Complexidade Aleatória de Problemas Computacionais

Edson Tadashi Miyamoto

DISSERTAÇÃO APRESENTADA
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO

PARA OBTENÇÃO DO GRAU DE
MESTRE EM
MATEMÁTICA APLICADA

Área de Concentração: Ciência da Computação
Orientador: **Prof. Dr. Yoshiharu Kohayakawa**

Durante a elaboração deste trabalho,
o autor recebeu apoio financeiro da CAPES

–São Paulo, Setembro de 1992–

Resumo

Nos quatro primeiros capítulos deste trabalho, estudamos a complexidade computacional do reconhecimento de propriedades de grafos invariantes por isomorfismos. Em particular, estudamos este problema para propriedades monotônicas não-triviais de grafos. Sabe-se atualmente que a complexidade *determinística* de pior caso dessas propriedades é $\Omega(n^2)$, onde n é o número de vértices dos grafos considerados. Existe entretanto uma conjectura de Yao e Karp de 1977 que diz que a complexidade *aleatória* destas propriedades é também $\Omega(n^2)$. Apresentamos resultados de Yao e King sobre esta conjectura e o melhor limite inferior conhecido atualmente de $\Omega(n^{4/3})$, provado recentemente por P. Hajnal.

Caso a conjectura de Yao e Karp venha a ser provada, pode-se pôr em dúvida a eficácia de métodos probabilísticos aplicados a problemas computacionais. No último capítulo apresentamos resultados de Valiant e Reischuk que comprovam que no caso do problema de seleção em paralelo do menor elemento de um conjunto, existe um algoritmo aleatório que é assintoticamente mais eficiente do que qualquer algoritmo determinístico. Neste mesmo capítulo, apresentamos um resultado de Alon e Azar que prova que no caso de ordenação de conjuntos com n elementos, usando algoritmos que utilizam mais de n processadores, a eficiência de algoritmos aleatórios não é melhor do que a de algoritmos determinísticos.

Abstract

In the first four chapters of this dissertation, we study the computational complexity of graph properties invariant under isomorphisms. More specifically, we study non-trivial monotone graph properties. It is now known that the *deterministic* complexity of such properties is $\Omega(n^2)$, where n is the number of vertices of the graphs considered. However, there is a conjecture due to Yao and Karp that goes back to 1977 that says that the *random* complexity of those properties is also $\Omega(n^2)$. We present results by Yao and King about this conjecture and the best lower bound known at the moment, namely $\Omega(n^{4/3})$, recently proved by P. Hajnal.

Assuming that the above conjecture is true, one might put in doubt the power of probabilistic methods applied to computational problems. In the last chapter, we present results by Valiant and Reischuk that prove that for the problem of selecting in parallel the smallest element of a given set, there exists a random algorithm which is asymptotically more efficient than any deterministic algorithm. In this same chapter, we present a result by Alon and Azar that shows that for the problem of sorting n elements, if we only consider parallel algorithms that use more than n processors, then the efficiency of random algorithms is not better than that of deterministic ones.

Agradecimentos

Gostaria de agradecer ao meu orientador, Yoshiharu, por sua infinita paciência (pelo menos até onde pude testar, e que certamente não foi pouco), por sua extrema dedicação (que o levou a digitar correções para enviá-las para mim o mais rápido possível), por seu minucioso perfeccionismo (corrigindo desde hífens até os mais escabrosos erros conceituais; se esta página contiver erros é porque sem dúvida não passou por suas mãos previamente) e por sua generosidade ao partilhar comigo de seus conhecimentos sem a mínima indignação diante da minha ignorância. Enfim gostaria de tornar público o quanto eu realmente me felicito pela escolha de meu orientador, tanto a nível pessoal quanto acadêmico.

Gostaria de agradecer também a todas as pessoas que me apoiaram e em alguns casos até acreditaram mais do que eu na realização deste trabalho. Agradeço a todas essas pessoas pela paciência e simpatia.

Finalmente, gostaria de agradecer à minha irmã Emi, já que a realização deste trabalho não teria sido possível sem seu apoio superlativo.

