TeoCOMP-NE 2023 - III Escola de Teoria da Computação NE

4-6 Oct 2023
Recife
Brazil

Table of contents

Imersão de caminhos curtos em (di)grafos

Anna Karolinna Maia[1], **Matheus Ribeiro Alencar**[1]

[1]Departamento de Computação, Centro de Ciências, Universidade Federal do Ceará
CEP 60.440-900 – Fortaleza – CE – Brasil

karolmaia@ufc.br, matheusriale@alu.ufc.br

Abstract. *Immersions are an important containment relation between (di)graphs. Determining if a digraph H is immerse in a digraph G is an NP-complete problem in the general case. In this work, we propose a polynomial-time algorithm for detecting the immersion of small path in any (di)graph.*

Resumo. *Imersão é uma importante relação de contenção entre (di)grafos. Determinar se um digrafo H está imerso em um digrafo G é um problema NP-completo no caso geral. Neste trabalho, propomos um algoritmo polinomial para a detecção de imersões de caminhos pequenos em um (di)grafo qualquer.*

1. Introdução

Nesse trabalho, nós seguimos a notação padrão de grafos, digrafos (ou grafos direcionados) e complexidade computacional que podem ser encontradas em [Bondy and Murty 2008, Bang-Jensen and Gutin 2008, Garey and Johnson 1990].

Existem diferentes formas de verificarmos de um (di)grafo H está contido em um (di)grafo G. Por exemplo, podemos procurar uma cópia exata de H em G, isto é, checar se G possui um sub(di)grafo isomorfo a H. Mas podemos também nos perguntar se G possui um grafo "similar a" H, ou obtido a partir de H. O significado de "similar"ou "obtidos a partir de" variam.

O estudo dessas relações de contenção desempenham um papel fundamental em teoria dos grafos, dado que muitas classes interessantes são definidas pela proibição de certos subgrafos (induzidos). Grafos planares são um exemplo bem conhecido: eles foram caracterizados por Kuratowski como os grafos que não possuem grafos "obtidos a partir" do $K_{3,3}$ ou K_5. Outro exemplo notório considerando subgrafos induzidos são os grafos perfeitos. O teorema forte dos grafos perfeitos afirma que um grafo é perfeito se e somente se não contém um subgrafo induzido que é um ciclo ímpar ou o complemento de um ciclo ímpar [Chudnovsky et al. 2006]. Outras classes de grafos determinadas pela proibição de um conjunto de subgrafos podem ser vistas em [Chudnovsky and Seymour 2007, Granot et al. 2000].

Nós examinamos uma dessas relações de contenção, chamada de *imersão*.

Definição 1. *Imersão*: Dizemos que H é imerso em G, ou que G possui uma imersão de H se existe mapeamento ϕ tal que:

- $\phi(v) \in V(G), \forall v \in V(H)$;
- $\forall e = uv \in E(H)$, $\phi(e)$ é um caminho (direcionado) em G entre $\phi(u)$ e $\phi(v)$;
- Se $e, f \in E(H)$ são distintas, então $\phi(e)$ e $\phi(f)$ são disjuntos em arestas;
- $\phi(u) \neq \phi(v), \forall u, v \in V(H)$ onde $u \neq v$.

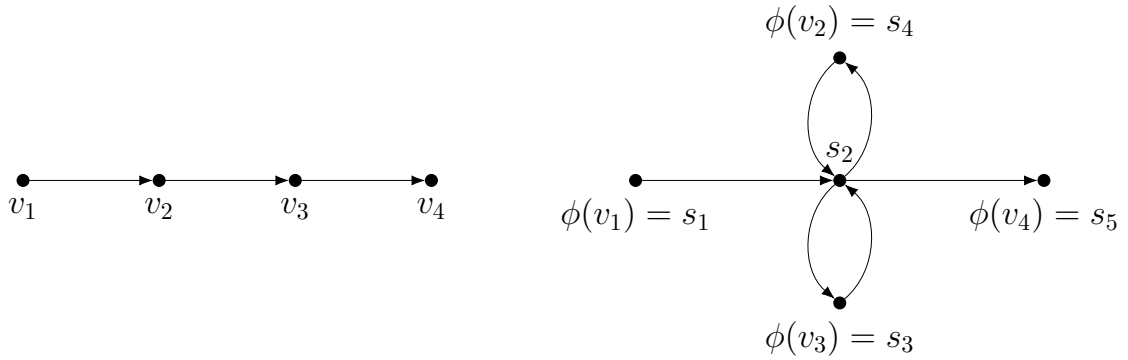


Figura 1. Grafos P_4 e G respectivamente.

A Figura 1 apresenta um exemplo de imersão de H (nesse caso um caminho com 4 vértices) em G.

Note que não existe um subgrafo P_4 em G, no entanto, ainda conseguimos encontrar imersão de P_4 em G.

Na figura 1, temos os vértices $\{v_1, v_2, v_3, v_4\}$ de P_4 mapeados respectivamente nos vértices $\{s_1, s_4, s_3, s_5\}$ de G. Ou seja, temos o seguinte: $\phi(v_1) = s_1$; $\phi(v_2) = s_4$; $\phi(v_3) = s_3$; $\phi(v_4) = s_5$.

O mapeamento das arestas nesse caso possui uma peculiaridade. Não mapearemos sempre arestas de uma para uma, pelo conceito de imersão podemos mapear arestas em caminhos, o que ocorre nesse caso para encontrarmos a imersão de P_4 em G. Caso nos restrinjamos a mapear arestas apenas de uma para uma, não encontraremos imersão, levando a uma solução errada para nosso problema.

Nesse caso, as arestas de $E(P_4) = \{v_1v_2, v_2v_3, v_3v_4\}$ serão mapeadas em $\{\{s_1s_2, s_2s_4\}, \{s_4s_2, s_2s_3\}, \{s_3s_2, s_2s_5\}\}$ respectivamente. Note que pelo conceito de imersão, caso o vértice s_2 tivesse sido mapeado, não poderíamos utilizar os caminhos que passam por ele. Ao final teremos o seguinte mapeamento para as arestas: $\phi(v_1v_2) = \{s_1s_2, s_2s_4\}$; $\phi(v_2v_3) = \{s_4s_2, s_2s_3\}$; $\phi(v_3v_4) = \{s_3s_2, s_2s_5\}$.

Nós consideramos o seguinte problema para um digrafo fixo H:

H-IMERSÃO
Entrada: Um digrafo G.
Pergunta: G possui uma imersão de H?

Para grafos não direcionados, o problema acima possui solução de tempo polinomial [Robertson and Seymour 1995]. Já para digrafos, H-IMERSÃO é NP-completo [Chudnovsky et al. 2012]. Nesse mesmo artigo, os autores um algoritmo polinomial para encontrar a imersão de um digrafo fixo H em um digrafo semi-completo G. Em [Kim and Seymour 2013], é provado que para alguns digrafos, o problema pode ser resolvido em tempo polinomial. Para inteiros $i, j \geq 0$ é denotado por $I_{i,j}$ o digrafo com dois vértices v_1, v_2 e com i arcos paralelos de v_1 para v_2 e j arcos paralelos de v_2 para v_1. Para um inteiro $k \geq 1$ denota-se por A_k o digrafo acíclico com $2k$ vértices tal que o grafo não direcionado subjacente é um ciclo de tamanho $2k$ e todo vértice possui grau de saída

igual zero, ou grau de entrada igual a zero, chamamos A_k de ciclo alternado. É provado que testar a imersão de $I_{1,2}$ e testar a imersão de A_k (para um k fixo) pode ser feito em tempo polinomial.

Em [DeVos et al. 2012], são estudadas condições em digrafos que implicam em certas imersões. Mais especificamente, são apresentadas condições de grau para um digrafo completo H ser imerso em um digrafo euleriano G, além de serem apresentados digrafos não-eulerianos com grau de entrada e saída suficientemente grande que não contém imersão do digrafo completo. Já em [Lochet 2019], é provada a existência de uma função $h(k)$ tal que todo digrafo simples com grau mínimo de saída maior que $h(k)$ contém uma imersão do torneio transitivo com k vértices.

Até onde sabemos, entretanto, a complexidade de imergir um caminho direcionado em um digrafo qualquer ainda não foi determinada. Nós estudamos a complexidade H-IMERSÃO quando H é um caminho direcionado com 4 vértices (P_4) e G é um digrafo qualquer. Na próxima sessão, apresentamos um algoritmo polinomial para resolver o problema.

2. Algoritmo para detecção de Imersão de P_4

Apresentamos um algoritmo para detecção de pelo menos uma imersão de P_4 em um digrafo G, bem como a ideia da sua corretude.

Algorithm 1 Detecção de pelo menos uma imersão de P_4

1: **Entrada:** Grafo G.
2: **Saída:** Booleano True caso exista imersão de P_4 em G, False caso contrário.
3: **if** $\exists P_4 \in G$ **then**
4: Return True
5: **else if** $\nexists$ Ciclo $\in G$ **then**
6: Return False
7: **end if**
8: **if** $\exists P_3 = \{uv, vw\} \in G$ com $u, v, w \in G$ **then**
9: **for** all $P_3 \in G$ **do**
10: **if** $v \in V(Ciclo_1), V(Ciclo_2), ..., V(Ciclo_n)$ com $n \geq 2$ e $u, w \notin V(Ciclo_1), V(Ciclo_2), ..., V(Ciclo_n)$ **then**
11: Return True
12: **end if**
13: **end for**
14: **end if**
15: Return False

2.1. Caso em que tenhamos um P_4 em G.

Esse subcaso está sendo verificado nas linhas 3 e 4, se possuirmos um P_4 já possuímos imersão de P_4.

2.2. Caso em que não tenhamos um P_4 em G e não tenhamos nenhum ciclo em G.

Subcaso verificado nas linhas 5 e 6, como o único caso em que não tivermos P_4 e conseguimos ter uma imersão de P_4, necessariamente tem que passar por ciclos, retornaremos falso, pois nesse caso G não possui ciclos.

2.3. Caso tenhamos P_3 em G e existam ciclos em G (envolvendo v).

Para todo caminho direcionado de tamanho 3, de formato $P = \{uv, vw\}$, verificaremos se o vértice central v pertence a mais de um ciclo e se ele é o único vértice de P que participa de um caminho. Caso v pertença a mais de um ciclo além dos ciclos que contenham u, w (caso existam), teremos uma imersão de P_4 em G. Esse caso é tratado entre as linhas 8 a 14.

2.4. Caso tenhamos P_3 em G e não existam pelo menos 2 ciclos envolvendo apenas o vértice central de P_3.

Único caso restante, tratado ao final do algoritmo na linha 15.

3. Conclusão

Como principal resultado desse trabalho, somos capazes de identificar através de um algoritmo polinomial quando existem imersões de caminhos direcionados pequenos (até 4 vértices) em digrafos quaisquer. A questão natural que pretendemos analisar na sequência dessa pesquisa é se o problema continua tratável para caminhos maiores de tamanho fixo, e se existe um algoritmo geral para detecção desses caminhos em digrafos quaisquer em tal caso.

Referências

Bang-Jensen, J. and Gutin, G. Z. (2008). *Digraphs: Theory, Algorithms and Applications*. Springer Publishing Company, Incorporated, 2nd edition.

Bondy, J. and Murty, U. (2008). *Graph Theory*. Springer Publishing Company, Incorporated, 1st edition.

Chudnovsky, M., Fradkin, A., and Seymour, P. (2012). Tournament immersion and cutwidth. *Journal of Combinatorial Theory, Series B*, 102(1):93–101.

Chudnovsky, M., Robertson, N., Seymour, P., and Thomas, R. (2006). The strong perfect graph theorem. *Ann. of Math. (2)*, 164(1):51–229.

Chudnovsky, M. and Seymour, P. (2007). Excluding induced subgraphs. In *Surveys in combinatorics 2007*, volume 346 of *London Math. Soc. Lecture Note Ser.*, pages 99–119. Cambridge Univ. Press, Cambridge.

DeVos, M., McDonald, J., Mohar, B., and Scheide, D. (2012). Immersing complete digraphs. *European Journal of Combinatorics*, 33(6):1294–1302.

Garey, M. R. and Johnson, D. S. (1990). *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.

Granot, D., Granot, F., and Zhu, W. R. (2000). Naturally submodular digraphs and forbidden digraph configurations. *Discrete Appl. Math.*, 100(1-2):67–84.

Kim, I. and Seymour, P. (2013). *On Containment Relations on Digraphs*. PhD thesis, Princeton, NJ : Princeton University.

Lochet, W. (2019). Immersion of transitive tournaments in digraphs with large minimum outdegree. *Journal of Combinatorial Theory, Series B*, 134:350–353.

Robertson, N. and Seymour, P. D. (1995). Graph minors. xiii. the disjoint paths problem. *Journal of combinatorial theory, Series B*, 63(1):65–110.

GBL no Ensino de Aspectos Teóricos da Computação

Alcides Cunha de Andrade[1], Jeane Cecília Bezerra de Melo[1]

[1]Departamento de Computação – Universidade Federal Rural de Pernambuco (UFRPE)
alcides.ca0@gmail.com, jeane.bmelo@ufrpe.br

Abstract. *This paper consists of a study of Game-Based learning (GBL) approaches to teaching theoretical aspects of computing, with the aim of exploring the suitability of the teaching methodology for this area of knowledge. The methodology is promising and there is a demand for educational games that support its use.*

Resumo. *O presente trabalho consiste de um estudo sobre abordagens Game-Based Learning (GBL) para o ensino de Aspectos Teóricos de Computação, objetivando pesquisar a adequação da metodologia de ensino para esta área do conhecimento. A metodologia se mostra promissora, havendo demanda por jogos educacionais que favoreçam a sua utilização.*

1. Introdução

De acordo com as Diretrizes Curriculares Nacionais, Aspectos Teóricos da Computação permeiam os conteúdos curriculares de todos os cursos de graduação em computação [Brasil 2016]. Os fundamentos matemáticos da computação, que abrangem Teoria da Computação, Linguagens Formais e Autômatos, têm um papel essencial na formação dos estudantes, além de corroborar com outras disciplinas, tais como Compiladores e Inteligência Artificial [Souza *et al.* 2016]. Concomitantemente, estudos relacionados à evasão em cursos superiores de computação, indicam as dificuldades em disciplinas do curso ligadas à matemática, bem como a didática dos professores, como fatores que corroboram para a desistência de tais cursos [Fukao *et al.* 2023].

