métodos e entre cada método com os valores *ub*. Assim, esse teste permite que sejam observados grupos de métodos que não possuem diferença significativa de resultados. Na Tabela 9 são apresentados os resultados dessa execução do *Teste de Nemenyi*.

Ao analisar os valores apresentados na segunda coluna da Tabela 9, se observa que

Tabela 9 – Resultados do primeiro *Teste de Nemenyi*.

	ub[1]	HB11[2]	CCRT13[3]	AN14[4]	BDD16[5]	BKSS17[6]	BSS18[7]
HB11[2]	0,001	-					
CCRT13[3]	0,001	1,000	-				
AN14[4]	0,002	1,000	1,000	-			
BDD16[5]	1,000	0,129	0,129	0,145	-		
BKSS17[6]	0,001	1,000	1,000	1,000	0,089	-	
BSS18[7]	0,357	0,610	0,610	0,665	1,000	0,489	-
C-partition-org	1,000	0,034	0,034	0,041	1,000	0,020	1,000

[1] Melhores valores de soluções viáveis (limitantes superiores) - Kiefer, Hartl e Schnell (2017).
[2] HB11 - Hao e Benlic (2011) - Abordagem *dividir para conquistar*.
[3] CCRT13 - Cacchiani et al. (2013) - *Geração de Colunas*.
[4] AN14 - Achá e Nieuwenhuis (2014) - Codificações SAT.
[5] BDD16 - Bagger, Desaulniers e Desrosiers (2019) - Relaxações no modelo matemático de padrões.
[6] BKSS17 - Bagger et al. (2019) - Conexão de dois modelos independentes através de técnicas de formulação de fluxo.
[7] BSS18 - Bagger, Sørensen e Stidsen (2019) - *Geração de Colunas*.

em três casos (destaques em *cinza*) os valores obtidos de $p - valor$ foram superiores a π no *Teste de Friedman* entre os métodos e os melhores valores de limitantes superiores atualmente conhecidos (*ub*). Isso indica que entre os valores de *ub* e os resultados dos métodos de Bagger, Desaulniers e Desrosiers (2019), de Bagger, Sørensen e Stidsen (2019) e do método *C-partition-org* não há diferença significativa. Além disso, ainda é possível observar na Tabela 9 que os métodos de Bagger, Desaulniers e Desrosiers (2019), de Bagger, Sørensen e Stidsen (2019) e o *C-partition-org*, entre si, também tiveram valores de $p - valor$ superiores a π (demais destaques em *cinza*). Portanto, pode-se afirmar que os valores de *ub* e os resultados de Bagger, Desaulniers e Desrosiers (2019), Bagger, Sørensen e Stidsen (2019) e do *C-partition-org* formam um subconjunto que não se diferenciam significativamente entre si. Por isso, o *Teste de Friedman* foi novamente executado, mas dessa vez apenas com os resultados desses métodos e com os valores de *ub*.

Na segunda execução do *Teste de Friedman* o $p-valor$ obtido foi igual a $2,85 \times 10^{-4}$. Da mesma forma que no primeiro *Teste de Friedman* o $p - valor$ é menor que π, então H_0 é novamente rejeitada, ou seja, ainda existe pelo menos um método, dentre os métodos de Bagger, Desaulniers e Desrosiers (2019), de Bagger, Sørensen e Stidsen (2019) e do *C-partition-org*, cujos resultados se distanciam significativamente dos demais e dos valores de limitantes superiores (*ub*). Com isso, esses valores foram submetidos ao *Teste de Nemenyi*, ou seja, o *Teste de Nemenyi* foi novamente aplicado, mas apenas com os valores obtidos por Bagger, Desaulniers e Desrosiers (2019), Bagger, Sørensen e Stidsen (2019), *C-partition-org* e com os valores de limitantes superiores (*ub*). A Tabela 10 apresenta os resultados dessa segunda execução do *Teste de Nemenyi*.