Conteúdo

Introdução iii

1 Definições iniciais 1

1.1 Propriedades de grafos 2

1.2 Árvores de decisão 3

1.3 Complexidade determinística média de pior caso 6

1.4 Complexidade aleatória 7

2 Primeiras técnicas 10

2.1 Dualidade e empacotamento 11

2.2 Técnicas e resultados básicos 14

2.3 Dois resultados probabilísticos 19

3 Técnicas utilizando grau máximo de grafos 24

3.1 Preliminares 25

3.2 Propriedades auxiliares 25

3.3 Propriedades com grafos minimais de grau baixo 28

3.4 O limite inferior de $\Omega(v^{5/4})$ 33

4 Aperfeiçoamento de resultados anteriores 37

4.1 Modificação de uma técnica anterior 38

4.2 Um resultado em empacotamento de grafos 40

4.3 Preliminares para o resultado mais recente 44

4.4 O resultado mais recente: $\mathcal{C}^{\mathcal{AI}}(P) = \Omega(v^{4/3})$ 47

5 Problemas de seleção 53

5.1 Definição dos problemas e do modelo de computação 54
5.2 Um limite inferior para selecionar o mínimo em paralelo 57
5.3 Um algoritmo paralelo aleatório para SELEÇÃO$_k$ 60
5.4 Ordenação 66
5.5 Um limite inferior para a complexidade aleatória de Ord(n, m) . . . 73

A Definições para grafos **77**

B Algumas notações **81**

Introdução

Em anos recentes, algoritmos aleatórios têm provado ser extremamente mais poderosos que algoritmos determinísticos. Hoje, talvez um dos melhores exemplos que ilustram este ponto seja o algoritmo de Dyer, Frieze, e Kannan [16] para a aproximação de volumes de corpos convexos em $\mathbb{R}^n$ para n grande.

Um resultado de Elekes [17] afirma que qualquer algoritmo determinístico polinomial ou, mais geralmente, sub-exponencial que estima o volume de um corpo convexo $B \subset \mathbb{R}^n$ é tal que só se pode garantir que sua estimativa está correta a menos de um fator exponencial em n. Aqui, supomos que o corpo é dado por um oráculo de pertinência, i.e., dado um ponto no $\mathbb{R}^n$ podemos descobrir se este ponto pertence ou não a B com uma consulta a este oráculo. Ademais, definimos a complexidade de um algoritmo como sendo o número de consultas que ele precisa fazer ao oráculo. O resultado de Elekes diz então que com um número sub-exponencial em n de consultas, podemos apenas restringir o volume de B entre um limite superior e um inferior cujo quociente é pelo menos $(2^{1/2} + o(1))^n$. (Para uma versão mais forte deste resultado, veja Bárány e Füredi [7].)

O algoritmo aleatório de Dyer, Frieze, e Kannan obtém uma estimativa do volume de B a menos de um fator entre $(1+\epsilon)^{-1}$ e $(1+\epsilon)$, para qualquer ϵ fixo, com um número polinomial em n de consultas ao oráculo, e com probabilidade pelo menos $1-\epsilon$. Aqui supomos que B é não-degenerado no seguinte sentido: escrevendo $B(0,\rho)$ para a bola de raio ρ centrada na origem, admitimos que sabemos de antemão que $B(0,r) \subset B \subset B(0,R)$, onde $r > 0$ e R são constantes arbitrárias. É facil de ver que sem hipóteses deste gênero um algoritmo como acima não pode existir. Mencionamos de passagem que este é um dentre vários algoritmos recentes que se baseiam na simulação de cadeias de Markov que se aproximam da distribuição estacionária rapidamente. (Veja [40] para uma discussão sobre esta classe

de algoritmos.) Grosseiramente falando, as consultas ao oráculo são escolhidas aleatoriamente de acordo com um certo passeio aleatório. Uma ótima referência para resultados relacionados a algoritmos de estimativa de volumes e aplicações é Dyer e Frieze [15].

O contraste entre o resultado negativo de Elekes e o resultado de Dyer, Frieze e Kannan é fascinante. O tema desta dissertação centra no poder que procedimentos computacionais adquirem quando permitimos o uso de passos aleatórios, como ilustrado no exemplo acima.