Uma pesquisa realizada por Qurat-ul-Ain *et al.* (2019), discorre sobre mudanças pedagógicas benéficas decorrentes da utilização de tecnologias no processo de ensino-aprendizagem de Ciência da Computação, dentre as ferramentas que integram o estudo estão os jogos educacionais. Segundo Carvalho (2015), uma metodologia pedagógica que abrange a concepção, desenvolvimento, uso e aplicação de jogos na educação é denominada *Game-Based Learning* (GBL), traduzida, por alguns autores, como Aprendizagem Baseada em Jogos. Carvalho destaca elementos de (bons) jogos que tornam sua utilização no apoio ao processo de ensino-aprendizagem vantajosa: interatividade, percepção de acertos e erros, *feedback*, entre outros.

O presente artigo traz um estudo inicial sobre abordagens GBL para o ensino de Aspectos Teóricos de Computação, objetivando pesquisar a adequação da metodologia para esta área do conhecimento, bem como detectar elementos que contribuam para sua utilização. O trabalho encontra-se dividido como segue: a Seção 2, apresenta a metodologia *Game-Based Learning* (GBL). Estudos que utilizam a GBL no ensino de diferentes aspectos teóricos da computação compreendem a Seção 3. Considerações finais e trabalhos futuros são discutidos na Seção 4.

2. Metodologia

Definições e características que compõem essa seção, estão relacionadas à metodologia aqui considerada para o ensino de Aspectos Teóricos de Computação, *Game-Based Learning* (GBL), que tem se mostrado promissora em diversas áreas do conhecimento.

2.1. Game-Based Learning

Plass *et al.* (2015), destacam que diferentes definições para *Game-Based Learning* dão ênfase a utilização de jogos, digitais ou não, com resultados de aprendizagem definidos, entendendo-se jogo como "um sistema no qual os jogadores se envolvem num conflito artificial, definido por regras, cujo resultado é quantificável". Dentre os argumentos para a utilização de jogos para ensino-aprendizagem, a característica mais citada é a função motivacional dos jogos, os quais conseguem manter os estudantes envolvidos durante longos períodos de tempo, seguida por outros argumentos tais como: engajamento, adaptabilidade, e lidar com falhas.

Carvalho (2015) destaca o avanço de propostas para utilização de GBL em diferentes áreas, um fato que se observa, inclusive, em estudos recentes [Camacho-Sánchez *et al.* 2023]. Em seu artigo, Carvalho (2015) apresenta jogos, os quais destinam-se a públicos e faixas etárias distintas, que podem ser integrados a diferentes práticas pedagógicas.

3. Abordagens GBL para o Ensino de Aspectos Teóricos de Computação

A presente seção traz abordagens GBL para diferentes tópicos relacionados a aspectos teóricos de Computação.

3.1. Linguagens Formais e Teoria dos Autômatos

Santini *et al.* (2022) realizaram um mapeamento sistemático de jogos destinados ao ensino de Linguagens Formais e teoria do Autômatos (LFA). Os autores destacam que apesar do potencial educacional, há poucos jogos educacionais voltados para LFA, e sugerem que novas abordagens lúdicas sejam desenvolvidas. Como exemplos de jogos para LFA testados com o público alvo, temos: Automata Toy Factory [Tomizawa and Campano Junior 2021] e Labirinto GRamágico [Garozi *et al.* 2021]. A maioria dos jogos analisados trazem mecanismos reconhecedores de autômatos finitos, determinísticos e não determinísticos, autômatos a pilha e Máquina de Turing.

3.2. Teoria dos Grafos

A modelagem de problema através de grafos é comum em Ciência da Computação. Uma proposta de GBL para o ensino de Grafos apresentada no trabalho de Santos (2021), propõe o jogo chamado Formigas em Grafo, no qual o jogador precisa alimentar formigueiros, representados por nós no grafo, no menor tempo possível. Honda (2022) também apresenta um jogo para ensino de grafos, no qual o usuário precisa entregar todas as pizzas, buscando economizar combustível. As duas propostas utilizam o conceito de GBL para ensino do conceito Caminho Ótimo em grafos.

3.3. Algoritmos

Algoritmos compreendem um dos pilares da Computação, sendo fundamentais no desenvolvimento de lógica de programação e na resolução de problemas. Em [Bastiani *et al.* 2019] foi realizado um estudo para avaliar a disciplina de algoritmos e suas dificuldades, a fim de criar uma proposta de jogo sério desplugado como um método de apoio para a aprendizagem de algoritmos. Essa proposta de jogo utiliza cartas e RPG chamado DExHA como auxiliar de aprendizado da lógica dos problemas, sem direcionar para uma linguagem de programação específica. O jogo forma equipes de Hackers e Desenvolvedores para disputarem entre si, percorrendo um caminho em um tabuleiro, resolvendo algoritmos, de modo a atingir o objetivo e vencer. A proposta utiliza GBL para amenizar a desmotivação dos alunos e estimular o raciocínio lógico.

Na literatura há outros exemplos de utilização de GBL para ensino de aspectos teóricos da computação, tanto no ensino superior como no ensino médio e fundamental. Porém, dada a limitação de espaço, foram discorridas algumas abordagens de modo a ilustrar sua utilização nesta área do conhecimento

4. Considerações Finais e Trabalhos Futuros

Aspectos teóricos da computação, presentes em matérias tais como Teoria da Computação, incluindo Autômatos, Linguagens Formais e Algoritmos, são fundamentais para a formação de um profissional da área de computação. Porém, estudos indicam que disciplinas com conteúdo matemático são apontadas como complexas pelos estudantes, e surgem como um dos fatores relacionados a altas taxas reprovação, bem como evasão dos cursos superiores de computação. Portanto, torna-se necessário aplicar novas abordagens de ensino nessas disciplinas, de modo a promover a motivação, o engajamento e que favoreçam o envio de *feedbacks* durante o processo de aprendizagem. Atualmente, os jogos têm se mostrado como fortes aliados nas propostas de solução para diversificar o ensino, estabelecendo a metodologia *Game-Based Learning* (GBL) como promissora no ensino de diversas áreas do conhecimento.

O presente trabalho apresentou algumas abordagens GBL para o ensino aspectos teóricos da computação. Através de uma revisão de literatura, observou-se uma demanda por jogos que auxiliem no processo de ensino-aprendizagem dessa área do conhecimento. Estudos sobre jogos educacionais desenvolvidos recentemente, indicam que a metodologia é promissora, porém, há demandas por jogos que abranjam mais tópicos, avaliando sua adequação às demandas desse público alvo.

Como trabalho futuro, pretende-se aprofundar a pesquisa sobre os jogos educacionais disponíveis para a área, de modo a detectar problemas e propor soluções, avaliando-as segundo as diretrizes da GBL.

References

Bastiani, E., Roman, L. M., & Botelho, V. (2019). “Proposta de um Jogo Sério e Desplugado para o Ensino de Algoritmos”, Revista do CCEI, 24(39), 61-74.

Brasil. (2016). Ministério da Educação. Conselho Nacional de Educação. Câmara de Educação Superior. Resolução CNE/CES nº 05, de 16 de novembro de 2016,

http://portal.mec.gov.br/index.php?option=com_docman&view=download&alias=52101-rces005-16-pdf&category_slug=novembro-2016-pdf&Itemid=30192.

Camacho-Sánchez, R., Manzano-León, A., Rodríguez-Ferrer, J. M., Serna, J., & Lavega-Burgués, P. (2023). Game-based learning and gamification in physical education: a systematic review. Education Sciences, 13(2), 183.

Carvalho, C. V. (2015). "Aprendizagem baseada em jogos-Game-based learning", In: II World Congress on Systems Engineering and Information Technology, p. 176-181.

Fukao, A., Colanzi, T., Martimiano, L., & Feltrim, V. (2023). "Estudo sobre Evasão nos Cursos de Computação da Universidade Estadual de Maringá". In Anais do III Simpósio Brasileiro de Educação em Computação, (pp. 86-96). Porto Alegre: SBC. doi:10.5753/educomp.2023.228209

Garozi, P. H. T., Campano Junior, M. M., and Costa, Y. M. (2021). Labirinto gramágico: Um jogo educativo para o ensino de gramáticas regulares. In Anais Estendidos do XX Simpósio Brasileiro de Jogos e Entretenimento Digital, pages 489–498. SBC.

Honda, F., Pires, F., Pessoa, M., & Maia, J. (2022, October). Cadê minha pizza? um jogo para exercitar matemática e pensamento computacional através de grafos. In Anais Estendidos do XXI Simpósio Brasileiro de Jogos e Entretenimento Digital (pp. 876-885). SBC.

Plass, J. L., Homer, B. D., & Kinzer, C. K. (2015) "Foundations of Game-Based Learning", Educational Psychologist, 50:4, 258-283, DOI: 10.1080/00461520.2015.1122533.

Qurat-ul-Ain, Shahid, F., Aleem, M., Islam, M. A., Iqbal, M. A., & Yousaf, M. M. (2019). "A Review of Technological Tools in Teaching and Learning Computer Science", Eurasia Journal of Mathematics, Science and Technology Education, 15(11), em1773. https://doi.org/10.29333/ejmste/109611

Santos, A. V., & Ferreira, A. B. (2021). "Formigas em grafo: Uma proposta de jogos digital e de tabuleiro para o ensino de algoritmos em grafos", Revista de Sistemas e Computação-RSC, 11(3).

Santini, L., Campano Junior, M., Felinto, A., & Aylon, L. (2022). Jogos no Ensino de Linguagens Formais e Autômatos: Um Mapeamento Sistemático. In Anais Estendidos do XXI Simpósio Brasileiro de Jogos e Entretenimento Digital, (pp. 886-895). Porto Alegre: SBC. doi:10.5753/sbgames_estendido.2022.226064

Souza, Í., Matos, E., Santos, D., & Sousa, R. (2016). "Recursos Computacionais para Suporte ao Ensino de Teoria da Computação, Linguagens Formais e Autômatos", In Anais do XXIV Workshop sobre Educação em Computação, (pp. 2373-2382). Porto Alegre: SBC. doi:10.5753/wei.2016.9681.

Tomizawa, M. M. and Campano Junior, M. M. (2021). Automata toy factory: Um jogo educativo para ensino de autômato com pilha. In Anais Estendidos do XX Simpósio Brasileiro de Jogos e Entretenimento Digital, pages 389–397. SBC.

N Grafos para Lógica de Primeira Ordem: Um estudo

Jorge Farias[1], **Anjolina de Oliveira** [1], **Ruan Carvalho** [2]

[1]Centro de Informática – Universidade Federal de Pernambuco (UFPE)

[2] Departamento de Computação – Universidade Federal Rural de Pernambuco (UFRPE)

jsfj@cin.ufpe.br

Abstract. *This abstract is based on the master's thesis developed by the authors. The work is developing an application of proof theory, where the study of N-Graphs proposed for classical propositional logic [de Oliveira 2001] is being continued. The main motivation is to determine the conditions for validating proofs with N-Graphs for first-order logic.*

Resumo. *Esse resumo é baseado na dissertação de mestrado desenvolvida pelos autores. O trabalho está desenvolvendo uma aplicação da teoria da prova, onde está sendo dado prosseguimento ao estudo de N-Grafos propostos para a lógica proposicional clássica [de Oliveira 2001]. A principal motivação é determinar as condições de validação de provas com N-Grafos para a lógica de primeira ordem.*

1. Introdução

As pesquisas direcionadas à área da teoria da prova buscam estudar os conceitos de prova formais e os fundamentos relacionados. O desenvolvimento de sistemas dedutivos e de técnicas como a normalização de provas são exemplos de investigações relevantes na área. A teoria da prova tem diversas aplicações tanto na matemática, na filosofia, assim como na ciência e teoria da computação.

Durante os anos 80 alguns estudos sobre "Lógica e Computação" foram propostos com a intenção de dar um tratamento lógico para problemas computacionais. Como, por exemplo, o trabalho sobre Lógica Linear introduzido por Girard [Girard 1987]. Após isso a lógica linear se firmou como o formalismo mais utilizado para o estudo da interface entre lógica e computação [de Oliveira 2001]. Um dos principais aspectos da lógica linear são as *proof-nets*, elas trazem a prova em uma perspectiva geométrica fazendo uma ligação entre o cálculo de sequentes e a dedução natural.

Os N-grafos são um sistema de provas proposto por de Oliveira [de Oliveira 2001] para a lógica proposicional. Eles possuem dois tipos de regras assim como o cálculo de sequentes: regras estruturais e lógicas. Porém, as regras de inferência são baseadas na dedução natural. As provas são representadas geometricamente por meio de grafos direcionados (dígrafos) e trazem um sistema de múltipla conclusão, com isso os n-grafos se assemelham a outros sistemas de múltipla conclusão para lógica clássica, que apresentam uma solução para a falta de simetria da dedução natural. O critério de corretude desse sistema utiliza a mesma técnica para *proof-nets* proposta por Danos Regnier [Danos and Regnier 1989] e a partir desse trabalho tornou-se possível definir impérios usando os grafos DR.

2. Lógica de Primeira Ordem

A lógica proposicional(LP), tem linguagem declarativa e sua forma de escrita é baseada em uma relação de verdade entre as sentenças e os mundos possíveis, porém tem um problema que é a falta de capacidade de expressão para descrever um ambiente composto por múltiplos objetos. A lógica de primeira ordem(LPO), ou como também é chamada lógica de predicados, traz uma solução para isso. A LPO faz uma suposição que o mundo é feito de objetos com certas relações e propriedades entre eles. A LP trata fatos, já a LPO trata objetos, relações, propriedades e funções. A sintaxe traz dois quantificadores:

- **quantificador universal** ($\forall$)
  Esse quantificador é lido da forma *"para todo"* como, por exemplo: *"Para todo y, se y é médico, então y cursou medicina"*.
  Formalmente fica,
  $\forall x$**Q**, onde Q é uma sentença lógica, o quantificador afirma que Q é verdade para **todo** objeto x.
- **quantificador existencial** ($\exists$)
  Esse quantificador é lido da forma *"existe algum"* como, por exemplo: *"Existe uma amiga de João, essa amiga de João é enfermeira"*. Diferencia-se do universal porque não se refere a todos os elementos de um conjunto.
  Formalmente fica,
  $\exists xP$, onde P é verdade para **algum** objeto no universo.