Os destaques em *cinza* na Tabela 10 indicam que os resultados do método de Bagger, Desaulniers e Desrosiers (2019), juntamente com os resultados do método *C-partition-org* e

Tabela 10 – Resultados do segundo *Teste de Nemenyi*.

	ub[1]	BDD16[2]	BSS18[3]
BDD16[2]	0,091	-	
BSS18[3]	0,010	0,865	-
C-partition-org	0,441	0,837	0,371

[1] Melhores valores de soluções viáveis (limitantes superiores) - Kiefer, Hartl e Schnell (2017).
[2] BDD16 - Bagger, Desaulniers e Desrosiers (2019) - Relaxações no modelo matemático de padrões.
[3] BSS18 - Bagger, Sørensen e Stidsen (2019) - *Geração de Colunas*.

os valores de limitantes superiores (*ub*) não apresentam diferença significativa entre si, pois apresentam valores para $p - valor$ superiores a π. O método de Bagger, Sørensen e Stidsen (2019) apesar de apresentar valores para $p - valor$ superiores a π quando executado com os resultados do método *C-partition-org* e do método de Bagger, Desaulniers e Desrosiers (2019), não obteve o mesmo desempenho com os melhores valores de limitantes superiores atualmente conhecidos (*ub*).

A partir disso, o *Teste de Friedman* foi executado mais uma vez, mas agora desconsiderando os resultados de Bagger, Sørensen e Stidsen (2019), ou seja, apenas com os resultados de Bagger, Desaulniers e Desrosiers (2019), do método *C-partition-org* e os valores de *ub*. Nessa terceira execução do *Teste de Friedman* o $p - valor$ obtido foi igual a $0,094$. Dessa forma, H_0 é aceita, pois $p - valor > \pi$, o que indica que não há diferença significativa entre os resultados do método proposto por Bagger, Desaulniers e Desrosiers (2019) e do método *C-partition-org*, além de não existir diferença significativa deles com os melhores valores de limitantes superiores atualmente conhecidos (*ub*).

Ao final dessa análise, destaca-se que as relaxações no modelo matemático de padrões proposto por Bagger, Desaulniers e Desrosiers (2019) provou a otimalidade em 13 das 21 instâncias do PTHU da formulação CB-CTT, enquanto que o *C-partition-org* provou a otimalidade em 15 dessas instâncias. Além disso, o método de Bagger, Desaulniers e Desrosiers (2019) e o *C-partition-org* são os métodos que atualmente apresentam o melhor valor médio de *%gap*. Esses resultados evidenciam o que foi observado ao final da análise de significância estatística.

Após a apresentação dos resultados e da realização da análise de significância estatística, também foi possível realizar uma análise comparativa em relação ao tempo limite de execução. Ao final da Tabela 7 são apresentadas algumas informações sobre os experimentos realizados por Hao e Benlic (2011), Cacchiani et al. (2013), Achá e Nieuwenhuis (2014), Bagger, Desaulniers e Desrosiers (2019), Bagger et al. (2019) e Bagger, Sørensen e Stidsen (2019). Dentre essas informações também é apresentado o tempo limite de execução de cada método. Na terceira coluna da Tabela 11 esses valores são

apresentados novamente para facilitar a análise, sendo que na segunda coluna são exibidos os critérios usados na definição desses valores. Os trabalhos de Hao e Benlic (2011) e Achá e Nieuwenhuis (2014) não justificaram a definição dos limites de tempo dos seus métodos e, por isso, são representados na segunda coluna da Tabela 11 com um *traço* (-). No caso dos métodos *C-partition-org* e *Geração de Colunas* (BAGGER; SøRENSEN; STIDSEN, 2019), os valores apresentados se referem ao tempo gasto, pelos respectivos métodos, na instância comp12, já que foi nessa instância que ambos os métodos precisaram de mais tempo para obter o valor de limitante inferior. Os demais trabalhos definiram o tempo limite dos seus métodos usando o valor τ informado pelo programa *benchmark*, disponibilizado pela organização do ITC-2007, nos respectivos computadores usados para os experimentos computacionais.

Tabela 11 – Análise comparativa de tempo do método *C-partition-org* com outros trabalhos da literatura.

	Critério	T(*seg.*)
Hao e Benlic (2011)	-	25200
Cacchiani et al. (2013)	40τ ($\tau = 570$)	22800
Achá e Nieuwenhuis (2014)	-	100000
Bagger, Desaulniers e Desrosiers (2019)	100τ ($\tau = 208$)	20800
Bagger et al. (2019)	40τ ($\tau = 208$)	8320
Bagger, Sørensen e Stidsen (2019)	comp12	67399
C-partition-org	comp12	54807

É importante frisar que essa análise comparativa em relação ao tempo limite de execução não é perfeitamente justa, pois os tempos limites de execução definidos nos trabalhos da literatura, e usados nessa comparação, se referem à apenas uma execução dos respectivos métodos por instância, enquanto que o tempo considerado no método *C-partition-org* para a instância comp12, apresentado na Tabela 11, se refere a cinco execuções ($k = 2, \ldots, \delta$, sendo $\delta = 6$). Além disso, os computadores usados nos experimentos computacionais desses trabalhos se diferem entre si, e isso também contribui para que a comparação não seja justa. Por fim, alguns dos trabalhos da literatura, citados nessa comparação, usam versões do otimizador CPLEX diferentes da que foi usada neste trabalho, enquanto que outros trabalhos usam até mesmo outros otimizadores.