O âmbito em que estudamos este problema é porém restrito. De fato, restringimos o nosso estudo a uma conjectura de Yao e Karp sobre a complexidade aleatória de propriedades de grafos e a alguns problemas de ordenação e seleção. Com isto mantemos o nosso estudo sempre em 'terra firme', evitando uma teoria muito geral ou demasiado abstrata.

Para pôr a conjectura de Yao e Karp em perspectiva, precisamos primeiro discutir certos problemas em complexidade *determinística* de grafos. Consideramos um jogo $J(n, P)$ entre dois jogadores com as seguintes regras. Ambos os jogadores estarão pensando em grafos com conjunto de vértices $V = [n] = \{1, \ldots, n\}$, e eles estarão considerando uma propriedade fixa P de grafos. Isto é, escrevendo $\mathcal{G}_V$ para o conjunto de todos os grafos com conjunto de vértices V, temos que $P \subset \mathcal{G}_V$ e P é invariante por isomorfismos de grafos.

No começo do jogo, o segundo jogador, o *Diabinho*, 'escolhe' um grafo G. O objetivo do primeiro jogador, o *Alginho*, é descobrir o mais eficientemente possível se o grafo G tem ou não a propriedade P em questão. A cada rodada, o Alginho faz uma pergunta do tipo "os vértices x e y são adjacentes em G?" (x, $y \in V$), e o Diabinho responde consultando o seu grafo G. O objetivo do Alginho é descobrir se G pertence a P com o menor número possível de perguntas. Naturalmente, o objetivo do Diabinho é de maximizar tal número. Note que o grafo G só é revelado conforme o Alginho faz suas perguntas, de tal forma que é perfeitamente legal para o Diabinho ir construindo o grafo G de acordo com as perguntas prévias. Posto de outra forma, não há qualquer motivo para o Diabinho escolher um grafo no começo do jogo; ele pode ir construindo G da forma que mais lhe convier, considerando o desenrolar do jogo.

Como um exemplo, o leitor é convidado a provar que se P for a propriedade 'o

grafo G é conexo', então no jogo $J(n, P)$ o Diabinho pode forçar o Alginho a fazer $\binom{n}{2}$ perguntas. Assim, o Alginho é forçado a pedir para que o Diabinho revele todo o grafo antes que ele possa decidir se ele é ou não conexo. Tais propriedades $P \subset \mathcal{G}_V$ para os quais o Alginho pode ser forçado a fazer $\binom{n}{2}$ perguntas são conhecidas como propriedades *elusivas*, ou alternativamente *evasivas*.

Vamos dizer que a propriedade P é *não-trivial* se $\varnothing \neq P \neq \mathcal{G}_V$, e diremos que ela é *monotônica* se todo grafo contendo um subgrafo que tem P também tem P. Aanderaa, Rosenberg, Lipton, e Snyder (veja [30] e [36]) conjecturaram que toda propriedade monotônica não-trivial é tal que o Diabinho sempre tem uma estratégia que força o Alginho a fazer $\Omega(n^2)$ perguntas no jogo $J(n, P)$.

Conjectura 1 *Existe uma constante universal $c > 0$ tal que para todo n e qualquer propriedade monotônica não-trivial P sobre grafos em $\mathcal{G}_V$, o segundo jogador em $J(n, P)$ pode forçar pelo menos cn^2 rodadas em $J(n, P)$.*

A conjectura de Aanderaa, Rosenberg, Lipton, e Snyder foi provada por Rivest e Vuillemin [35] (veja também [34] e [9], Capítulo 8), que mostraram que a afirmativa acima vale para a constante $c = 1/16$. Esta constante foi melhorada para $1/9$ por Kleitman e Kwiatkowski [28]. Mais recentemente, Kahn, Saks, e Sturtevant [23] melhoraram esta constante para $1/4 + o(1)$ usando, entre outros, uma engenhosa e inesperada conexão estabelecida por eles entre este problema e a teoria de pontos fixos em ações de grupos finitos em espaços topológicos triangulados. O resultado principal destes autores é que se n é uma potência de um primo, então toda propriedade monotônica não-trivial é elusiva. (Pode-se deduzir daí que a Conjectura 1 vale para $c = 1/4 + o(1)$.) Outros resultados sobre elusividade de propriedades de grafos foram obtidos por Yap [45]. (Veja também [46], Capítulo 5.)