3. Dedução Natural

Introduzida por Gentzen [Gentzen 1935], a dedução natural é um sistema de prova dedutivo utilizado para demonstrações formais da lógica. Esse sistema consiste em um conjunto de regras de dedução [Gentzen 1935], sendo que a partir delas as provas ficam próximas dos modelos naturais de raciocínio. Essas regras, como a introdução da implicação, buscavam substituir os axiomas lógicos, que não são naturais. Nesse sistema cada conectivo é definido com um par de regras: introdução e eliminação. Na década de 1960, Dag Prawitz continuou o estudo da dedução natural e demonstrou um teorema de eliminação de corte, nele Prawitz mostra que a partir de uma dedução em dedução natural chega-se em sua forma normal por meio de operações de redução [Schroeder-Heister 2018].

3.1. Dedução Natural para Lógica de Primeira ordem

A dedução natural para lógica de primeira ordem é semelhante a da lógica proposicional, no caso específico novas regras são utilizadas para tratar os quantificadores. As regras de exclusão do quantificador universal($\forall$) e introdução do quantificador existencial($\exists$) são simples, nelas basta apenas remover ou adicionar o quantificador Porém, as regras de introdução do quantificador universal($\forall I$) e exclusão do quantificador existencial($\exists E$) precisam de um pouco mais de cuidado, pois tem suas restrições

4. Cálculo de Sequentes

Um cálculo para lógica clássica (LK) e intuicionista (LJ) foi introduzido por Gentzen [Gentzen 1935], a fim de provar a consistência de Hilbert (mais precisamente, livre de contradição) para lógica pura e Aritmética de Peano, baseados na noção de sequentes, o

cálculo de sequentes. Um sequente consiste em duas sequências de fórmulas separadas por uma ”catraca”($\vdash$) [Paolo Mancosu 2021].

O que diferencia o cálculo de sequentes da dedução natural não é a estrutura das fórmulas, mas sim as estrutura de sequentes e a formulação das regras de inferência como as regras que operam em sequências.

4.1. Cálculo de Sequentes para Lógica de Primeira ordem

Assim como a dedução natural, o cálculo de sequentes também tem suas regras para a lógica de primeira ordem, dentro das regras operacionais temos as regras para os quantificadores:

4.1.1. Quantificador Universal

A regra ∀R qualificada pelo ponto de exclamação ”!”é uma regra crítica. Isso significa que só pode ser aplicado se a variável livre indicada a na premissa não aparece na sequência inferior, ou seja, não ocorre em Γ,Θ ou em $\forall x$ A(x).

$$\frac{\Gamma \vdash \Theta, A(a)}{\Gamma \vdash \Theta, \forall x A(x)}^{\forall R!}$$

4.1.2. Quantificador Existencial

A regra ∃L quantificada pelo ponto de exclamação ”!”é também uma regra crítica. Também é sujeito à restrição que a variável livre a indicada na premissa da regra (autovariável) não aparece em Γ, Θ ou $\exists x$ A(x).

$$\frac{A(a), \Gamma \vdash \Theta}{\exists x A(x), \Gamma \vdash \Theta}^{\exists L!}$$

5. *Proof Nets*

As *proof-nets* são representações gráficas concisas de provas em MLL- (*Multiplicative Linear Logic*, o -”significa sem constantes), elas são a dedução natural para a lógica linear. Esse fragmento é composto pelos conectivos $\otimes$ (conjunção multiplicativa) $\wp$ (disjunção multiplicativa). As provas são representadas por ”estruturas de provas”, formadas por fórmulas e *links*. *Links* são geralmente representados como:

$$\frac{X_1, \ldots, X_m}{X_m + 1, \ldots, X_m + n}$$

5.1. *Proof Nets* para Lógica de Primeira Ordem

Assim como a dedução natural ou o cálculo de sequentes, as *proof-nets* também tem sua linguagem de primeira ordem para MLL-, ela necessita da adição de dois novos *links*: o quantificador universal (∀) e o quantificador existencial (∃).

$$para \quad todo : \frac{A}{\forall x A} \qquad existe : \frac{A[t/x]}{\exists x A}$$

A lógica de primeira ordem tem algumas restrições nas suas regras de introdução do quantificador universal ($\forall I$) e eliminação do quantificador existencial ($\exists E$). Como as *proof-nets* só tem as regras de introdução (*links*) a restrição levada em consideração é apenas a da regra de introdução do quantificador universal. Para introduzi-lo é necessário ter a certeza da não ocorrência da variável livre em qualquer fórmula não descartada, essa variável é chamada de *eigenvariable*, a mesma nomenclatura utilizada por Gentzen.

6. N Grafos para Lógica de Primeira Ordem

Nos N-Grafos as provas são abordadas por uma perspectiva geométrica: o critério de corretude utiliza a técnica da lógica linear para proof-nets proposta por Danos & Regnier [Danos and Regnier 1989]. Na prova de sequentização o teorema split é usado de modo semelhante à proposta de Girard [Girard 1987]. O conectivo da implicação é concebido conforme a abordagem de Statman e são adotadas algumas definições usadas por Carbone no seu trabalho com grafos de fluxo lógico (do inglês logical flow graphs).

Como visto, os métodos de provas tem suas versões para lógica de primeira ordem, e o objetivo desse trabalho foi desenvolver a versão dos n-grafos para esse escopo lógico. Essa é uma etapa ainda em andamento.

Pode-se observar diversas semelhanças entre as *proof-nets* e os n-grafos, o primeiro tem sua versão para lógica de predicados apenas com as regras de introdução ($\forall I, \exists I$), logo para os n-grafos as regras de exclusão também devem ser propostas.

Por se tratar de uma dissertação, os métodos de prova citados nas sessões anteriores foram minuciosamente estudados e documentados.

7. References

Referências

Danos, V. and Regnier, L. (1989). The structure of multiplicatives. *Archive for Mathematical Logic*, 28(3):181–203.

de Oliveira, A. G. (2001). *Proofs from a Geometric Perspective*. PhD thesis, Universidade Federal de Pernambuco, Centro de Informática, Recife.

Gentzen, G. (1935). Untersuchungen Über das logische schliesen. ii. *Mathematische Zeitschrift*, 39:405–431.

Girard, J.-Y. (1987). Linear logic. *Theor. Comput. Sci.*, 50(1):1–102.

Paolo Mancosu, Sergio Galvan, R. Z. (2021). *An Introduction To Proof Theory*, volume 1. Oxford University press, Oxford University Press 198 Madison Avenue, New York, NY 10016, United States of America, first edition.

Schroeder-Heister, P. (2018). Proof-Theoretic Semantics. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Spring 2018 edition.

Semântica de Reescrita para PEG Tipada

Daniel F. Pereira[1], **Elton M. Cardoso**[2], **Leonardo V. S. Reis**[1],
Regina S. M. A. de Paula[1], **Rodrigo G. Ribeiro**[2]

[1]Departamento de Ciência da Computação – Universidade Federal de Juiz de Fora (UFJF)
Juiz de Fora – MG – Brasil

[2]Prog. de Pós-Graduação em Ciência da Computação – Universidade Federal de Ouro Preto
Ouro Preto – MG – Brasil

Abstract. *In this paper we describe a type approach to verifying well-formed PEGs. The formalization guarantees that no PEG will get in infinite recursion with any input string. We implemented the formalization of the inference algorithm proposed by Cardoso et al. (2023) in PLT Redex following the rewriting semantic method.*

Resumo. *Este artigo descreve uma abordagem baseada em tipos para verificar a boa-formatura de PEGs. A formalização da abordagem garante que nenhum termo de PEG entre em laço infinito para qualquer entrada. Implementamos a formalização do algoritmo de inferência proposto por Cardoso et al. (2023) em PLT Redex seguindo o método de semântica de reescrita.*

1. Introdução

Uma *Parsing Expression Grammar* (PEG) é um formalismo que descreve analisadores sintáticos descendentes (Ford, 2004). Uma gramática PEG é uma quádrupla (V, Σ, R, e_s), em que V é um conjunto de não-terminais, Σ é um alfabeto, R é uma função de não-terminais para *parsing expressions* e e_s é a *parsing expression* inicial. Uma *parsing expression* é definida conforme a gramática abaixo:

$$e \ \rightarrow \epsilon \mid a \mid A \mid e_1 \ e_2 \mid e_1 \ / \ e_2 \mid e \star \mid ! \ e$$

A expressão ϵ denota uma cadeia vazia, $a \in \Sigma$ um terminal, $A \in V$ um não-terminal, $e_1 \ e_2$ uma sequência, $e_1 \ / \ e_2$ uma alternativa prioritária, $e\star$ uma repetição gulosa e $! \ e$ um predicado de negação.

Dada uma cadeia de caracteres (*string*) de entrada, uma PEG pode consumir um prefixo possivelmente vazio da entrada ou falhar. Considere, por exemplo, a PEG $ab*$ aplicada à entrada "`abbc`". Neste caso, o prefixo "`abb`" será consumido, pois o primeiro símbolo da entrada consome o primeiro elemento da sequência, a, e o segundo elemento da sequência, $b*$, consome uma sequência de zero ou mais ocorrências contíguas de b. A mesma PEG quando aplicada a entrada "`bbc`" resulta em falha, pois o primeiro elemento da sequência falha.

Uma questão central em PEGs é o conceito de completude, o qual determina se uma PEG tem sucesso ou falha no processamento de qualquer *string* de entrada. Dizemos que uma PEG é bem-formada se a mesma não entra em recursão infinita num processamento de uma entrada.

Um exemplo de uma expressão PEG que não é bem-formada é a expressão $a(!b)*$, a qual, em princípio, aceita *strings* formadas de um "a" seguida por zero ou mais ocorrências de caracteres que não sejam "b". No entanto, a expressão $!b$ resulta em sucesso sem consumir nenhum símbolo da entrada, e dada a natureza gulosa do operador $*$, temos como resultado um laço infinito no processo de reconhecimento da *string*. Como exemplo, a entrada "ac" faz com que o algoritmo de reconhecimento entre em um laço infinito. O mesmo problema pode ocorrer se a PEG contiver recursão à esquerda como na regra $A \leftarrow Aa/b$.

Ford (2004) mostrou que o problema de determinar se uma PEG é completa é indecidível, no entanto, ele definiu um critério ao qual assegura que toda PEG que a satisfaça é bem-formada. Ribeiro et al. (2019) argumentam que o critério do Ford pode não ser suficientemente claro e propõe uma abordagem alternativa baseada em tipos para determinar se uma PEG é bem-formada. Eles definem um tipo para uma *parsing expression* como uma tupla $\langle b, S \rangle$, em que b é um valor booleano que informa se a expressão pode aceitar a entrada vazia, ϵ, e S é um conjunto de variáveis que podem aparecer imediatamente à esquerda no corpo de uma regra. O conjunto S é chamado de *headset*. Uma PEG é considerada bem-tipada se todas as variáveis e a expressão inicial são bem-tipadas. Toda PEG bem-tipada tem a garantia de terminação para qualquer entrada, adicionalmente toda PEG bem-tipada também é bem-formada (Ribeiro et al., 2019). O sistema de tipos de PEG proposto originalmente requer que todos as variáveis sejam anotadas com seus respetivos tipos (Ribeiro et al., 2019), no entanto, essa requisição pode se mostrar impraticável para extensas especificações sintáticas. Para contornar este problema, um algoritmo de inferência foi desenvolvido para permitir que se obtenha os tipos das variáveis de forma automática (Cardoso et al., 2023).

Neste trabalho apresentamos a formalização utilizando uma semântica de rescrita (Kuan et al., 2007) em PLT Redex (Felleisen et al., 2009) do algoritmo de inferência de PEGs bem-formadas baseado em tipos proposto por Cardoso et al. (2023).

As seções a seguir apresentam a descrição da formalização da abordagem baseado em tipo de PEG, a avaliação feita a respeito da abordagem descrevendo os testes produzidos e uma conclusão a respeito da inferência de tipo e trabalhos futuros.

2. Formalização

A formalização proposta aqui contempla uma especificação de PEG, uma especificação da linguagem de restrições, incluindo a semântica para a redução das restrições, em PLT Redex (Felleisen et al., 2009). Abaixo apresentamos a especificação da sintaxe de PEG:

```
(define-language Peg
  (e natural      ; Terminal
     (/ e e)      ; Choice
     (• e e)      ; Sequence
     (* e)        ; Repetition
     (! e)        ; Not complement
     ϵ            ; Empty
     x)           ; Non-Terminal
  (x variable-not-otherwise-mentioned))
```

A linguagem de restrições, abaixo listada, descreve tipos básicos (τ), termos (t) e fórmulas (C). O tipo τ pode ser uma variável de tipo α; ou um tipo concreto $(b\ S)$. Foi

necessário complementar a linguagem de restrições com as operações sobre tipos, a saber $(\times\ \tau\ \tau)$, $(+\ \tau\ \tau)$, $(\star\ \tau)$ e $(!\ \tau)$ cujos nomes correspondem ao que está descrito em (Ribeiro et al., 2019). A operação $(clone\ \tau\ x)$ foi inserida para capturar as restrições referentes ao tipo de variável. Os termos t podem ser variáveis ou tipos e por fim, as fórmulas C, que podem ser a equivalência entre dois termos $(t\ \equiv\ t)$, a conjunção de duas fórmulas $(\wedge\ C\ C)$, ou uma associação de um tipo a variável, $(def\ x\ :\ \tau\ in\ C)$.

Os contextos ψ e φ contém, respectivamente, o ambiente de tipos de variáveis e a substituição. A cada passo uma equação é selecionada, resolvida por unificação e o resultado é composto com a substituição corrente. O processo é iterado até que todas as restrições tenham sido resolvidas.