Dessa maneira, todas essas diferenças precisam ser consideradas ao analisar e comparar os tempos de execução apresentados na Tabela 11 e, por isso, a análise comparativa em relação ao tempo limite de execução, apresentada neste trabalho, tem o intuito apenas de destacar a eficiência do método *C-partition-org*.

Na Tabela 11 observa-se ainda que a abordagem *dividir para conquistar* de Hao e Benlic (2011), usada como motivação para esse trabalho, limitou o tempo de execução

do seu método em 25200 segundos por subproblema. Nesse valor não é considerado o tempo gasto para particionar o grafo G_C pela heurística *Busca Tabu Iterativa*, que foi de no máximo 600 segundos. Dessa forma, considerando a instância comp12, a qual o método *C-partition-org* demorou mais tempo (54807 segundos) para particionar e resolver todos os subproblemas nas cinco execuções, pode-se afirmar que o método *C-partition-org* encontrou valores de limitantes inferiores precisando de, no máximo, 117,49% mais tempo que a abordagem de Hao e Benlic (2011). Apesar disso, o método *C-partition-org* obteve os valores de limitantes inferiores em 19 das 21 instâncias, gastando menos tempo que o limite estabelecido por Hao e Benlic (2011).

De maneira semelhante, ao comparar o tempo limite de execução da *Geração de Colunas* proposta por Cacchiani et al. (2013) e da resolução do modelo de padrões de alocação proposto por Bagger, Desaulniers e Desrosiers (2019), com o tempo gasto pelo método *C-partition-org* para encontrar o valor de limitante inferior na instância comp12 (54807 segundos), pode-se afirmar que o método *C-partition-org* encontrou os valores de limitantes inferiores precisando, respectivamente, de 140,38% e 163,50% mais tempo. Entretanto, o método *C-partition-org* também encontrou em 19 instâncias, da mesma forma que na comparação com o trabalho de Hao e Benlic (2011), o valor de limitante inferior em menos tempo que o definido como limite nos trabalhos de Cacchiani et al. (2013) e Bagger, Desaulniers e Desrosiers (2019).

As codificações SAT apresentadas por Achá e Nieuwenhuis (2014) possuem tempo limite de execução definido em 10^5 segundos. Ao comparar o tempo gasto pelo método *C-partition-org* para encontrar o valor de limitante inferior na instância comp12 (54807 segundos) com o valor definido por Achá e Nieuwenhuis (2014), é possível afirmar que o método *C-partition-org* encontrou os valores de limitantes inferiores, no mínimo, 45,19% mais rápido que a abordagem de Achá e Nieuwenhuis (2014).

Na resolução do modelo unificado com técnicas de formulação de fluxo, conforme proposto por Bagger et al. (2019), o método *C-partition-org* encontrou, em 18 das 21 instâncias, o valor de limitante inferior em menos tempo que o limite estabelecido naquele trabalho, que foi de 8320 segundos. Para finalizar essa análise comparativa em relação ao tempo limite de execução, destaca-se que foi também na instância comp12 que a *Geração de Colunas* proposta por Bagger, Sørensen e Stidsen (2019) demorou mais tempo (67399 segundos) para encontrar o valor de limitante inferior. A partir disso, ao comparar esse valor, na instância comp12, com o tempo gasto pelo método *C-partition-org*, na mesma instância, é possivel afirmar que o método *C-partition-org* foi 18,68% mais rápido.

Os trabalhos de Cacchiani et al. (2013), Bagger, Desaulniers e Desrosiers (2019) e Bagger, Sørensen e Stidsen (2019) são os trabalhos da literatura que atualmente apresentam, em média, os melhores valores de limitantes inferiores para as instâncias da formulação CB-CTT do ITC-2007.