Embora os resultados acima sejam bastante satisfatórios, uma conjectura fascinante ainda está aberta. Best, van Emde Boas, e Lenstra [8], e independentemente Karp (veja [36]) conjecturam o seguinte.

Conjectura 2 *Toda propriedade monotônica não-trivial sobre grafos é elusiva.*

Observe que enquanto a Conjectura 1 afirma que o Diabinho pode forçar o Alginho a fazer um número quadrático de perguntas, aqui conjecturamos que o Diabinho pode forçar o Alginho a fazer todas as $\binom{n}{2}$ perguntas possíveis. Os melhores

resultados na direção desta segunda conjectura são os de Kahn, Saks, e Sturtevant, e os de Yap mencionados acima. Yao [43] provou a conjectura análoga para grafos bipartidos, usando também técnicas de topologia algébrica. King [26] estudou a conjectura análoga para grafos bipartidos dirigidos.

Podemos finalmente voltar a algoritmos aleatórios e à conjectura de Yao e Karp. Aqui mudamos as regras de $J(n,P)$, e permitimos ao Alginho usar uma moeda. No jogo modificado $J'(n,P)$, o Alginho está munido de uma moeda e ele pode a qualquer instante usá-la para decidir como proceder: se ele tiver várias opções para a próxima pergunta e não conseguir se decidir entre elas, ele pode usar a moeda para escolher uma aleatoriamente. Naturalmente, em $J'(n,P)$ não é permitido ao Diabinho ir construindo o grafo G ao longo do jogo, pois caso mantivéssemos esta liberdade $J'(n,P)$ se reduziria essencialmente a $J(n,P)$. O objetivo do Alginho em $J'(n,P)$ é de minimizar o número esperado de perguntas, usando um algoritmo aleatório apropriado $A = A(P)$, e o do Diabinho é de escolher o grafo $G = G(A,P) \in \mathcal{G}_V$ de tal forma que tal esperança seja a mais alta possível. Aqui admitimos que o Diabinho conhece de antemão o algoritmo A.

Considere os números reais μ para os quais existe um algoritmo aleatório $A = A(P,\mu)$ que o Alginho pode usar em $J'(n,P)$ tal que, para qualquer grafo G em $\mathcal{G}_V$ que o Diabinho escolha, a esperança do número de rodadas em $J'(n,P)$ é no máximo μ. Definimos a *complexidade aleatória* $C^{Al}(P)$ da propriedade P como sendo o ínfimo desses μ. Ademais, pomos $C^{Al}(n) = \min_P C^{Al}(P)$, onde o mínimo é tomado sobre todas as propriedades monotônicas não-triviais P sobre grafos em $\mathcal{G}_V$, $|V| = n$. A conjectura a seguir, devido a Yao e Karp, é a analoga à Conjectura 1 de Aanderaa, Rosenberg, Lipton, e Snyder neste contexto, e é o assunto principal desta dissertação.

Conjectura 3 *Existe uma constante universal $c > 0$ tal que $C^{Al}(n) \geq cn^2$ para todo n.*

Esta conjectura aparece como um problema em Yao [41], e é atribuída a Karp em [37]. Em [41], Yao enunciou um resultado que implica facilmente que $C^{Al}(n) = \Omega(n)$. Entretanto, foi apenas uma década mais tarde que Yao publicou um 'extended abstract' com o esboço da demonstração de um limite inferior superlinear. De fato, o resultado principal em [42] é que $C^{Al}(n) = \Omega\left(n(\log n)^{1/12}\right)$. (O artigo correspondente a este 'abstract' é [44].) Baseada nas técnicas usadas neste trabalho e

acrescentando idéias novas, King [27] melhorou este limite para $\Omega(n^{5/4})$. O melhor resultado conhecido no momento é devido, entretanto, a P. Hajnal [22], que provou que $\mathcal{C}^{\mathcal{A}l}(n) = \Omega(n^{4/3})$.