O contexto de redução CEval permite que as regras semânticas de redução possam ser aplicadas a sub-termos da linguagem de expressões. A definição desse contexto deixa claro que cada contexto da aplicação das regras de redução pode ocorrer de qualquer lado do operador de conjunção, mas não em uma definição de tipo de uma variável.

```
(define-extended-language Typed-Peg Peg
  [τ α (b S) (× τ τ) (+ τ τ) (⋆ τ) (! τ) (clone τ x)]
  [t x τ]
  [C b (t ≡ t) (∧ C C) (def x : τ in C)]
  [CEval hole (∧ CEval C) (∧ C CEval)]
  [α (∨ natural)]
  [S (x ...)]
  [b #t #f]
  [x variable-not-otherwise-mentioned]
  [ψ ((x τ)...)]
  [φ ((α τ)...)])
```

A relação de redução em Redex, é definida por um conjunto de regras de redução na forma $(--> \ (\psi\ \varphi\ C_1)\ (\psi\ \varphi\ C_2))$, onde $(\psi\ \varphi\ C_1)$ é o de termo a ser reescrito, formado pelo contexto de tipos ψ, a substituição φ e uma fórmula C_1, e o termo $(\psi\ \varphi\ C_2)$ resultante da reescrita. Abaixo exemplificamos uma das regras referente a tipagem da repetição, que julga que a repetição de uma expressão que pode aceitar ϵ é mal-tipada.

```
(--> (ψ φ (in-hole CEval ( t ≡ (⋆ (#t S)))))
     (ψ φ #f)
     "r-3")
```

A regra tipa uma operação $\star$ cujo campo *nullable* não poderá ser falso, caso contrário, a PEG entrará em recursão infinita. Como queremos evitar essa situação, quando o tipo resultante da operação $\star$ de uma PEG anulável é falso, independente da lista de não-terminais, o resultado da redução também será falso.

3. Avaliação

A avaliação da semântica foi realizada usando um gerador de PEGs bem-formadas (Cardoso et al., 2022). Com a biblioteca de testes baseadas em propriedades *rackcheck* verificamos se os tipos inferidos pela nossa implementação é o mesmo obtido pela ferramenta usando o Z3 de Cardoso et al. (2023). A seguir, apresentamos o código em Racket para testar esta propriedade.

```
(define-property type-checks([peg  (gen:peg 3 5 2)])
```

```
(compare-types (get-type-of-typed-peg peg)
               (get-type-of-genConstraint peg)))
```

Exemplificando o que foi dito acima, seja o termo $((R\ (*\ 0)\ \oslash)\ \ (/\ R\ 5))$, em que a gramática contém a regra R associada a uma expressão de repetição, $(*\ 0)$ e a expressão a ser analisada é a alternativa, $(/\ R\ 5)$. A PEG é inserida na função para geração de restrições que gera a seguinte restrição:

```
(((R (v 4)))                                                    ψ
 ()                                                             φ
 (∧
  (∧ ((v 5) ≡ (#f ())) ((v 4) ≡ (⋆ (v 5))))
  (∧ (∧ (∧ (R ≡ (v 2)) ((v 1) ≡ (clone (v 2) R)))
           ((v 3) ≡ (#f ())))
     ((v 0) ≡ (+ (v 1) (v 3))))))                               C
```

A primeira lista se refere ao ambiente ψ o qual tem a lista de não-terminais da gramática associado ao seu respectivo tipo. A segunda lista se refere a ϕ que inicialmente é vazia. E a última lista é composta pelas restrições relativas à gramática e ao termo. O tipo $(v\ 0)$ indica o tipo relativo a 5, $(v\ 5)$ e $(v\ 4)$ se associam ao tipo de R. E a conjunção do tipo de 5 com a equivalência de R formam a restrição para a alternância.

Essa restrição é inserida na relação de redução composta pelas regras da linguagem Typed-Peg mostrada anteriormente e o resultado obtido é composto pelos ambientes ψ, ϕ e uma fórmula C que indica se o termo faz parte ou não da linguagem:

```
'((((R (#t ()))))                                               ψ
   (((v 0) (#t (R))) ((v 1) (#t (R))) ((v 2) (#t ())))          φ
   ((v 3) (#f ())) ((v 4) (#t ())) ((v 5) (#f ())))
   #t))                                                         C
```

No teste de propriedade acima, obtemos o resultado da mesma PEG para a biblioteca "typed-peg" (utilizando o Z3) e extraímos o tipo relativo aos não-terminais presentes na gramática. Comparamos este resultado com o ψ obtido acima e concluímos que eles são iguais, mostrando que o resultado de duas bibliotecas diferentes em relação a uma mesma PEG obtém a mesma saída.

4. Conclusão

O artigo apresenta uma abordagem sólida e fundamentada em tipos para verificar se PEGs são bem-formadas. Apresentamos uma formalização do algoritmo de inferência de tipos proposto por Cardoso et al. (2023) usando uma abordagem baseada na semântica de reescrita Kuan et al. (2007).

Usando uma semântica executável implementada em PLT Redex (Felleisen et al., 2009), executamos testes que nos fornecem evidências que a formalização está correta. Planejamos estender o conjunto de testes, usando as técnicas de testes baseados em propriedades, para aumentar a confiança na correção da formalização, analisando cobertura de código.

Referências

Cardoso, E. M., Paula, R. D., Pereira, D., Reis, L., and Ribeiro, R. G. (2023). Type-based termination analysis for parsing expression grammars. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*. ACM.

Cardoso, E. M., Pereira, D. F., De Paula, R. S. M. A., Reis, L. V. D. S., and Ribeiro, R. G. (2022). A type-directed algorithm to generate random well-formed parsing expression grammars. In *Proceedings of the XXVI Brazilian Symposium on Programming Languages*, SBLP '22, page 8–14, New York, NY, USA. Association for Computing Machinery.

Felleisen, M., Findler, R. B., and Flatt, M. (2009). *Semantics engineering with PLT Redex*. Mit Press.

Ford, B. (2004). Parsing expression grammars: A recognition-based syntactic foundation. In *Proceedings of the 31st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '04, page 111–122, New York, NY, USA. Association for Computing Machinery.

Kuan, G., MacQueen, D., and Findler, R. B. (2007). A rewriting semantics for type inference. In *Programming Languages and Systems*, pages 426–440. Springer Berlin Heidelberg.

Ribeiro, R., Reis, L. V. S., Feitosa, S., and Cardoso, E. M. (2019). Towards typed semantics for parsing expression grammars. In *Proceedings of the XXIII Brazilian Symposium on Programming Languages*. ACM.

Jogos Fechados de Convexidade em Árvores: complexidade e estratégias vencedoras

João Marcos Brito[1], Samuel N. Araújo[1,2], Raquel Folz[3], Rosiane de Freitas[3], Rudini Sampaio[1]

[1]Departamento de Computação, Universidade Federal do Ceará, Fortaleza, Brazil.

[2]Instituto Federal de Educação, Ciência e Tecnologia do Ceará, Crato, Brazil.

[3]Instituto de Computação, Universidade Federal do Amazonas, Manaus, Brazil.

jmbmaluno@alu.ufc.br, samuel.araujo@ifce.edu.br,

{raquel.folz,rosiane}@icomp.ufam.edu.br, rudini@dc.ufc.br

Abstract. *Accordingly to Duchet (1987), the first paper of convexity on general graphs is the 1981 paper "Convexity in graphs". One of its authors, Frank Harary, introduced in 1984 the first graph convexity games, on the geodesic convexity, investigated in a sequence of five papers that ended in 2003. In this paper, we define new convexity games, and obtain winning strategies and complexity results. Among them, we obtain winning strategies for trees from the Sprague-Grundy theory on impartial games.*

Resumo. *De acordo com Duchet (1987), o primeiro artigo de convexidade em grafos gerais é o artigo de 1981 intitulado "Convexity in graphs". Um dos seus autores, Frank Harary, introduziu em 1984 os primeiros jogos de convexidade em grafos, na convexidade geodésica, investigados em uma sequência de cinco artigos que terminou em 2003. Neste artigo, definimos novos jogos de convexidade de grafos e obtemos estratégias vencedoras e resultados de complexidade. Entre eles, obtemos estratégias vencedoras para árvores através da teoria de Sprague-Grundy em jogos imparciais.*

1. Introduction

Convexidade é um tema clássico, estudado em muitos ramos diferentes da matemática. O estudo de convexidades aplicadas a grafos começou recentemente, cerca de 50 anos atrás. O artigo de 1972 de Erdős et al. [Erdős et al. 1972] é um dos primeiros neste tópico, focado em torneios. De acordo com Duchet [Duchet 1987], o primeiro artigo em grafos gerais, publicado em inglês, é o artigo de 1981 entitulado "*Convexity in graphs*" de Frank Harary e Juhani Nieminem [Harary and Nieminem 1981]. Em 1984, Frank Harary introduziu os primeiros jogos de convexidade de grafos em seu resumo "*Convexity in graphs: achievement and avoidance games*" [Harary 1984]. Esta linha de pesquisa sobre jogos de convexidade em grafos terminou em 2003 após uma sequência de cinco artigos [Harary 1984, Buckley and Harary 1985b, Buckley and Harary 1985a, Nečásková 1988, Haynes et al. 2003], todos eles focados na convexidade geodésica. Neste artigo, continuamos esta linha de pesquisa definindo novos jogos naturais de convexidade e obtendo resultados sobre os antigos e novos jogos para algumas convexidades clássicas de grafos. Para explicá-los precisamos de algumas definições básicas.

Dado um grafo G e um conjunto $S \subseteq V(G)$, a função de intervalo geodésico $\mathrm{I}_g(S)$ resulta em S e em todo vértice que pertence a algum caminho mínimo entre dois vértices de S. Dizemos que S é *convexo em uma convexidade geodésica*[Everett and Seidman 1985, Farber and Jamison 1987] se $\mathrm{I}_g(S) = S$. O fecho convexo geodésico de S é o menor conjunto convexo $\mathrm{hull}_g(S)$ que contém S. Sabe-se que o $\mathrm{hull}_g(S)$ pode ser obtido aplicando $\mathrm{I}_g(\cdot)$ exaustivamente em S até obter um conjunto convexo.

Dentre os jogos definidos por Haray[Buckley and Harary 1985a, Buckley and Harary 1985b, Harary 1984] para os jogos de convexidade geodésico, temos o jogo de intervalo geodésico, o jogo de envoltória geodésico, o jogo de intervalo geodésico fechado e o jogo de envoltória geodésico fechado, definidos abaixo.

Definição 1.1. *Seja G um grafo. Nos jogos definidos abaixo, seja L o cojunto dos vértices rotúlados inicialmente vazio e as definições dos conjuntos $f_1(L)$ e $f_2(L)$ dependem do jogo. Dois jogadores (Alice e Bob, começando por Alice) rotulam alternadamente um vértice não rotulado v que não está em $f_1(L)$. O jogo acaba quando $f_2(L) = V(G)$.*

- *No jogo de envoltória geodésico HG$_g$: $f_1(L) = L$ e $f_2(L) = hull_g(L)$.*
- *No jogo de intervalo geodésico IG$_g$: $f_1(L) = L$ e $f_2(L) = I_g(L)$.*
- *No jogo de envoltória geodésico fechado CHG$_g$: $f_1(L) = f_2(L) = hull_g(L)$.*
- *No jogo de intervalo geodésico fechado IG$_g$: $f_1(L) = f_2(L) = I_g(L)$.*

Na variante normal, o primeiro jogador que não possui jogadas possíveis perde. Na variante misère, o primeiro jogador que não possue jogadas possíveis é o vencedor. Pelo teorema clássico de Zermelo-von Neumann [Zermelo 1913], um dos dois jogadores possui uma estretégia vencedora em cada um desses jogos, desde que os jogos sejam finitos, de informação perfeita e sem empates. Logo, o problema de decisão desses jogos são se Alice possui uma estratégia vencedora.

Nesse artigo, obtemos um algoritmo em tempo polinomial baseado na teoria de Sprague-Grundy nos jogos fechados CIG$_g$ e CHC$_g$ em árvores.

2. Jogos imparciais e a Teoria de Sprague-Grundy

Dizemos que um jogo de combinatória na variante normal ou misère é imparcial se o conjunto dos movimentos possíveis para qualquer posição dada (ou configuração) é a mesma para ambos os jogadores. É fácil de ver que os jogos da Definição 1.1 são jogos imparciais.

Nim é um dos jogos imparciais mais importantes em que a cada turno ambos os jogadores removem objetos de montes disjuntos até não haver mais objetos. Em cada turno, um jogador escolhe um monte não vazio e remove uma quantidade positiva de objetos desse monte. Uma instância de Nim é dada pela sequência $\Phi = (h_1, h_2, ..., h_k)$, onde k é o número de montes e $h_i \geq 1$ é o tamanho do iésimo monte. Nim possui um papel fundamental na teoria de Sprague-Grundy e foi matematicamente resolvido [Bouton 1901] por Bouton em 1901 para qualquer instância Φ. Em outras palavras, o problema de decisão para decidir qual dos jogados tem uma estratégia vencedora possui tempo polinomal em ambas variantes do Nim. Ela está relacionadas com o **nim-sum**$(\Phi) = h_1 \oplus h_2 \oplus ... \oplus h_k$, onde $\oplus$ é a operação bitwise xor. Alice tem uma estratégia vencedora na variante normal de Nim se e somente se o **nim-sum**$(\Phi) > 0$. Assim como, Alice tem uma estratégia

vencedora na variante misère de Nim se e somente se **nim-sum**$(\Phi) > 0$ e há pelo menos um monte com mais de um objeto ou **nim-sum**$(\Phi) = 0$ e todos os montes possuem exatamente um objeto.

