Heurísticas para o Problema de Partição de Strings Comuns Mínima

Wallesson C. S.[1], Paulo Henrique M. A.[1], Fábio C. S. D.[1], Emanuel F. C.[1], Criston P. S.[1]

[1]Universidade Federal do Ceará (UFC)
Quixadá – CE – Brazil

wallessondm@alu.ufc.br, phmacedoaraujo@ufc.br, fabiodias@ufc.br,

emanuel.coutinho@ufc.br, criston@ufc.br

Abstract. *The large number of available data related to computational biology makes studies necessary in search of better analyzes and tools. A problem that arises in this field is called Minimum Common String Partition problem (MCSP). This work presents two heuristics within a time-controlled environment. For that, we run experiments with artificial and realistic instances to evaluate the performance of such heuristics. By the results of the experiments, we see that one of the heuristics is promising, although its results do not surpass those of the "state of the art".*

Resumo. *O grande número de dados disponíveis relacionados à biologia computacional torna necessários estudos em busca de melhores análises e ferramentas. Um problema que surge neste campo é chamado de problema de Partição de Strings Comuns Mínima (em inglês MCSP). Este trabalho apresenta duas heurísticas dentro de um ambiente controlado por tempo. Para isso, executamos experimentos com instâncias artificiais e realistas para avaliar o desempenho de tais heurísticas. Pelos resultados dos experimentos, vemos que uma das heurísticas é promissora, embora seus resultados não superado o "estado da arte".*

1. Introdução

Biologia computacional é um dos temas de estudo da ciência da computação que vem tendo crescimento nas últimas decadas. Alguns dos problemas abordados em tais estudos são os de otimização, mais especificamente, os de otimização combinatória. Problemas de otimização combinatória são aplicados para encontrar soluções em conjuntos de dados finitos, onde é fácil testar se os elementos de uma solução pertencem ao domínio, porém é inviável testar todas as soluções. Entre os problemas de biologia computacional existem os que se dedicam à comparação de strings. Um dos problemas conhecidos é o problema de Partição de Strings Comuns Mínima (*Minimum Common String Partition*, em inglês MCSP) [C. Blum and Davidson 2015, Ferdous and Rahman 2013].

O problema MCSP recebe duas strings s_1 e s_2, onde $|s_1| = |s_2| = n$. As strings são definidas por um alfabeto Σ com quantidade de símbolos finita. Para cada símbolo w do alfabeto, a quantidade de ocorrências de w em cada uma das strings s_1 e s_2 é igual, podendo ser igual à zero. Uma partição de uma string s é um conjunto de substrings compostas por caracteres consecutivos em s, onde não há sobreposição entre eles, ou

seja, um caracter de uma posição específica em s não pode pertencer a mais de uma das substrings da partição. Uma solução viável é dada por duas partições p_1 e p_2, cada uma relacionada a uma das strings da entrada, tais que p_1 = p_2. O valor de uma solução é dado pelo tamanho das partições $|p_1|$ = $|p_2|$. O problema MCSP busca encontrar uma solução com o menor tamanho de partições. Por exemplo, seja $s_1 = \text{“}GUTAGATT\text{”}$ e $s_2 = \text{“}AGUTTTGA\text{”}$. Podemos particionar as strings em p_1 = $\{\text{“}GUT\text{”}, \text{“}A\text{”}, \text{“}GA\text{”}, \text{”}TT\text{”}\}$ e p_2 = $\{\text{“}A\text{”}, \text{“}GUT\text{”}, \text{“}TT\text{”}, \text{“}GA\text{”}\}$. Dispondo dessa forma, o valor da solução viável encontrada é 4. O problema sempre tem solução viável, pois a partição de tamanho n é sempre válida. O objetivo deste trabalho é comparar as heurísticas aqui propostas, a saber, GRP-2 e GELD-2 com a heurística 2-*phase* (conhecida como “estado-da-arte”) de [C. Blum and Davidson 2015], a qual é baseada em um modelo de programação linear inteira (PLI), e com a heurística gulosa de [M. Crobak and J.Sgall 2004] para o problema MCSP. Os resultados são promissores para a heurística GELD-2, porém não são favoráveis para a heurística GRP-2.

2. Algoritmos GRP-2 e GELD-2 para o problema MCSP

Nesta seção apresentamos os algoritmos GRP-2 e GELD-2 propostos por este trabalho para o problema MCSP. Os algoritmos recebem como entrada duas strings s_1 e s_2 e retornam uma solução viável x para o problema MCSP.