O nosso objetivo neste trabalho é de expor os resultados de Yao, King, e Hajnal em detalhe, de forma auto-contida mas concisa. Tomamos especial cuidado na forma e precisão com que apresentamos os vários resultados, melhorando assim substancialmente as exposições originais. Esperamos que, através deste trabalho, os leitores interessados na intrigante conjectura de Yao e Karp possam tomar conhecimento rapidamente de essencialmente todas as técnicas que têm se mostrado mais frutíferas em trabalhos ligados a esta conjectura.

Descrevemos agora um pouco mais detalhadamente os resultados de Yao, King e Hajnal, e ao mesmo tempo delineamos o conteúdo deste trabalho. Fixemos uma propriedade monotônica não-trivial P sobre grafos $G \in \mathcal{G}_V$. A definição de $\mathcal{C}^{\mathcal{A}l}(P)$ é bastante complexa, e um teorema fundamental de Yao [41] é usado ao longo desta dissertação para podermos trabalhar com esta quantidade. Yao define a *complexidade determinística média de pior caso* $\mathcal{C}^{\mathcal{D}}(P)$ da propriedade P, ou simplesmente a *complexidade determinística média* de P, como sendo a melhor complexidade média que podemos esperar de um algoritmo deterministico A que determina se um grafo G dado satisfaz ou não a propriedade P, caso admitirmos que o grafo G é escolhido aleatoriamente de acordo com a 'pior' distribuição possível.

O seguinte jogo $J''(n, P)$ captura a noção embutida na definição de $\mathcal{C}^{\mathcal{D}}(P)$. Aqui o Diabinho escolhe uma distribuição α de acordo com a qual ele escolherá os seus grafos G, e fornece ao Alginho esta distribuição. O jogo agora começa com o Diabinho escolhendo um grafo G aleatoriamente de acordo com α. Como em $J'(n, P)$, aqui o Diabinho não pode ir construindo o grafo G ao longo do jogo. Como em $J(n, P)$, o Alginho faz perguntas sucessivas, escolhidas deterministicamente, e tenta descobrir se G satisfaz ou não P. O objetivo do Alginho é fazer, em média, um número pequeno de perguntas escolhendo um algoritmo determinístico apropriado $A = A(\alpha, P)$, enquanto que o objetivo do Diabinho é escolher uma distribuição $\alpha = \alpha(P)$ que force um número médio alto de rodadas. O valor de $\mathcal{C}^{\mathcal{D}}(P)$ é então o número médio de perguntas que o Diabinho consegue forçar o Alginho fazer em $J''(n, P)$, quando ele escolhe a pior distribuição α possível. (A existência de uma *pior* distribuição decorre de um simples argumento de compacidade.)

Baseado num resultado de von Neumann, Yao enunciou em [41] o teorema que afirma que $C^{\mathcal{D}}(P) = C^{\mathcal{AI}}(P)$. Assim, podemos passar a considerar algoritmos determinísticos e mover a aleatoriedade para os dados de entrada. Após um pouco de reflexão este resultado se torna natural, bastando perceber que este nada mais é que uma igualdade minimax. A sua importância está no fato de que temos através dele um método para provar limites inferiores para $C^{\mathcal{AI}}(P)$. De fato, para demonstrar uma desigualdade da forma $C^{\mathcal{AI}}(P) \geq \nu$, basta provar que o Diabinho consegue achar uma distribuição α ruim o suficiente para a qual o Alginho seja forçado a fazer, em média, pelo menos ν perguntas em $J''(n, P)$. Todos os limites inferiores para $C^{\mathcal{AI}}(P)$ provados até o momento usam este método.

Provamos este resultado de Yao no Capítulo 2 como o Teorema 2.2.1, usando o teorema de dualidade forte em programação linear. Neste capítulo também provamos dois outros resultados fundamentais de Yao. O primeiro deles, o Teorema 2.2.2, é um limite inferior simples para $C^{\mathcal{AI}}(P)$ que envolve certos grafos que satisfazem P. Como curiosidade, demonstraremos em seguida o Lema 2.3.1, que usa de maneira bastante simples uma técnica probabilística fundamental. O segundo resultado, o Teorema 2.3.4, concerne $C^{\mathcal{AI}}(P')$ onde P' é uma propriedade genérica de grafos bipartidos. A técnica envolvida na demonstração deste resultado é extremamente engenhosa, e tem um papel crucial nos trabalhos de Yao [42], [44], assim como nos trabalhos subseqüentes de King [27] e Hajnal [22]. Os resultados apresentados neste capítulo e uma *técnica de redução* de problemas sobre propriedades de grafos genéricos para problemas sobre propriedades de grafos bipartidos foram os ingredientes principais da prova do primeiro limite inferior superlinear para $C^{\mathcal{AI}}(n)$, devido a Yao [42], [44].