A teoria do Sprague-Grundy, desenvolvida independentemente por R.P.Sprague [Sprague 1936] e P. M. Grundy [Grundy 1939] em 1930, diz que é possível associar um número (chamado de *nimber*) a toda posição finita em um jogo imparcial, associando a um jogo de Nim de apenas um monte. A expecitativa com essa associação é que após um movimento em uma posição com nimber $h > 0$ de um jogo imparcial, é possível obter uma posição com qualquer nimber em $\{0, 1, ..., h-1\}$. Além disso, o nimber de uma posição pode ser calculado pelo valor mex$\{h_1, ..., h_k\}$, onde $\{h_1, ..., h_k\}$ contem todos os nimbers de uma posição obtida através de um movimento e o mínimo excludente mex é o mínimo inteiro não negativo que não pertence ao conjunto. Por fim, uma posição que pode ser obtida por uma união de k posições disjuntas com nimbers $h_1, ..., h_k$ tem nimber $h_1 \oplus ... \oplus h_k$.

3. Jogos de convexidade fechados em árvores

Diante do exposto, vejamos primeiro uma versão simplificada do jogo de invervalo fechado CIG$_g$ definido abaixo.

Definição 2.1. *Dado um grafo G e um vértice v de G, seja SCIG$_g$ de uma instância (G, v) um CIG$_g$ no grafo G com exceção de que v já está rotulado no início do jogo. Isso significa que ao invés de estar vazio no começo do jogo, o conjunto L de vértices rotulados é tal que $L = \{v\}$. Análogamente, definimos as versões simplificadas para os outros jogos da Definição 1.1.*

Teorema 2.2. *O problema de decisão para as variantes normal e misère do SCIG$_g$ e SCHG$_g$ é resolvido em tempo polinomial em árvores.*

Demonstração. Seja T uma árvore e seja v um vértice de T. Em um jogo simplificado com instância (T, v) em uma árvore, vamos considerar que sempre que um novo vértice w é rotulado, os vértice no caminho entre v e w também serão rotulados no jogo. Isso acontece pois só há um caminho entre um par de vértices em uma árvore e esse caminho é mínimo.

Se v não tem vizinhos, então o nimber de (T, v) é 0 e Alice perde na variante normal e ganha na variante misère. Então, seja u um vizinho de v. Dizemos que (T, v, u) é um jogo simplificado da instância $(T_{v,u}, v)$ onde $T_{v,u}$ é uma subárvore de T contendo v obtida através da remoção de todas as arestas que incidem em v exceto vu. Note que v é uma folha de $T_{v,u}$.

Primeiro resolvemos o jogo para instância (T, v, u). Seja $u_1, ..., u_k$ os vizinhos de u distintos de v. Assuma que h_1 de (T, u, u_i) em SCIG$_g$ é conhecido. Se Alice rotular u, então os jogos de $(T, u, u_1), ..., (T, u, u_k)$ são independentes entre si, isto é, um movimento em uma instância não afeta outra. Pela teoria de Sprague-Grundy, a posição resultande tem nimber $h_1 \oplus ... \oplus h_k$.

Se Alice rotular um vértice w_1 em (T, u, u_i), que é distindo de u, então u também será rotulado e novamente o jogo nas subárvores serão independetes um do outro. Seja h'_i o nimber da posição resultante de $T(u, u_i)$. Então, pela teoria de Sprague-Grundy, a

posição resultante de (T, v, u) após o movimento em w_1 tem nimber $h_1 \oplus ... \oplus h'_i \oplus ... \oplus h_k$. Além disso, h'_i pode ter qualquer valor em $\{0, 1, ..., h_i - 1\}$.

Seja N o conjunto de todos os nimbers de $h'_1 \oplus ... \oplus h'_k$ onde $h'_i = h_i$ para todo $i = 1, ..., k$ exceto que $h'_i \in \{0, ..., h_i - 1\}$. Portanto, pela teoria de Sprague-Grundy, o nimber de (T, v, u) é exatamente mex(N), desde que N contenha todas as nimbers possíveis após um movimento em (T, v, u).

Esses argumentos conduzem a um algoritmo recursivo de tempo polinomial para cálcular o nimber de (T, v, u) de uma árvore T enraizada em uma folha v.

Agora, condisere o jogo em toda árvore T e seja $v_1, ..., v_k$ os vizinhos de v para $k \geq 2$. Pelos argumentos supracitados, é possível determinar o nimmber l_i de (T, v, v_i) em SCIG$_g$. Então, como antes, temos que os jogos nas subárvores são independentes e o nimber de (T, v) é exatamente $h = l_1 \oplus ... \oplus l_k$. Logo, conseguimos calcular o nimber de (T, v) de uma árvore T enraizada em v.

Por fim, fazendo uma associação direta ao jogo Nim, temos que Alice possui uma estratégia vencedora na variante normal se e somente se $h > 0$. Além disso, Alice possui uma estretégia vencedora na variante misère se e somente se $h > 0$ e há pelo menos uma subárvore com mais do quê um vértice não rotulado, ou $h = 0$ e todas as subárvores possuem exatamente um vértice não rotulado. □

Veja o exemplo da árvore T da Figura 1, assuma que Alice rotulou c no seu primeiro movimento no CIG$_g$, sabemos o valor da posição (T, c) pelo SCIG$_g$ onde Bob é o primeiro a jogar. Seja T' obtido da remoção de x_1. Vamos determinar o nimber de (T', c). Se Bob rotular v, então temos dois "montes"de tamanho $(4, 4)$, onde o nimber é $4 \oplus 4 = 0$. Se Bob rotular outro vértice de T', temos um "monte"de tamanho $k \leq 3$ e outro "monte"de tamanho 4. Então o nimber de (T', c) é mex$\{4 \oplus 4, 4 \oplus 3, 4 \oplus 2, 4 \oplus 1, 4 \oplus 0\} = 1$. Por fim, com o vértice x_1, o nimber de (T, c) é $1 \oplus 1 = 0$, que é uma posição vencedora nas variantes normal e misère para os segundos jogadores (que é Alice nesse exemplo).

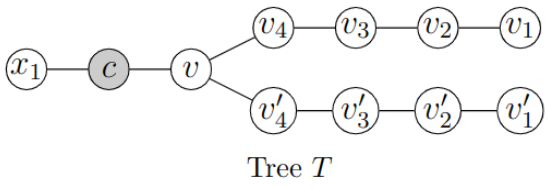


Figura 1. Jogo convexo em árvore

Corolário 2.3. *Os problemas de decisão relacionados às variantes normal e misère do jogo de intervalo fechado CIG$_g$ e do jogo de envoltória fechado HIG$_g$ são resolvidos em tempo polinomial.*

Demonstração. Seja T uma árvore. Alice possui uma estretégia vencedora em CIG$_g$ em T se e somente se há um vértice v tal que o segundo jogador do SCIG$_g$ em (T, v) possui uma estratégia vencedora. Pelo Teorema 2.2, concluímos que esse problema de decisão é resolvido em tempo polinomial. Além disso, em uma árvore, o jogo de envoltória fechado é equivalente ao jogo de intervalo. □

Referências

Bouton, C. L. (1901). Nim, a game with a complete mathematical theory. *Annals of Mathematics*, 3(1/4):35–39.

Buckley, F. and Harary, F. (1985a). Closed geodetic games for graphs. *Congressus Numerantium*, 47:131–138. Proc. of 16th Southeastern Conf. on Combinatorics, Graph Theory and Computing.

Buckley, F. and Harary, F. (1985b). Geodetic games for graphs. *Quaestiones Mathematicae*, 8:321–334.

Duchet, P. (1987). Convexity in combinatorial structures. In *Proceedings of the 14th Winter School on Abstract Analysis*, pages 261–293. Circolo Matematico di Palermo. Proceedings of the 14th Winter School on Abstract Analysis.

Erdős, P., Fried, E., Hajnal, A., and Milner, E. C. (1972). Some remarks on simple tournaments. *Algebra universalis*, 2:238–245.

Everett, M. G. and Seidman, S. B. (1985). The hull number of a graph. *Discrete Mathematics*, 57(3):217–223.

Farber, M. and Jamison, R. E. (1987). On local convexity in graphs. *Discrete Mathematics*, 66(3):231–247.

Grundy, P. M. (1939). Mathematics and games. *Eureka*, 2:6–8.

Harary, F. (1984). Convexity in graphs: Achievement and avoidance games. In Rosenfeld, M. and Zaks, J., editors, *Convexity and Graph Theory*, volume 87 of *North-Holland Mathematics Studies*, page 323. North-Holland.

Harary, F. and Nieminem, J. (1981). Convexity in graphs. *Journal of Differential Geometry*, 16(1):185–190.

Haynes, T. W., Henning, M. A., and Tiller, C. (2003). Geodetic achievement and avoidance games for graphs. *Quaestiones Mathematicae*, 26:389–397.

Nečásková, M. (1988). A note on the achievement geodetic games. *Quaestiones Mathematicae*, 12:115–119.

Sprague, R. (1936). Über mathematische Kampfspiele. *Tôhoku Mathematical Journal*, 41:438–444.

Zermelo, E. (1913). Über eine Anwendung der Mengenlehre auf die Theorie des Schachspiels. In *Proc. 5th Int. Congress of Mathematicians*, pages 501–504.

Algoritmo de shor: uma introdução à criptografia quântica

William K. S. Silva, Maigan S. da S. Alcântara

Departamento de computação – Universidade Federal Rural de Pernambuco (UFRPE)
Recife – PE – Brazil

williamkauasoaresdasilva@gmail.com, maigan.salcantara@ufrpe.br

Abstract. *The propagation of computational artifacts in modern life caused an exponential advance in data sharing, which is immersed in a continuous exchange of information between governments, companies and users. This caused an increasing concern in cybersecurity, which studies methods of information protection, cryptography, which has proven to be extremely important in our society, since the information contained in the exchanged bits is becoming more personal and important every day. In this sense, this paper describes the functioning of Shor's algorithm and the computational and mathematical relations involved.*

Resumo. *A difusão de artefatos computacionais na vida moderna trouxe um avanço exponencial no compartilhamento de dados, que está imerso numa troca contínua de informações entre governos, empresas e pessoas físicas. Isso trouxe uma preocupação crescente na área da chamada cibersegurança, a qual lida com métodos de proteção à informação, a criptografia, que vem se demonstrando importantíssima na nossa sociedade, onde o teor dos bits trocados vem se tornando cada dia mais pessoal e importante. Nesse sentido, este artigo descreve o funcionamento do algoritmo de Shor e as relações computacionais e matemáticas envolvidas.*

1. Introdução

A criptografia vem se demonstrando importantíssima na nossa sociedade, onde o teor dos bits trocados vem se tornando cada dia mais pessoal e importante [BURNETT, 2002]. Com intuito de fortalecer a privacidade dos usuários, uma série de estudos foram realizados ao longo dos anos, desenvolvendo os mais variados protocolos de segurança digital, os quais fazem parte ativamente de nosso dia a dia [KESSLER, 2003; MENEZES et al, 2018; GOLDREICH, 2004].

Dentre as tecnologias de criptografia mais utilizadas está a RSA (Rivest-Shamir-Adleman), desenvolvida em 1978 por R. L. Rivest, A. Shamir e L. Adleman, que atuavam como pesquisadores do Massachusetts Institute of Technology (MIT), método esse que nos permite codificar uma mensagem utilizando o produto de dois grandes fatores primos, as chaves privadas, onde, para a decodificação é necessário conhecer ambos os componentes, sabendo apenas do produto entre eles, a chave pública, ou seja, um problema clássico de fatoração de grandes números [GARY, 1976; PETER SHOR, 1996]. O que torna esse protocolo de segurança tão eficaz em seu trabalho, é o fato de que nenhum algoritmo não quântico conhecido é capaz de quebrá-lo em tempo útil, já que as abordagens clássicas se baseiam na tentativa e erro, o que diminui muito sua eficiência.

2. Acelerando processos de fatoração utilizando das relações da teoria dos números

Ao longo dos anos foram feitos vários esforços para quebrar o método RSA, entre esses está o uso da teoria dos números, de onde se originam dois recursos importantes, o algoritmo de Euclides e o algoritmo descrito nos artigos “Riemann's hypothesis and tests for primality” [GARY, 1976] e “Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer” [PETER SHOR, 1996] que explora a relação da álgebra de grupos com a fatoração.

3. Relação de divisibilidade em grupos cíclicos

Seja k pertencente aos inteiros, tal que, k < N e o mdc(k,N) = 1, isso implica que k é gerador de um grupo finito sobre a multiplicação módulo N, denotado por (Z/NZ)×, Como o grupo é finito e k é gerador deve existir uma ordem P tal que $k^P \equiv 1 \bmod N$, o'que significa que N | (k^P - 1), podemos então fatorar essa parcela como:

$$(k^P - 1) = (k^{\frac{p}{2}} - 1)\,(k^{\frac{p}{2}} + 1),$$

o que mantém a relação de divisibilidade, N | ($k^{\frac{p}{2}}$ - 1) ($k^{\frac{p}{2}}$ + 1), caso N não divida nenhuma das parcelas, isso significa que o mesmo compartilha fatores primos com pelo menos uma delas, o'que nos possibilita achar esses fatores através do algoritmo de euclides [COUTINHO, 1997].

4. O algoritmo clássico

Para o sucesso do algoritmo descrito acima existe uma série de condições necessárias,como por exemplo a paridade da ordem P, pois caso a ordem seja ímpar, $\frac{P}{2}$ não é inteiro, logo, k^P pode retornar um número não inteiro, o que não satisfaz nosso algoritmo. Uma importante relação diz que, sejam x e y inteiros positivos e $k^y \equiv$ x mod N então $k^{y+p} \equiv$ x mod N, pois P representa também o tamanho do ciclo ou período do grupo e após somar P ao expoente o resultado se repete.

Considerando os requerimentos, o algoritmo através de tentativa e erro seleciona valores inteiros de k tal que, k < N e o mdc(k,N) = 1, e obtém sua ordem P, com o intuito de achar co fatores primos com N; Algoritmo esse que pode ser executado em computadores clássicos, porém, os algoritmos que tem como objetivo achar o valor de P classicamente, tem altíssimas complexidades assintóticas, sendo esta uma questão NP (Não-Polinomial).

5. O algoritmo polinomial de Shor

Desenvolvido em 1994 pelo matemático americano Peter Williston Shor, visando resolver problemas relacionados com o período de funções, este é um algoritmo quântico com o objetivo de achar o valor P, que como já discutido representa a ordem de

um elemento gerador num grupo multiplicativo, em sumo a versão não clássica funciona basicamente como seu algoritmo antecessor, porém utilizando se de alguns recursos do mundo quântico para acelerar sua velocidade exponencialmente [PETER SHOR, 1996].