2.1. Algoritmo GRP-2

O algoritmo GRP-2 utiliza a função denominada de *Construção* para geração de soluções. A função *Construção* recebe uma lista L de substrings maiores que 1. Depois, seleciona aleatoriamente uma substring entre as 20% primeiras substrings de L e adiciona-se à solução. Em seguida, todas as substrings em L que se sobrepôem à substring selecionada são removidas de L. O procedimento é realizado até que L não possua mais substrings. Ao final, são adicionadas as substrings de tamanho 1, que estão descobertas, à solução, e por último a solução gerada é retornada. A ideia geral do algoritmo GRP-2 é executar chamadas à função *Construção* e armazenar a melhor solução encontrada de acordo com um limite de tempo de execução imposto ao algoritmo.

2.2. Algoritmo GELD-2

O algoritmo GELD-2 utiliza uma função chamada *GulosoEstendido*, a qual recebe como entrada as strings s_1 e s_2, uma substring w para marcar como coberta nas strings da entrada no início da execução. Depois, executa a heurística gulosa de [M. Crobak and J.Sgall 2004]. A heurística gulosa seleciona e marca como coberta a maior substring comum a s_1 e s_2 que ainda não tenha sido coberta, repetindo esse procedimento até que todas as substrings comuns à s_1 e s_2 sejam cobertas. O GELD-2 executa iterativamente a função *GulosoEstendido*, uma chamada para cada substring w inicial diferente. A ordem das substrings iniciais passadas para a função *GulosoEstendido* em cada iteração é definida pelo tamanho das substrings de maneira decrescente, ou seja, as maiores substrings em comum a s_1 e s_2 são passadas nas primeiras iterações do GELD-2. Este procedimento é realizado até que um limite de tempo seja atingido.

3. Execução e análise dos experimentos

Os experimentos deste trabalho foram realizados em um computador com Intel Core i3-4005U de 1.70 GHz, memória RAM de 8 GB e sistema operacional Windows 7 Ul-

timate. O tempo que os algoritmos aqui propostos executaram tem como limite superior o tempo de execução que o algoritmo 2-*phase* de [C. Blum and Davidson 2015] levou para encontrar a solução retornada para a instância observada, disponibilizados em [C. Blum and Davidson 2015]. Destacamos que o ambiente computacional de [C. Blum and Davidson 2015] é melhor do que o usado neste trabalho, e que o tempo de execução da heurística gulosa de [M. Crobak and J.Sgall 2004] é insignificante. As instâncias dos experimentos são divididas em conjuntos artificiais e reais de DNA disponibilizado por [Ferdous and Rahman 2013]. Os dados artificiais possuem um total de 30 instâncias divididas em três subconjuntos de strings com comprimento de até 200, de 201 a 400 e de 401 a 600 bps (pares de base), denominados de Grupo 1, Grupo 2 e Grupo 3 respectivamente. Os dados reais consistem em 14 instâncias realistas que variam entre 200 e 600 bps. Os resultados obtidos pela execução dos algoritmos GRP-2 e GELD-2 são mostrados nas Tabelas 1 e 2, indicando respectivamente as soluções das instâncias artificiais e das realistas. Valores em negrito mostram a melhor solução. O símbolo * é usado para mostrar soluções ótimas conhecidas para o problema. Para comparar os resultados, destacamos a diferença em porcentagem na média das soluções, onde valores negativos indicam o quão os resultados foram melhores e valores positivos indicam o contrário.

Para o Grupo 1, a diferença média das soluções do GELD-2, em relação ao $Guloso$, 2-*phase* e GRP-2 foi respectivamente -6,6%, 4,4%, e -29,9% aproximadamente. Já o GRP-2 em relação ao $Guloso$, 2-*phase* e GELD-2 foi respectivamente 33,2%, 49%, e 29,9% aproximadamente. Para o Grupo 2, a diferença média das soluções do GELD-2 em relação ao Guloso, 2-*phase* e GRP-2 foi respectivamente -4%, 8,7% e -37,1% aproximadamente. Já o GRP-2 em relação ao Guloso, 2-*phase* e GELD-2 foi respectivamente 52,6%, 72,9% e 37,1% aproximadamente. Para o Grupo 3, o GELD-2 em relação ao Guloso, 2-*phase* e GRP-2 foi respectivamente -3,7%, 3,9%, e -40,4% aproximadamente. O GRP-2 em relação ao Guloso, 2-*phase* e GELD-2 foi respectivamente 61,53%, 74,4%, e 40,4% aproximadamente. Observe que, apesar de tudo, o GELD-2 conseguiu encontrar uma solução melhor que a heurística 2-*phase* para a instância 9. Para o grupo de instâncias reais, onde os valores são apresentados na Tabela 2, o GELD-2 em relação ao Guloso, 2-*phase* e GRP-2 foi respectivamente -2%, 6,3%, e -37,8% aproximadamente. E o GRP-2 em relação ao Guloso, 2-*phase* e GELD-2 foi respectivamente 57,7%, 71,22%, e 37,8% aproximadamente.