Portanto, o método *C-partition-org* foi eficaz em encontrar bons valores de limitantes inferiores, sendo na maioria das instâncias, iguais ou melhores que os valores atualmente conhecidos na literatura; e eficiente, por ter encontrado esses resultados mais rápido, ou em tempo similar, quando comparado com o tempo limite de execução definido por outros métodos.

6 Conclusões e trabalhos futuros

Neste trabalho foram propostos quatro métodos, denominados *U-partition*, *C-partition*, *C-partition-null* e *C-partition-org*, para encontrar limitantes inferiores para o PTHU da formulação CB-CTT, proposta no ITC-2007, tendo como motivação a abordagem *dividir para conquistar* de Hao e Benlic (2011). Outra motivação é a importância de se encontrar bons valores de limitantes inferiores para o PTHU da formulação CB-CTT, pois esses limitantes permitem avaliar a qualidade das soluções obtidas por outros métodos, ou até mesmo provar a otimalidade dessas soluções, já que em algumas instâncias a otimalidade ainda não foi provada.

Os métodos apresentados consistem em dividir o problema original em k subproblemas, a serem resolvidos posteriormente pelo otimizador CPLEX. Para particionar o problema original é usada a API da biblioteca *METIS* que implementa heurísticas eficientes de particionamento de grafos.

Dessa forma, o problema original é representado na forma de um grafo, que após ser particionado, irá representar k subproblemas independentes. No entanto, algumas restrições representadas pelas arestas do grafo cortadas durante o particionamento são relaxadas. Assim, a soma dos valores obtidos ao resolver cada um dos k subproblemas representa um limitante inferior para o problema original.

A principal diferença dos métodos *C-partition-null* e *C-partition-org*, em relação aos outros dois métodos (*U-partition* e *C-partition*), é que eles usam variáveis de cópia para representar as disciplinas que serão consideradas em mais de um subproblema, já que pertencem a algum currículo representado por alguma aresta cortada durante o particionamento do grafo.

As disciplinas consideradas em dois ou mais subproblemas, através das variáveis de cópia, permitem que os currículos representados por alguma aresta cortada durante o particionamento sejam considerados completamente, com todas as suas disciplinas, em um subproblema. Portanto, a utilização das variáveis de cópia torna a relaxação mais forte, pois cada subproblema se torna mais parecido com o problema original.

A partir disso, dentre os métodos que usam variáveis de cópia, o método *C-partition-org* apresenta uma relaxação mais forte que o método *C-partition-null*, já que nesse caso os coeficientes das variáveis de cópia, na função objetivo dos subproblemas, também é idêntico aos coeficientes, das respectivas variáveis, na função objetivo do problema original.

Após os experimentos computacionais, os resultados obtidos pelos quatro métodos se mostraram de boa qualidade, já que foram melhores, em média, que os valores obtidos

pelo trabalho que os motivou (HAO; BENLIC, 2011). Dentre os quatro métodos, o *C-partition-org* encontrou resultados melhores ou iguais que os outros três em todas as instâncias da formulação CB-CTT do ITC-2007.

A média dos valores de limitantes inferiores encontrada pelo método *C-partition-org* foi 20,46%, 11,02% e 4,29% melhor que a encontrada, respectivamente, pelos métodos *U-partition*, *C-partition* e *C-partition-null*. Além disso foi 22,70% melhor que o valor apresentado por Cacchiani et al. (2013). O método *C-partition-org* também atualizou os melhores valores de limitantes inferiores, atualmente conhecidos, em duas (comp03 e comp15) das 21 instâncias, além de ter provado a otimalidade em outras 15 instâncias.

Apesar do método *C-partition-org* ter obtido valor médio de limitantes inferiores menor do que o obtido por Bagger, Sørensen e Stidsen (2019), os resultados do método *C-partition-org*, com exceção nas instâncias comp05, comp12 e comp18, foram, em média, 4,53% melhores. Além disso, o método *C-partition-org* obteve os valores de limitantes inferiores de forma mais eficiente, ou seja, em menor tempo do que o método *Geração de Colunas* apresentado por Bagger, Sørensen e Stidsen (2019).

O método *C-partition-org* também obteve o melhor valor médio de *%gap* dentre os trabalhos na literatura. Essa medida é importante pois indica que, em média, a distância entre os valores de limitantes inferiores e superiores é menor.