5.1. Superposição quântica

Diferentemente dos computadores clássicos que trabalham com bits representados por 0 e 1, a linguagem binária, suas versões quânticas trabalham com os Qubits, quantum bits, que existem em um estado físico chamado superposição, o qual os situa entre 0 e 1, e quando medidos colapsam nos valores clássicos a partir de uma certa porcentagem, estes são descritos como dois valores complexos a e b, onde a seguinte relação $|a|^2 + |b|^2 = 1$ é respeitada, e ambos representam as chances de numa medição serem obtidos os valores binários, onde $|a|^2$= P(Qubit = 1) e $|b|^2$= P (Qubit = 0) [NIELSEN, 2010].

Além disso, os computadores quânticos podem usar da superposição para realizar múltiplas tarefas de uma vez, colocando as entradas de um algoritmo em um estado que representa todas elas ao mesmo tempo, e então realizando as operações no mesmo, porém, ao realizar a medição para observar os resultados a superposição colapsa e a saída será uma das entradas iniciais após as operações, onde cada entrada possui uma probabilidade em forma de porcentagem de ser observada, e a soma das chances de todas as prováveis saídas é igual a um [YANOFSKY, 2008].

Outra característica importante da superposição é que ao observar um elemento que após as operações compartilha o resultado com outras entradas não observadas, obtemos uma nova superposição que só possui elementos com esse mesmo resultado, basicamente como um filtro que seleciona apenas alguns resultados específicos, característica essa que acontece naturalmente por causa das propriedades da física quântica [NIELSEN, 2010].

5.2 Transformada de Fourier quântica (TFQ)

A transformada de fourier clássica é fortemente usada em várias áreas da tecnologia, com o objetivo de representar um sinal em função das frequências que o compõem, em sua versão quântica esse método também tem o objetivo de retornar frequências, por exemplo, para uma entrada de apenas um número real positivo x, o algoritmo retorna uma superposição quântica de todos os outros números com diferentes probabilidades de serem observados entre eles, que possui uma frequência de x, e assim entradas maiores retornam maiores frequências [CAMPS et al, 2021].

Outra importante propriedade da TFQ acontece quando a entrada é uma superposição de estados quânticos, já que a soma das TFQs dos elementos é igual a TFQ de toda a superposição, e como este método tem frequências como saída, a soma das frequências resultantes dos elementos pode causar interferência destrutiva ou construtiva, em razão disso a TFQ de uma superposição de números separados por uma constante real positiva P, seu período, retorna uma frequência de $\frac{1}{P}$.

5.3 Construindo o algoritmo

O algoritmo de shor inicialmente coloca todos os possíveis valores y para a ordem P em uma superposição, e então aplica a função

$$F(y) = k^{y} \bmod N,$$

seja N o número que é desejado fatorar, e k pertencente aos inteiros, tal que, k < N e o mdc(k,N) = 1; Em seguida observando um dos elementos e assim obtendo uma superposição de todos os valores de y que possuem o mesmo valor na função, separados entre si por P onde

$$y_i + P = y_{i+1},$$

e assim por último usando essa superposição como entrada numa transformada de Fourier quântica, que tem como resultado a frequência de $\frac{1}{P}$, sendo P o período desejado inicialmente. Possuindo o valor de P é feita a checagem se esse valor é par, caso não seja, o algoritmo re-começa com um novo valor de k, caso verdadeiro esse valor é usado na relação

$$N \mid (k^{\frac{p}{2}} - 1)\,(k^{\frac{p}{2}} + 1),$$

uma nova checagem é feita com intuito de confirmar que ambas as parcelas individualmente não são divisíveis por N, mais uma vez caso falso o algoritmo re-começa com um novo valor de k, caso verdadeiro isso significa que $(k^{\frac{p}{2}} - 1)$ e $(k^{\frac{p}{2}} + 1)$ compartilham ambos ao menos um fator primo com N , então, é utilizado o algoritmo de euclides que facilmente pode determinar quais os fatores compartilhados, assim achando as chaves privadas e quebrando a criptografia RSA com uma complexidade de

$$O(log(N)^2 \log(\log(N))\log(\log(\log(N)))).$$

Referências

- BURNETT, S. (2002). *Criptografia e segurança: o guia oficial RSA*. Gulf Professional Publishing.

- KESSLER, Gary C. An overview of cryptography. 2003.

- MENEZES, Alfred J.; VAN OORSCHOT, Paul C.; VANSTONE, Scott A. Handbook of applied cryptography. CRC press, 2018.

- GOLDREICH, Oded. Foundations of Cryptography, Volume 2. Cambridge: Cambridge university press, 2004.

- CAMPS, Daan; VAN BEEUMEN, Roel; YANG, Chao. Quantum Fourier transform revisited. *Numerical Linear Algebra with Applications,* v. 28, n. 1, p. e2331, 2021.

- YANOFSKY, Noson S.; MANNUCCI, Mirco A. *Quantum computing for computer scientists.* Cambridge University Press, 2008.

- NIELSEN, Michael A.; CHUANG, Isaac L. *Quantum computation and quantum information.* Cambridge university press, 2010.

- COUTINHO, Severino Colier. Números inteiros e criptografia RSA. IMPA, 1997.

Formal Analysis of Multi-Robot Patrolling Metrics

Pablo A. Sampaio[1], Rodrigo N. P. M. de Souza[1], Geber Ramalho[2], Patrícia Tedesco[2]

[1]Departamento de Computação – Universidade Federal Rural de Pernambuco (UFRPE)
Recife – PE – Brazil

[2]Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Recife – PE – Brazil.

{pablo.sampaio,rodrigo.npmsouza}@ufrpe.br, {glr,pcart}@cin.ufpe.br

Abstract. *This paper explores metrics for Timed Multi-Agent Patrolling (TMAP), a non-adversarial multi-robot patrolling problem. We define TMAP metrics, establish formal correlations, and link them to real-world patrolling task requirements. Our analysis may guide metric selection for specific applications.*

Resumo. *Este artigo explora métricas para o Patrulhamento Multiagente Temporal (TMAP), um problema de patrulhamento multirrobô não adversarial. Definimos métricas da TMAP, estabelecemos correlações formais e as associamos a requisitos de tarefas reais de patrulhamento. Nossa análise pode orientar a escolha de métricas para aplicações específicas.*

1. Introduction

The multi-agent or multi-robot patrolling problems focus on coordinating robot movements to repeatedly visit an environment. They have potential utility in diverse tasks, such as protecting property fences, maintaining distributed devices, and cleaning. However, the formal definitions of these problems differ in several aspects (e.g., the environment, the capabilities of the agents, the presence or absence of an adversary). In non-adversarial patrolling, the primary source of incompatibility lies in how the patrolling team's performance is evaluated, with different works using different metrics.

However, metrics are often not compared, and the choice of one metric over another is not always justified. This study, based on a doctoral thesis [5], reflects an effort to create a common framework for defining and comparing various metrics. We propose a meta-definition called Timed Multi-agent Patrolling (TMAP) to encompass several variants of patrolling. Within the TMAP framework, we redefine seven metrics from existing literature and introduce three new ones. We also prove results that establish relationships between the metrics based on the optimal or satisfactory behavior that the metrics impose on the team of agents. Finally, we categorize the metrics based on specific real-world requirements, serving as a guide for choosing the most suitable metric for a given application.

2. TMAP Definition

A TMAP problem is a sequential decision-making task where the objective is to maneuver agents (robots) within an environment to repeatedly visit a finite set of predefined *points of interest* (or nodes) over a specified duration T, with the aim of *optimizing* a collective performance metric M computed using the timestamps of the visits to each interest point. We also consider decision versions of TMAP problems, where the objective is to ensure that M satisfies a given bound k. This definition of a TMAP problem is generic enough to encompass many of the contrasting definitions of multi-robot patrolling problems of the literature. Notably, it enables us to redefine metrics from the literature on a common foundation.

3. TMAP Metrics

In TMAP, a metric M can be any function calculated from the *patrolling trajectory* of the agents (robots), defined as r sequences, each corresponding to a node x. Each sequence $(v_{x,k}) = (v_{x,1}, v_{x,2}, ..., v_{x,visits(x)})$ contains, in ascending order, the timestamps of visits to a given node x. The number of visits to node x is denoted by $visits(x)$. In the remaining of this section, we redefine patrolling metrics from literature and define new ones based on these basic structures. Due to space constraints, our focus was primarily on the essential metrics, requiring us to omit many details.

Three significant metrics from literature [2] rely on visit frequencies associated with each node. The frequency of visits to a single node x can be expressed as $freq(x) = visits(x) / T$. Consequently, the metrics of *minimum frequency* (F_{min}), *average frequency* (F_{avg}), and *standard deviation of frequencies* (F_{stddev}) found in the literature can be defined straightforwardly, e.g. F_{avg} is the average of all $freq(x)$ values for every node x.

Another set of metrics is defined in terms of intervals between visits at each node. The metrics in this category utilize all intervals from all nodes, remaining agnostic to the nodes (i.e., intervals affect the calculation in the same way independently of the node where they occurred). In this manner, the metrics *maximum interval* (I_{max}), *average interval* (I_{avg}), and *standard deviation of the intervals* (I_{stddev}) are defined as their names indicate. The *quadratic mean of intervals* (I_{qmean}) is calculated by squaring all intervals, computing their mean, and then taking the square root of the result. In this study, we propose a generalization of this approach – the metric $I_{gmean-<e>}$, which represents the *generalized mean of intervals* for arbitrary integer values e. It is calculated by elevating all intervals to the power of e , then averaging these values, then taking the e-th root of the result. Note that $I_{gmean-1}$ corresponds to I_{avg}, and $I_{gmean-2}$ precisely aligns with the *quadratic mean*. To the best of our knowledge, at least I_{stddev}, I_{qmean} and $I_{gmean-<e>}$ were first proposed in our original research work [5].

Two of the most widely used metrics [1][3][4] are defined in terms of the *idleness* of a node, which refers to the time elapsed since the last visit to that node (e.g., an idleness of 4 indicates that x was last visited 4 units of time ago). For a specific time discretization in turns, the metrics in this category are defined based on sequences comprising all idleness measurements across all turns for all nodes. We can represent these consecutive measurements, for any given node x, as a sequence $(o_{x,k})$ with T terms. The metrics of *maximum (or worst) idleness* (O_{max}) and *average (or global) idleness*

(O_{avg}) are calculated from all the values from the described sequences, considering all nodes in all turns.

4. Formal Framework

To establish relationships among the metrics, we introduce three formal concepts in this section, assuming that all metrics are framed as minimization objectives. Metrics that are naturally expected to be maximized appear with a -1 exponent.

Firstly, we establish an order relation between patrolling trajectories, denoted as the *performance order*. In a specific context of TMAP Φ (encompassing the environment, initial agent positions, etc.), we use $X \trianglelefteq_{\mathbf{M},\Phi} Y$ to represent that a patrolling trajectory X performs as well as or better than another patrolling trajectory Y in context Φ, as evaluated by metric M. Now, we can establish an equivalence relation among the metrics. Two metrics, M and N, are considered *rank-equivalent* iff they establish identical performance orders in any given context. We denote this equivalence as $M \equiv N$. In simpler terms, $M \equiv N$ iff both metrics consistently rank the solutions in the same manner.

We also define that *M has positive influence on N* iff, in any context, there exists a function *f* such that: $M \leq f(k)$ implies $N \leq k$. A *positive influence* (which is also an order relation) may reflect, in general, a weaker relationship than *rank-equivalence*. But it indicates that a kind of reduction exists between the problems, in which a solution to a *decision version* of TMAP with bound *k* for metric *N* can be obtained by solving the corresponding problem with bound *f(k)* for metric *M*.

5. Formal Results and Analysis

Using the definitions given before, we proved some results that establish relationships between the metrics. In this section, we present these results and use them to try to group the metrics by semi-formal *performance requirements*. Each subsection presents one of these requirements, the metrics and the corresponding results.

5.1. Maximize Overall Visits

Optimizing metrics within this group results in maximizing the number of visits to all nodes. Assuming that N_{visits} represents the total number of visits, Theorem 1 also identify I_{avg} and F_{avg} as metrics equally suitable for modeling this requirement. For decision versions of TMAP problems, I_{max} and F_{min} can also be used as substitutes, as shown in Theorem 2.

Theorem 1: $F_{avg}^{-1} \equiv I_{avg} \equiv N_{visits}^{-1}$.

Theorem 2: I_{max} and F_{min} have positive influence in N_{visits}.

5.2. Worst-case Spatial Visitation

The metrics in this group can serve to validate specific performance aspects related to the least visited node. F_{min} is a metric that naturally captures this requirement. Theorem 4 highlight that I_{max} can be effectively employed for decision versions.

Theorem 4: I_{max} has positive influence in F_{min}^{-1}.

5.3. Worst-case Interval

In this case, the team must reduce the maximum time between visits. The appropriate metrics are O_{max} and I_{max}. Note that they are not just rank-equivalent, but identical.

Theorem 5: $I_{max} = O_{max}$.

5.4. Spatially Uniform Visitation

In some applications it may be required that the team must visit nodes as equally as possible. F_{stddev} is an appropriate metric for this requirement. The following result shows that the ideally optimal case for $I_{stdddev}$ is also the ideal optimal for F_{stddev}, suggesting some relevance for I_{stddev} for this requirement.

Theorem 6: If $I_{stddev} = 0$ *then* $F_{stddev} = 0$.

5.5. Temporal Regularity

This addresses applications needing nodes to be visited at regular intervals. The metric that naturally captures this idea is I_{stddev}.

5.6. Visitation-Regularity Balance

This requirement represents that the team should aim for maximum visits while ensuring these visits are evenly distributed over time. It can be viewed as a general requirement when application needs aren't specific enough to fit previously defined requirements. A metric representing this requirement should strike a balance between two other requirements: 'maximize overall visits' and 'temporal regularity'. Theorem 9 highlights I_{qmean} as a suitable metric since it is essentially the combination of metrics from the two aforementioned requirements. Theorem 8 establishes O_{avg} as somewhat similar to I_{qmean} but with less emphasis given to I_{avg}. We remark that, in applications where each agent makes precisely one visit per unit of time (as in many references), I_{avg} is constant and so I_{qmean} and O_{avg} are rank-equivalent.