Tabela 1. Resutados para as Artificiais

Resultados para Grupo 1					Resultados para Grupo 2				Resultados para Grupo 3			
Instância	GRP-2	GELD-2	2-phase	Guloso	GRP-2	GELD-2	2-phase	Guloso	GRP-2	GELD-2	2-phase	Guloso
1	60	44	**42**	46	177	114	**103**	119	299	176	**172**	182
2	78	51	**48**	56	187	119	**110**	122	301	172	**165**	175
3	82	57	**53**	62	173	110	**99**	114	322	189	**180**	196
4	71	44	**43**	46	186	113	**105**	116	307	181	**171**	192
5	60	41	**40***	44	216	131	**120**	135	275	168	**163**	176
6	61	42	**40***	48	171	103	**97**	108	274	164	**155**	170
7	83	62	**57**	65	146	102	**91**	108	265	167	**160**	173
8	65	47	**44**	51	191	117	**108**	123	285	176	**166**	185
9	62	**44**	46	46	181	117	**109**	124	289	170	**169**	174
10	80	60	**58**	63	164	101	**94**	105	281	164	**160**	171
Média	70,2	49,2	**47,1**	52,7	179,2	112,7	**103,6**	117,4	289,8	172,7	**166,1**	179,4

Tabela 2. Resutados para as Reais

Resultados para os Reais				
Instância	GRP-2	GELD-2	2-phase	Guloso
1	129	90	**80**	95
2	245	155	**144**	161
3	187	117	**112**	121
4	268	164	**157**	172
5	207	167	**161**	153
6	249	148	**139**	140
7	217	136	**127**	134
8	218	129	**120**	149
9	238	143	**131**	151
10	237	148	**136**	126
12	300	137	**130**	180
13	229	176	**163**	152
14	242	144	**142**	157
15	265	153	**145**	157
Média	230,7857	143,3571	**134,786**	146,2857

É notável a superioridade do algoritmo GELD-2 em relação ao GRP-2. O GELD-2 mostrou bons resultados em comparação ao algoritmo $Guloso$. Já em relação ao 2-*phase*, os algoritmos deste trabalho mostram inferioridade na qualidade das soluções, mostrando pior desempenho nas soluções do algoritmo GRP-2. Pelo fato do 2-*phase* utilizar PLI para geração de soluções parciais, acreditamos que seja o motivo de possuir melhor qualidade nas soluções em comparação aos algoritmos deste trabalho.

4. Conclusões e trabalhos futuros

Neste trabalho, apresentamos a heurística Guloso Estendido de Lista Dinâmica 2 (GELD-2) e uma heurística adaptada da meta-heurística GRASP (GRP-2) para o problema MCSP. Foram avaliadas as soluções obtidas pelos algoritmos propostos em relação às heurísticas gulosa de [M. Crobak and J.Sgall 2004] e a 2-*phase* de [C. Blum and Davidson 2015]. Através dessas análises, o algoritmo GELD-2 mostrou resultados promissores, mesmo não conseguindo superar o 2-*phase*, já o GRP-2, mostrou-se ineficiente na busca por soluções, atingindo resultados piores em todos as instâncias comparando com as soluções dos demais algoritmos. Uma boa perspectiva futura seria buscar adaptações heurísticas nos dois algoritmos GELD-2 e GRP-2. Para o GRP-2, uma estratégia poderia ser aplicada na chamada do algoritmo *Construção* para escolher melhores substrings para a inclusão da solução. Já para o GELD-2, pode ser avaliada uma busca local eficiente na solução corrente a fim de melhorá-la.

Referências

C. Blum, J. A. L. and Davidson, P. (2015). Mathematical programming strategies for solving the minimum common string partition problem. *European Journal of Operational Research*, 242(3):769–777.

Ferdous, S. M. and Rahman, M. S. (2013). Solving the minimum common string partition problem with the help of ants. *Mathematics in Computer Science*, 11(2):233–249.

M. Crobak, P. K. and J.Sgall (2004). The greedy algorithm for the minimum common string partition problem. *ACM Transactions Algorithims*, 1(2):250–366.