Cabe aqui uma observação sobre esta técnica de redução mencionada acima. Seguindo Yao, ambos King e Hajnal provam seus limites inferiores para $C^{\mathcal{AI}}(n)$ primeiramente considerando $C^{\mathcal{AI}}(P')$ para propriedades P' de grafos *bipartidos*. Por conveniência, ponhamos $C^{\mathcal{AI}}(n_1, n_2) = \min_{P'} C^{\mathcal{AI}}(P')$, onde o mínimo é tomado sobre todas as propriedades monotônicas não-triviais P' de grafos bipartidos com classes de vértices de cardinalidade n_1 e n_2. Apresentando reduções apropriadas, King e Hajnal provam que limites inferiores para $C^{\mathcal{AI}}(n)$ podem ser obtidos a partir de limites inferiores para $C^{\mathcal{AI}}(\lceil n/2 \rceil, \lfloor n/2 \rfloor)$. As técnicas de redução usadas por estes autores são mais elaboradas que a de Yao, e aí encontramos um dos ingredientes de

seus sucessos na obtenção de melhores limites inferiores. Um outro ingrediente é um melhor limite inferior para $\mathcal{C}^{\mathcal{A}l}(n_1, n_2)$.

O Capítulo 3 é dedicado aos resultados de King [27]. A técnica de redução devida a King é dada no Teorema 3.4.1, e o seu resultado sobre $\mathcal{C}^{\mathcal{A}l}(n_1, n_2)$ é dado no Teorema 3.4.2. O Teorema 2.3.4 de Yao é importante neste capítulo, mas uma nova técnica é também desenvolvida para demonstrar resultados que em espírito se assemelham àquele resultado de Yao. (Veja Lemas 3.3.1 e 3.3.2.) É devido a esta nova técnica que decidimos incluir os trabalho de King aqui, mesmo que o seu limite inferior seja superado pelo de Hajnal.

No Capítulo 4 apresentamos os resultados de Hajnal [22]. A técnica de redução devida a este autor é dada no Teorema 4.4.3. O seu limite inferior para $\mathcal{C}^{\mathcal{A}l}(n_1, n_2)$ é dado no Teorema 4.4.1. Destes dois resultados segue o melhor limite inferior atual para a complexidade aleatória de propriedades monotônicas não-triviais de grafos de ordem n, a saber, $\mathcal{C}^{\mathcal{A}l}(n) = \Omega(n^{4/3})$. São dois os ingredientes novos que permitem a demonstração deste resultado. O primeiro, o Lema 4.1.1, é uma pequena modificacao do Teorema 2.3.4 de Yao que, embora a primeira vista não seja particularmente atraente, mostra-se extremamente útil no resto da prova. O segundo, o Teorema 4.2.4, é um novo resultado de empacotamento de grafos bipartidos que é usado de uma forma bastante não trivial. Observamos que o Teorema 4.2.4, cuja demonstração é bastante simples, é provado não-construtivamente através do 'método probabilístico'. Procuramos nestes quatro primeiros capítulos apresentar os resultados respeitando sua ordem cronológica. Entretanto é possível que o mais conveniente seja ler as demonstração do Teorema 2.3.4 e a do Lema 4.1.1 juntas, uma vez que suas demonstrações são muito parecidas e damos apenas um esboço da demonstração do segundo resultado.

A conjectura de Yao e Karp é num certo sentido negativa, pois ela afirma que a introdução de aleatoriedade não pode reduzir a complexidade computacional de propriedades monotônicas de grafos em termos de ordem de grandeza. Em Yao [41], o problema de achar um exemplo concreto onde algoritmos aleatórios são comprovadamente mais eficientes que algoritmos determinísticos é posto. A parte final deste trabalho é dedicada ao estudo de um tal exemplo.