Os resultados obtidos pelo método *C-partition-org* foram submetidos a uma análise de significância estatística juntamente com os resultados de outros trabalhos da literatura. Nessa análise, foi identificado que os resultados do método *C-partition-org* e os resultados apresentados por Bagger, Desaulniers e Desrosiers (2019) são os únicos, dentre os considerados na análise, que não se distanciam significativamente dos melhores valores de limitantes superiores atualmente conhecidos.

A partir disso, é apresentada como sugestão de continuidade a este trabalho a implementação de um novo método, denominado *U-partition-org*, baseado no método *U-partition*. O objetivo é adaptar o método *U-partition* para que use variáveis de cópia, e que os coeficientes dessas variáveis na função objetivo dos subproblemas sejam idênticos aos coeficientes das respectivas variáveis na função objetivo do problema original. Da mesma forma que o método *C-partition-org*, caso uma variável de cópia tenha contribuído com o aumento da função objetivo do subproblema, ao final da execução, esse valor é decrementado do valor obtido pelo CPLEX ao resolver aquele subproblema.

Sugere-se também a reimplementação dos métodos *U-partition-org* e *C-partition-org*, a partir de outro modelo matemático proposto para a formulação CB-CTT do ITC-2007. Nesse caso, espera-se que seja usado um modelo completo, ou seja, que considera as quatro restrições fracas (**S1**-**S4**) da formulação CB-CTT do ITC-2007, pois o modelo da primeira fase apresentado por Lach e Lübbecke (2012), usado nos métodos propostos neste trabalho,

considera apenas as três primeiras restrições fracas (**S1-S3**).

Considerando que em algumas instâncias a otimalidade ainda não foi provada, espera-se que esses métodos obtenham valores de limitantes inferiores melhores do que os já obtidos pelo método *C-partition-org*, principalmente nas instâncias comp05, comp12 e comp18, que são as únicas que possuem mais currículos do que disciplinas.

Referências

ACHÁ, R. A.; NIEUWENHUIS, R. Curriculum-based course timetabling with SAT and MaxSAT. *Annals of Operations Research*, v. 218, n. 1, p. 71–91, 2014.

AHUJA, R. K.; MAGNANTI, T. L.; ORLIN, J. B. *Network flows: Theory, algorithms, and applications*. Upper Saddle River, NJ, USA: Prentice-Hall Incorporation, 1993.

BABAEI, H.; KARIMPOUR, J.; HADIDI, A. A survey of approaches for university course timetabling problem. *Computers & Industrial Engineering*, v. 86, p. 43 – 59, 2015.

BAGGER, N.-C. F.; DESAULNIERS, G.; DESROSIERS, J. Daily course pattern formulation and valid inequalities for the curriculum-based course timetabling problem. *Journal of Scheduling*, v. 22, n. 2, p. 155–172, 2019.

BAGGER, N.-C. F. et al. Flow formulations for curriculum-based course timetabling. *Annals of Operations Research*, v. 280, n. 1, p. 121–150, 2019.

BAGGER, N.-C. F.; SøRENSEN, M.; STIDSEN, T. R. Dantzig–Wolfe decomposition of the daily course pattern formulation for curriculum-based course timetabling. *European Journal of Operational Research*, v. 272, n. 2, p. 430–446, 2019.

BAZARAA, M.; JARVIS, J.; SHERALI, H. *Linear Programming and Network Flows*. 4. ed. New York, NY, USA: John Wiley & Sons, 2009.

BELLIO, R. et al. Feature-based tuning of simulated annealing applied to the curriculum-based course timetabling problem. *Computers & Operations Research*, v. 65, p. 83–92, 2016.

BETTINELLI, A. et al. An overview of curriculum-based course timetabling. *TOP*, v. 23, n. 2, p. 313–349, 2015.

BOFILL, M. et al. The barcelogic SMT solver. In: GUPTA, A.; MALIK, S. (Ed.). *Lecture Notes in Computer Science*. Berlin, Heidelberg, Germany: Springer Berlin Heidelberg, 2008. p. 294–298.

BURKE, E. K. et al. A branch-and-cut procedure for the Udine course timetabling problem. *Annals of Operations Research*, v. 194, n. 1, p. 71–87, 2012.

BURKE, E. K. et al. Decomposition, reformulation, and diving in university course timetabling. *Computers & Operations Research*, v. 37, n. 3, p. 582 – 597, 2010.

CACCHIANI, V. et al. A new lower bound for curriculum-based course timetabling. *Computers & Operations Research*, v. 40, n. 10, p. 2466–2477, 2013.