Theorem 7: $(I_{qmean})^2 = I_{avg}^{\ 2} + I_{stddev}^{\ 2}$

Theorem 8: $O_{avg} \equiv I_{qmean}^{\ 2}/I_{avg}$.

6. Conclusion

TMAP as a formal framework to redefine and analyze various versions of patrolling problems and their diverse metrics. We anticipate that this framework will be used or expanded upon to advance knowledge in the field, with practical implications. As an initial step in this direction, the formal results presented here, combined with our approach of categorizing metrics based on performance criteria, are expected to offer valuable guidance to practitioners in choosing appropriate metrics for real-world applications.

References

[1] Chevaleyre, Y. (2004). Theoretical analysis of the multi-agent patrolling problem. In Proceedings. IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2004.(IAT 2004). (pp. 302-308). IEEE.

[2] Elmaliach, Y., Agmon, N., & Kaminka, G. A. (2009). Multi-robot area patrol under frequency constraints. Annals of Mathematics and Artificial Intelligence, 57, 293-320.

[3] Elor, Y., & Bruckstein, A. M. (2010). Autonomous multi-agent cycle based patrolling. In International Conference on Swarm Intelligence (pp. 119-130). Springer.

[4] Machado, A., Ramalho, G., Zucker, J. D., & Drogoul, A. (2002). Multi-agent patrolling: An empirical analysis of alternative architectures. In International workshop on multi-agent systems and agent-based simulation (pp. 155-170). Springer.

[5] Sampaio, P. (2013). Patrulha Temporal: Taxonomia, Métricas e Novas Soluções. Tese de Doutorado [Doctorate Thesis in portuguese]. Centro de Informática (CIn), UFPE.

[6] Sampaio, P., Ramalho, G., & Tedesco, P. (2010). The gravitational strategy for the timed patrolling. In 2010 22nd IEEE International Conference on Tools with Artificial Intelligence (Vol. 1, pp. 113-120). IEEE.

[7] Santana, H., Ramalho, G., Corruble, V., & Ratitch, B. (2004, July). Multi-agent patrolling with reinforcement learning. In Autonomous Agents and Multiagent Systems, International Joint Conference on (Vol. 4, pp. 1122-1129). IEEE Computer Society.

Acknowledgements

Research supported by FACEPE, project APQ-0882-1.03/14.

Uma prova elementar da caracterização dos $\mathbb{N}$-autômatos limitados

Rodrigo de Souza[1], Maria Virgínia dos Santos[2], Maria Aparecida Sibaldo[2]

[1]Departamento de Computação – Universidade Federal Rural de Pernambuco (UFRPE)

[2]Universidade Federal do Agreste Pernambucano (UFAPE)

rodrigo.npmsouza@ufrpe.br

Abstract. We present a short, combinatorial proof of a classical, old, and fundamental question in the theory of automata with multiplicities on the semiring $\mathbb{N}$ of natural numbers: is the formal series realized by an $\mathbb{N}$-automaton bounded by an integer n? Essentially two proof strategies have been proposed in the literature, the first, by Mandel and Simon (1977), depends on algebraic properties of matrix semigroups, the second, by Seidl and Weber (1991), consists of intricate structural elaborations. Our proof, more combinatorial, is short and self-contained.

Resumo. Apresentamos uma prova combinatória, curta, a uma questão clássica, antiga e fundamental na teoria dos autômatos com multiplicidade no seminanel $\mathbb{N}$ dos números naturais: a série formal realizada por um $\mathbb{N}$-autômato é limitada por um inteiro n? Essencialmente duas estratégias de prova foram propostas na literatura, a primeira, de Mandel e Simon (1977), depende de propriedades algébricas de semigrupos de matrizes, a segunda, de Seidl e Weber (1991), consiste de elaborações estruturais intrincadas. Nossa prova, mais combinatória, é curta e auto-contida.

1. Introdução

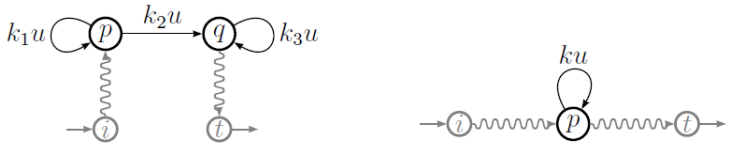
O estudo de autômatos com multiplicidade em um semi-anel forma um capítulo da Teoria dos Autômatos ao mesmo tempo matematicamente sofisticado e rico em aplicações. Um dos problemais clássicos nesse capítulo é caracterizar e decidir, com base em propriedades de um autômato, se a série formal realizada por esse autômato é limitada (= a multiciplidade de toda palavra é inferior a um valor fixo).

Nossa contribução nesta comunicação reside em uma das mais antigas dessas caracterizações, envolvendo autômatos com multiplicidades no semi-anel dos naturais, que é baseada em certos padrões que podem ser formados pelos passeios do autômato. Formalmente: dado um $\mathbb{N}$-autômato $\mathcal{A}$, existe um inteiro k tal que $us \leq k$, para toda palavra u, onde s é o comportamento (a série realizada) por $\mathcal{A}$? Em linhas gerais, a presença de certos padrões de passeio em $\mathcal{A}$ é o que determina s ser limitada ou não. Essa abordagem é recorrente em muitos outros problemas, envolvendo, inclusive, outros objetos como transdutores.

A primeira prova de que essa propriedade é decidível foi proposta por Mandel e Simon [Mandel and Simon 1977] (e independentemente por Jacob [Jacob 1977]), e consiste em uma redução ao problema da finitude de um monóide S de matrizes

$\mathbb{N}^{n\times n}$ finitamente gerado. Uma prova estrutural foi apresentada em 1991 por Seidl e Weber [Weber and Seidl 1991], que por um lado tem o mérito de explicitar os padrões de caminhos que determinam a limitação da série realizada, mas por outro é longa e trabalhosa.

Propomos uma prova curta, combinatória e auto-contida da caracterização dos autômatos $\mathbb{N}$-limitados, que apresentamos integralmente neste texto. A mesma depende dos esquemas de passeios ilustradas a seguir:



(a) Um haltere. (b) Um circuito pesado ($k > 0$).

Figura 1. Padrões proibidos em um $\mathbb{N}$-autômato.

Vamos chamar esses dois tipos de passeios – circuito pesado e haltere – de padrões proibidos. O resultado que vamos demonstrar se enuncia como segue:

Teorema 1 (Mandel-Simon 1977, Seidl-Weber 1991). Um $\mathbb{N}$-autômato bi-acessível $\mathcal{A}$ é limitado sse: $\mathcal{A}$ não tem nenhum circuito pesado; $\mathcal{A}$ não tem nenhum haltere.

Os conceitos e definições discutidos neste texto, bem como um panorama de resultados fundamentais, podem ser encontrados em [Sakarovitch 2009].

2. Prova

Basta provar a direção não-trivial, que é: se a série é limitada, o autômato tem ao menos um padrão proibido. Seja $\mathcal{A}$ um autômato com n estados, sobre o alfabeto A, e (λ, μ, ν) sua representação linear, de dimensão Q. Seja L o máximo dentre as multiciplicidades das transições de $\mathcal{A}$:

$$L = \mathsf{max}\left\{a\mu_{p,q} \mid a \in A,\ p,\, q \in Q\right\},$$

Vamos mostrar que, se $u \in A^*$ é uma palavra tal que

$$u|\mathcal{A}| \geq \left(\sum_{q\in Q} \lambda_q\right) (nL)^{2^n} \left(\sum_{q\in Q} \nu_q\right),$$

então $\mathcal{A}$ contém um padrão proibido.

Seja

$$u = a_1 \dots a_l \qquad (a_i \in A \text{ para } 1 \leq i \leq l).$$

Para cada i, $0 \leq i \leq l$, definimos um vetor e um conjunto de estados como segue (denotamos a linha p de uma matriz m por $m^{[p]}$):

$$\mathsf{v}^{(i)} = \lambda \cdot (a_1 \dots a_i)\mu, \qquad X_i = \left\{p \in Q \;\middle|\; ((a_{i+1} \dots a_l)\mu)^{[p]} \cdot \nu > 0\right\}.$$

Nessa definição, considere

$$\mathsf{v}^{(0)} = \lambda, \qquad X_l = \{p \in Q \mid \nu_p > 0\}.$$

Segue por indução que para todo estado p, $\mathsf{v}_p^{(i)}$ é a soma das multiplicidades dos passeios rotulados por $a_1 \dots a_i$ de um estado inicial a p, multiplicadas pelas multiplicidades iniciais respectivas, e que p pertence a X_i sse $\mathcal{A}$ tem um passeio de p a um estado final rotulado por $a_{i+1} \dots a_l$ (recorde que $\mathcal{A}$ é bi-acessível). Temos também que, para todo i e j, $0 \leq i < j \leq l$,

$$\forall\, p \in X_i \ \exists\, q \in X_j \ \ p \xrightarrow[\mathcal{A}]{a_{i+1} \dots a_j} q \tag{1}$$

$$\forall\, q \in X_j \ \exists\, p \in X_i \ \ p \xrightarrow[\mathcal{A}]{a_{i+1} \dots a_j} q \tag{2}$$

$$\forall\, q \in X_j \ \ \mathsf{v}_q^{(j)} = \sum_{p \in X_i} \mathsf{v}_p^{(i)}(a_{i+1} \dots a_j)\mu_{p,q} \tag{3}$$

Seja

$$s_i = \sum_{q \in X_i} \mathsf{v}_q^{(i)} \qquad 0 \leq i \leq l.$$

Segue de (1) que

$$s_0 \leq s_1 \leq \cdots \leq s_l.$$

Vamos construir a partir desses objetos um grafo dirigido rotulado $\mathcal{X}$ onde cada arco representa um passeio de $\mathcal{A}$ e que contém um padrão proibido. Para isso, vamos encontrar dois índices i e j tais que X_i e X_j coincidam. Seja

$$Y = \{i \in \mathbb{N} \mid 1 \leq i \leq l,\ s_{i-1} < s_i\}.$$

Escrevamos os elementos nesse conjunto em ordem crescente como

$$i_1, i_2, \dots, i_m$$

(onde m é a cardinalidade de Y). Considerando $i_0 = 0$, segue de (3) que

$$\forall\, i_j \in Y \ \ s_{i_j} \leq s_{(i_j - 1)}(nL).$$

Pela definição de Y, temos que $s_{i_{(j-1)}} < s_{i_j}$ e $s_{(i_j - 1)} = s_{i_{(j-1)}}$. Assim,

$$\forall\, i_j \in Y \ \ s_{i_{(j-1)}} < s_{i_j} \leq s_{i_{(j-1)}}(nL).$$

Isso implica em

$$s_l = s_m \leq s_0 (nL)^m,$$

e como

$$s_l \left(\sum_{q \in Q} \nu_q \right) \geq u|\mathcal{A}|,$$

obtemos

$$m \geq 2^n.$$

Assim, existem dois índices distintos i_j e $i_{j'}$ tais que

$$X_{i_j} = X_{i_{j'}}.$$

Observamos que, a despeito dessa identidade, tem-se

$$s_{i_j} < s_{i_{j'}}.$$

Seja

$$X = X_{i_j}, \qquad v = a_{i_j+1} \dots a_{i_{j'}}.$$

O conjunto dos vértices do grafo $\mathcal{X}$ é X, e os arcos são definidos por

$$\forall\, p, q \in X \;\; p \xrightarrow[\mathcal{X}]{kv} q \iff p \xrightarrow[\mathcal{A}]{kv} q.$$

Observamos que todos os arcos de $\mathcal{X}$ tem o mesmo suporte — a palavra v.

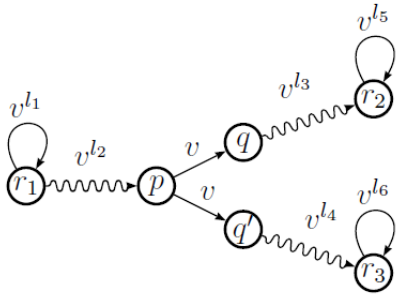
Claramente, um padrão proibido em $\mathcal{X}$ é também um padrão proibido em $\mathcal{A}$. Vamos mostrar que esse grafo tem um tal padrão, a partir da propriedade seguinte, consequência de (1) e (2): todo vértice de $\mathcal{X}$ tem pelo menos um arco entrando, e ao menos um arco saindo.

Vamos considerar dois casos. Supomos primeiro que os arcos de $\mathcal{X}$ induzem uma permutação de X, ou seja, $\mathcal{X}$ é uma união de ciclos. Como $s_{i_{j'}} > s_{i_j}$, segue de (3) que $\mathcal{X}$ tem pelo menos um arco com multiciplidade maior do que 0. Assim, $\mathcal{X}$ contém um circuito pesado.

Suponha que $\mathcal{X}$ não é uma união de ciclos. Então existe um estado $p \in X$ e ao menos dois arcos distintos com origem nesse vértice:

$$p \xrightarrow[\mathcal{X}]{v} q \quad \text{e} \quad p \xrightarrow[\mathcal{X}]{v} q' \qquad (q \neq q').$$

As multiciplidades dos arcos não tem nenhum papel nesse caso. Como cada vértice de $\mathcal{X}$ tem pelo menos um arco saindo, q e q' são acessíveis a algum circuito; e como cada vértice tem pelo menos um arco entrando, existe um circuito acessível a p. Essa situação está ilustrada abaixo:



Se $r_1 \neq r_2$, então podemos construir um haltere de r_1 a r_2 (rotulado por uma potência de de v). Vale o mesmo se $r_1 \neq r_3$. E se $r_1 = r_2 = r_3$, podemos construir um haltere de q a q' (recorde que esses dois estados são distintos).

Isso conclui nossa prova.

Referências

Jacob, G. (1977). Un algorithme calculant le cardinal, fini ou infini, des demi-groupes de matrices. Theor. Comput. Sci., 5(2):183–204.

Mandel, A. and Simon, I. (1977). On finite semigroups of matrices. Theor. Comput. Sci., 5(2):101–111.

Sakarovitch, J. (2009). Elements of Automata Theory. Cambdrige University Press.

Weber, A. and Seidl, H. (1991). On the degree of ambiguity of finite automata. Theor. Comput. Sci., 88(2):325–349.

Análise Comparativa dos Operadores do Algoritmo Genético aplicado ao Problema de Corte Bidimensional em Madeira

Abstract. *When building a wood composite product it is important to plan with the intention of reducing material consumption. This way, it is possible to use heuristics to obtain the necessary cuts to achieve a lower cost. Thus, the objective of this work is to identify a type of function selection operator to be integrated into the Parallel Genetic Algorithm, seeking to find the best way to cut wooden pieces. To reduce the processing cost, the Genetic Algorithm was developed in a distributed way with the help of the Message Passing Interface (MPI) API, and the selection operators adopted were sequential biased roulette and biased roulette with and without migration, in order to verify which selection operator best meets the solution of the cutting problem. With the experiments, it was possible to identify that the roulette with migration selection method stood out when compared to others.*

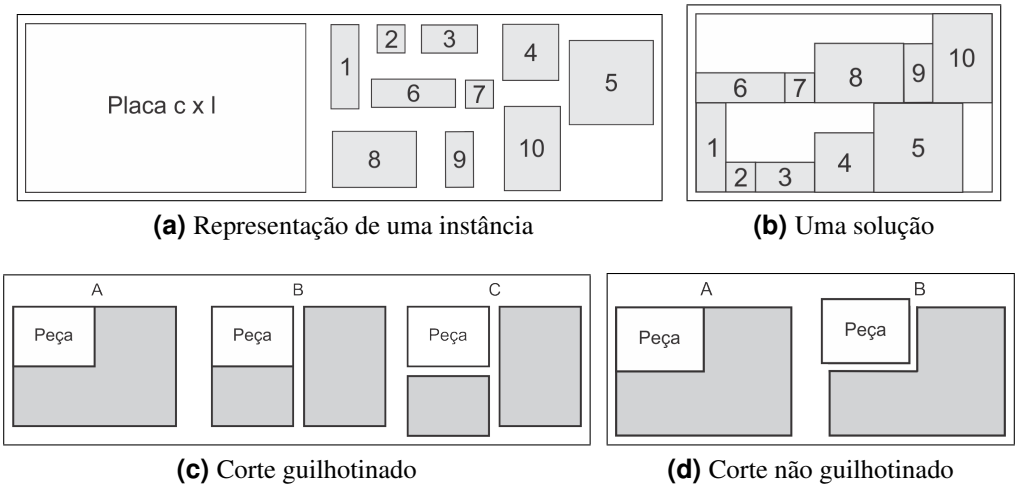
1. Introdução

As indústrias, em geral, se deparam com o Problema de Corte ao produzir determinados produtos, ou seja, necessitam realizar cortes em determinada peça com o intuito de retirar os itens desejados. Esse problema tem motivado pesquisadores devido as diversas aplicações [Rangel and Figueiredo 2008]. Em [Wäscher et al. 2007], os autores apresentam uma tipologia com base em critérios como: tipo de atribuição, natureza das dimensões e sortimento de itens.Em [Parmar et al. 2014], os autores realizaram um comparativo entre as principais técnicas adotadas para encontrar uma solução para o Problema de Corte, entre elas os Algoritmos Genéticos. Os Algoritmos Genéticos (do inglês, *Genetic Algorithm* (GA)) apresentaram bons resultados, no entanto, tiveram um custo computacional elevado quando comparado as duas melhores técnicas analisadas. Um alternativa possível é a adoção dos Algoritmos Genéticos Paralelos, a fim de que o tempo gasto para encontrar a solução seja reduzido. É válido ressaltar que ao realizar mudanças no Algoritmo Genético Sequencial para torna-lo paralelo e distribuído, um dos operadores que merece atenção é o operador de seleção. Dessa forma, o objetivo desse trabalho consiste em identificar um operador de seleção viável para o problema, dentre eles, o operador da roleta, ou seja o sequencial, e o paralelo sem migração e com migração, aplicados no problema de corte para madeira.

2. Problema de Corte

Conforme o trabalho de [Jorge et al. 2015], o problema de corte e empacotamento busca determinar um arranjo de objetos pequenos (itens) dentro de objetos maiores (recipientes), obedecendo a certas restrições e visando melhorar a qualidade da solução sob alguma perspectiva. Estes problemas são de difícil resolução, principalmente pelas restrições de não sobreposição entre os itens e posicionamentos dos itens de forma que não ultrapasse a capacidade do recipiente. Na Figura 1a é possível observar uma representação de uma instância do problema. No lado esquerdo da figura, tem-se uma placa (recipiente) com

dimensões c referente ao comprimento e l representando a largura do recipiente; no lado direito, encontram-se as peças (itens) que serão organizados de forma que se gaste o mínimo possível da placa.



(a) Representação de uma instância **(b)** Uma solução

(c) Corte guilhotinado **(d)** Corte não guilhotinado

Figura 1. a) Representação de uma instância do problema de corte. b) Exemplo de uma solução, c) e d) Tipos de corte

Ao fazer uma análise dos tipos de corte, pode-se destacar duas estratégias, sendo a primeira o corte guilhotinado, como mostrado na Figura 1c. Então, nesse tipo de corte, verifica-se a disponibilidade de espaço para encaixar a peça na placa. Realiza-se um corte verticalmente em toda placa e um corte horizontalmente em toda placa. O segundo tipo é o corte não guilhotinado, que pode ser visto na Figura 1d, esse corte não é feito por toda placa, ou seja, o tamanho do corte será conforme o tamanho da peça desejada, sendo assim ao final do corte obtém-se uma peça e um saldo, diferente do corte guilhotinado que, no exemplo anterior, resultou em uma peça e dois saldos.

Diante do problema, se faz necessário escolher uma técnica para soluciona-lo. Existem diversos trabalhos na literatura sobre o Problema de Corte. Dentre eles, um alinhado a esse trabalho é o *survey* de [Parmar et al. 2014], onde os autores fazem um levantamento sobre os principais trabalhos relacionados ao Problema de Corte. Em seguida, são feitos vários comparativos entre as técnicas de solução abordadas: GRASP (*Greedy Randomized Adaptive Search Procedure*); EP (*Evolutionary Programming*); GA (*Genetic Algorithm*); ACO (*Ant Colony Optimization*); PSO (*Particle Swarm Optimization*).

3. Metodologia

Após o aprofundamento no problema e das possíveis soluções existentes na literatura, foi desenvolvido um algoritmo para que o mesmo fosse capaz de encontrar soluções para resolver o Problema de Corte. Para minimizar o tempo computacional, ele foi desenvolvido de forma paralela e distribuída, em conjunto com a API MPI. Foram implementado três GAs, sendo um sequencial e dois paralelos, o GA sequencial é o citado nas seções anteriores e os GA paralelos serão abordados a seguir:

3.1. Algoritmo Genético com o operador de seleção sem migração

Ao analisar um GA com objetivo de torna-lo distribuído, é possível observar que o ponto mais crítico é o operador de seleção. Em um GA tradicional, o operador de seleção

precisa ter conhecimento do valor da avaliação de cada indivíduo - isto é, o operador de *crossover* necessita de dois indivíduos, já o de mutação precisa apenas do filho gerado pelo *crossover*. Diante desse problemática, o primeiro GA citado divide a população pela quantidade de processos, de forma que o operador de seleção tem conhecimento apenas de uma parte da população, a população que se encontra no processo atual. Nesse caso, não há comunicação entre os operadores de seleção. Assim, ao iniciar cada processo, a população terá apenas N/P indivíduos, onde N é o tamanho da população total e P a quantidade de computadores distribuídos.

3.2. Algoritmo Genético com o operador de seleção com migração

Nesse segundo GA, a população também é dividida em P partes. No entanto, o operador de seleção de cada computador se comunica, de forma que pode ser feita uma abstração observando-se como uma única população. Cada processo avalia os indivíduos da sua população, em seguida cada processo envia o somatório das avaliações dos indivíduos para todos os outros computadores conectados na rede. Quando cada processo recebe a soma dos demais, é feita a soma global com todos os indivíduos. Com a soma da avaliação de todos os indivíduos, inclusive dos computadores conectados, cada processo pode selecionar os pais de qualquer processo. Seguindo os passos do método da roleta, então será escolhido um valor de 0 até a soma de todos os indivíduos. Em seguida, será verificado a qual população o indivíduo pertence, isto é, será verificado qual o computador a população pertence, para que o valor escolhido seja enviado para o processo adequado. Quando um processo P_i recebe uma mensagem de outro processo P_j, o processo P_i identificará qual o indivíduo correspondente e enviará o cromossomo para o processo P_j. Ao receber todos os indivíduos, o algoritmo segue para o passo da reprodução. Dessa forma, um processo não fica limitado apenas aos seus indivíduos

3.3. Resultados computacionais

Para o executar o algoritmo, é preciso de alguns parâmetros como o tamanho da população, número de gerações, a taxa de mutação e as peças a serem cortadas. Sendo assim, uma entrada com dados reais para o problema, onde tem-se uma caixa com um tampa encaixável, que no total tem dimensões 20 cm de comprimento, 15 cm de largura e 11 cm de altura, possuindo 10 peças montar a mesma. Então, para os testes realizados foram inseridos, como entrada, 10 caixas com essas dimensões, e a placa a ser utilizada para o corte tem 185 cm de comprimento por 275 cm de largura.

Após a implementação dos GAs, foram realizados os testes a fim de se verificar o comportamento de cada algoritmo. Para cada algoritmo foram feitas variações no tamanho da população e na quantidade de gerações. Com 25 gerações cada algoritmo foi executado 30 vezes com o tamanho da população 40, 80, 120 e 160, totalizando 120 execuções para um algoritmo com 25 gerações. No final, foram gerados 4 valores correspondendo a média de cada 30 execuções com mesmos parâmetros. Ao concluir todos os experimentos, foram gerados quatro gráficos, sendo um parâmetro fixo que pode ser visto na Figura 2, no primeiro com 25 gerações, o segundo com 55 gerações, o terceiro com 85 gerações e o ultimo com 115 gerações. O melhor valor encontrado pelos algoritmos foi 175 cm de saldo na altura. Todos os algoritmos encontraram 175 cm, no entanto, nas várias execuções, o AG-CM (GA com migração) mostrou-se com maior frequência, em seguida o operador AG (Sequencial) e por último o AG-SM (GA sem migração), com a menor frequência.

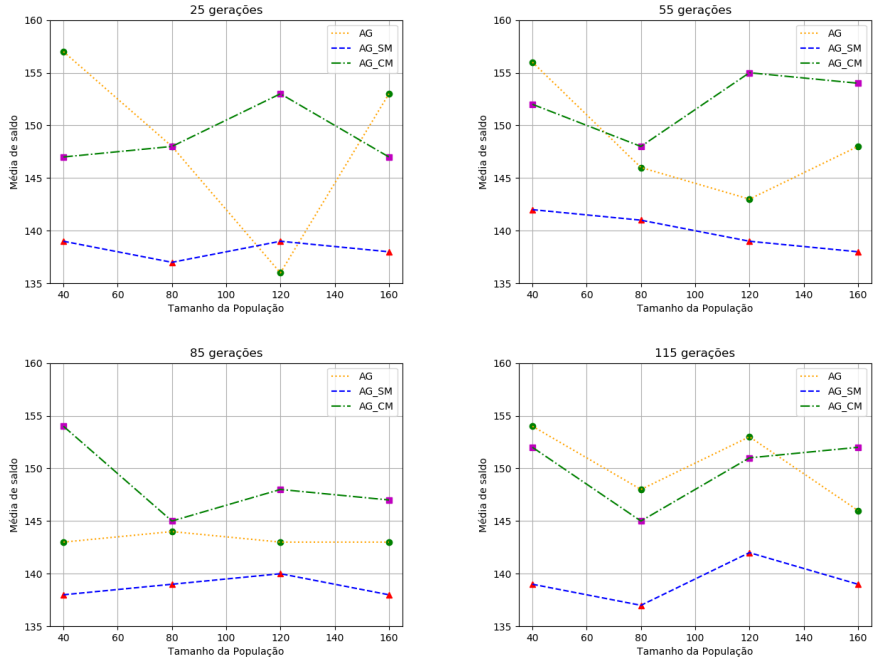


Figura 2. Resultados do experimento

4. Conclusão

Diante do trabalho realizado, foi possível identificar o bom desempenho do GA paralelo como o operador de seleção da roleta com migração. No entanto, é interessante fazer uma análise do problema para verificar se realmente se faz necessário o uso de GA paralelo. Trata-se de uma boa dica para problemas com o tamanho de população alto, pois cada processo fica com uma parte da população, sem que se perca a comunicação entre os processos. Diminui-se, dessa forma, o tempo de processamento total, ocorrendo ainda um aumento na memória devido ao alto tamanho da população. Como trabalhos futuros, pode-se encontrar uma modelagem matemática que aceite peças irregulares, como o círculo, entre outras, integrando-a a um módulo gráfico que facilite para os usuários finais, como marceneiros ou até mesmo clientes que desejam verificar quanto de material será gasto na fabricação de seus produtos.

Referências

Jorge, A. R., Mundim, L. R., Cherri, L. H., and Andretta, M. (2015). O problema de corte de itens irregulares: aplicaçao na industria de aventais e forros de luva. *Simpósio Brasileiro de Pesquisa Operacional, Porto de Galinhas, Pernambuco-PE*.

Parmar, K. B., Prajapati, H. B., and Dabhi, V. K. (2014). Cutting stock problem: A survey of evolutionary computing based solution. In *International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE)*, pages 1–6. IEEE.

Rangel, S. and Figueiredo, A. G. d. (2008). O problema de corte de estoque em indústrias de móveis de pequeno e médio portes. *Pesquisa Operacional*, 28(3):451–472.

Wäscher, G., Haußner, H., and Schumann, H. (2007). An improved typology of cutting and packing problems. *European journal of operational research*, 183(3):1109–1130.

Author Index