CARVALHO, E. C. A. de. *Particionamento de grafos de aplicações e mapeamento em grafos de arquiteturas heterogêneas*. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal do Rio Grande do Sul, 2002.

COIN-OR FOUNDATION. *COIN-OR Branch-and-Cut MILP Solver*. 2011. Disponível em: <https://www.coin-or.org/downloading/>.

DELLA CROCE, F.; SALASSA, F. A variable neighborhood search based matheuristic for nurse rostering problems. *Annals of Operations Research*, v. 218, n. 1, p. 185–199, 2014.

DEVORE, J. L. *Probabilidade e estatística: para engenharia e ciências*. São Paulo, SP, Brasil: Cengage Learning, 2006.

DIESTEL, R. *Graph Theory*. 5. ed. Berlin, Heidelberg, Germany: Springer-Verlag, 2016.

DUECK, G. New optimization heuristics: The great deluge algorithm and the record-to-record travel. *Journal of Computational Physics*, v. 104, n. 1, p. 86–92, 1993.

FEO, T. A.; RESENDE, M. G. A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, v. 8, n. 2, p. 67–71, 1989.

FRIEDMAN, M. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, v. 11, n. 1, p. 86–92, 1940.

GAREY, M.; JOHNSON, D. *Computers and Intractability: A Guide to the Theory of NP-completeness*. 1. ed. New York, NY, USA: W H Freeman, 1979.

GASPERO, L. D.; MCCOLLUM, B.; SCHAERF, A. The second international timetabling competition (ITC-2007): Curriculum-based course timetabling (track 3). In: *Proceedings of The International Conference on Automated Planning and Scheduling*. Providence, Rhode Island, USA: ICAPS Incorporation, 2007. p. 1–5.

GLOVER, F. Tabu search and adaptive memory programming — advances, applications and challenges. In: BARR, R. S.; HELGASON, R. V.; KENNINGTON, J. L. (Ed.). *Interfaces in Computer Science and Operations Research: Advances in Metaheuristics, Optimization, and Stochastic Modeling Technologies*. Boston, MA, USA: Springer US, 1997. p. 1–75.

GOLDBARG, M. C.; LUNA, H. P. L. *Otimização Combinatória e programação linear: modelos e algoritmos*. 2. ed. Rio de Janeiro, RJ, Brasil: Elselvier, 2005.

GOTLIEB, C. C. The construction of class-teacher timetables. In: *IFIP Congress*. Amsterdam, Netherlands: North Holland Publishing Company, 1962. p. 73–77.

GUROBI OPTIMIZATION. *Gurobi Optimizer*. 2016. Disponível em: <https://www.gurobi.com/products/gurobi-optimizer>.

HAO, J.-K.; BENLIC, U. Lower bounds for the ITC-2007 curriculum-based course timetabling problem. *European Journal of Operational Research*, v. 212, n. 3, p. 464–472, 2011.

IBM ILOG. *CPLEX Optimizer*. 2017. Disponível em: <https://www.ibm.com/products/ilog-cplex-optimization-studio>.

KAMPKE, E. H. et al. A GRASP algorithm with path relinking for the university courses timetabling problem. In: *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics*. São Carlos, SP, Brasil: SBMAC, 2015. v. 3, n. 2, p. 1081–1087.

KAMPKE, E. H. et al. A network flow based construction for a GRASP+SA algorithm to solve the university timetabling problem. In: MISRA, S. et al. (Ed.). *Lecture Notes in Computer Science*. Cham, China: Springer International Publishing, 2019. p. 215–231.

KAMPKE, E. H. et al. Neighborhood analysis on the university timetabling problem. In: GERVASI, O. et al. (Ed.). *Lecture Notes in Computer Science*. Cham, China: Springer International Publishing, 2017. p. 148–164.

KARYPIS, G.; KUMAR, V. Multilevel k-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed Computing*, v. 48, n. 1, p. 96–129, 1998.

KARYPIS, G.; KUMAR, V. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, v. 20, n. 1, p. 359–392, 1999.

KERNIGHAN, B. W.; LIN, S. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, v. 49, n. 2, p. 291–307, 1970.

KIEFER, A.; HARTL, R. F.; SCHNELL, A. Adaptive large neighborhood search for the curriculum-based course timetabling problem. *Annals of Operations Research*, v. 252, n. 2, p. 255–282, 2017.

KIRKPATRICK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by simulated annealing. *Science*, v. 220, n. 4598, p. 671–680, 1983.

LACH, G.; LÜBBECKE, M. E. Curriculum based course timetabling: new solutions to Udine benchmark instances. *Annals of Operations Research*, v. 194, n. 1, p. 255–272, 2012.

LEWIS, R. A survey of metaheuristic-based techniques for university timetabling problems. *OR Spectrum*, v. 30, n. 1, p. 167–190, 2008.

LÜ, Z.; HAO, J.-K. Adaptive tabu search for course timetabling. *European Journal of Operational Research*, v. 200, n. 1, p. 235–244, 2010.

LÜ, Z.; HAO, J.-K.; GLOVER, F. Neighborhood analysis: A case study on curriculum-based course timetabling. *Journal of Heuristics*, v. 17, n. 2, p. 97–118, 2011.

MAURI, G. R. *Novas abordagens para representação e obtenção de limitantes e soluções para alguns problemas de otimização combinatória*. Tese (Doutorado em Computação Aplicada) — Instituto Nacional de Pesquisas Espaciais, 2008.

MAURI, G. R. Improved mathematical model and bounds for the crop rotation scheduling problem with adjacency constraints. *European Journal of Operational Research*, v. 278, n. 1, p. 120–135, 2019.

MAURI, G. R.; LORENA, L. A. N. Improving a lagrangian decomposition for the unconstrained binary quadratic programming problem. *Computers & Operations Research*, v. 39, n. 7, p. 1577–1581, 2012.

MORGENSTERN, C. *Algorithms for general graph coloring*. Tese (Doutorado em Ciência da Computação) — University of New Mexico, 1989.

MÜLLER, T. *Constraint-based Timetabling*. Tese (Doutorado em Ciência da Computação) — Faculty of Mathematics and Physics–Charles University in Prague, 2005.

MÜLLER, T. ITC2007 solver description: a hybrid approach. *Annals of Operations Research*, v. 172, n. 1, p. 429–446, 2009.

NEMENYI, T. P. B. *Distribution-free multiple comparisons*. Tese (Doutorado em Matemática) — Princeton University, 1963.

PINHEIRO, J. I. D. et al. *Estatística básica : a arte de trabalhar com dados*. Rio de Janeiro, RJ, Brasil: Elsevier, 2009.

R FOUNDATION FOR STATISTICAL COMPUTING. *R: A Language and Environment for Statistical Computing*. 2018. Disponível em: <https://www.R-project.org/>.

RIBEIRO, C. C.; URRUTIA, S. Scheduling the brazilian soccer tournament: Solution approach and practice. *Interfaces*, v. 42, n. 3, p. 260–272, 2012.

RIBEIRO, G. M.; LORENA, L. A. N. Lagrangean relaxation with clusters for point-feature cartographic label placement problems. *Computers & Operations Research*, v. 35, n. 7, p. 2129–2140, 2008.

ROCHA, W. S. *Algoritmo GRASP para o Problema de Tabela-horário de Universidades*. Dissertação (Mestrado em Informática) — Universidade Federal do Espírito Santo, 2013.

RUSSELL, S. J.; NORVIG, P. *Artificial intelligence: a modern approach*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1995.

SANTOS, H. G.; OCHI, L. S.; SOUZA, M. J. A tabu search heuristic with efficient diversification strategies for the class/teacher timetabling problem. *Journal of Experimental Algorithmics*, ACM, New York, NY, USA, v. 10, n. 2.9, p. 1–16, 2005.

SCHAERF, A. A survey of automated timetabling. *Artificial Intelligence Review*, v. 13, n. 2, p. 87–127, 1999.

SEGATTO, E. A. *Um estudo de estruturas de vizinhanças no GRASP aplicado ao Problema de Tabela-Horário para Universidades*. Dissertação (Mestrado em Informática) — Universidade Federal do Espírito Santo, 2017.

SEGATTO, E. A. et al. Um algoritmo GRASP com cadeia de kempe aplicado ao problema de tabela-horário para universidades. In: *Anais do XLVII Simpósio Brasileiro de Pesquisa Operacional*. Rio de Janeiro, RJ, Brasil: SOBRAPO, 2015. p. 2643–2654.

WOLSEY, L. A. *Integer programming*. 1. ed. New York, NY, USA: John Wiley & Sons, 1998.