Claramente, o problema da estimativa do volume de corpos convexos descrito acima é certamente um exemplo extremamente ilustrativo. Temos para este proble-

ma um algoritmo aleatório que se comporta muito melhor que qualquer algoritmo determinístico. Preferimos entretanto estudar um exemplo muito mais simples e básico: estudamos o problema de determinar o mínimo de uma coleção de inteiros. Também consideramos aqui o problema relacionado de ordenação.

Apresentamos no último capítulo quatro resultados. O primeiro é devido a Valiant [39]. Este resultado afirma que qualquer algoritmo determinístico paralelo usando n processadores que determina o mínimo de uma coleção de n inteiros dados tem complexidade $\Omega(\log\log n)$. O segundo resultado que apresentamos é uma versão melhorada do algoritmo paralelo de Reischuk [33] que determina o k-ésimo menor elemento de uma coleção dada de n inteiros, usando n processadores. Esse algoritmo roda em tempo limitado por uma constante universal. Assim temos um exemplo de um problema extremamente simples onde vale que a introdução de aleatoriedade diminui a complexidade do problema em *ordem de grandeza*.

O terceiro resultado naquele capítulo é devido a Yao [41]. Este concerne a dependência da complexidade média de qualquer algoritmo determinístico de seleção ou ordenação com relação à distribuição dos dados de entrada. Este resultado é um dos ingredientes na prova do resultado final que apresentamos, a saber, um resultado de Alon e Azar [4] que dá um limite inferior para a complexidade de algoritmos paralelos aleatórios de ordenação. Devido à complexidade de sua prova, apresentamos apenas uma demonstração parcial do resultado.

Em relação à notação e à terminologia, em geral e da teoria dos grafos em particular, procuramos seguir aquelas padrões. Entretanto sempre que possível procuramos explicitar o significado de cada símbolo ao longo do próprio corpo do trabalho, correndo o risco até talvez de sermos inconvenientemente repetitivos. O Apêndice A é um resumo bastante simples das definições básicas da teoria dos grafos que estaremos usando aqui. O Apêndice B traz uma lista de algumas das notações mais usadas.

Uma observação sobre a notação que talvez seja útil é a seguinte. Dadas expressões a, b, c, denotaremos $a/(bc)$ por a/bc.

Capítulo 1

Definições iniciais

Neste capítulo inicial vamos definir o conceito de propriedade de grafos. Em seguida definiremos propriedades monotônicas não-triviais, que são aquelas a que mais nos dedicaremos nesta dissertação. Estaremos interessados nos capítulos seguintes em estudar a complexidade de tempo de algoritmos que, dado um grafo G, consultando apenas a sua matriz de adjacência, determinam se G satisfaz ou não uma propriedade monotônica não-trivial. Neste caso, a complexidade de tempo será baseada no número de consultas que o algoritmo precisa fazer à matriz de adjacência. (Iremos supor que essas consultas predominam nos algoritmos e ignoraremos outros fatores que numa situação prática afetariam o tempo de execução desses algoritmos.) Para formalizar o conceito de algoritmo e a maneira como vamos medir sua complexidade, definimos árvores de decisão e pré-grafos. As árvores de decisão são amplamente conhecidas nos meios computacionais devido a seu uso no estudo de ordenação de conjuntos; damos aqui a adaptação usual utilizada no estudo de propriedades de grafos (ver [22] e [41]). Os pré-grafos são utilizados no estudo da complexidade determinística de propriedades de grafos (ver [9]); o que fazemos aqui é unir os pré-grafos às árvores de decisão para que possamos ter uma apresentação formal do problema. Na seção final, definiremos os conceitos de complexidade média de pior caso e complexidade aleatória de uma propriedade de grafos devidas a Yao [41]. Essas complexidades refletem a partir de dois pontos de vista distintos a eficiência média de algoritmos que decidem se um grafo dado satisfaz ou não a propriedade em questão. Finalmente, enunciamos a conjectura de Yao e Karp que será o objeto principal de nosso estudo nos primeiros quatro capítulos deste trabalho.

1.1 Propriedades de grafos

Sejam U, V e W conjuntos finitos. Em todo este trabalho, o conjunto V será o conjunto de vértices de um grafo, enquanto que U e W serão as classes de vértices de grafos bipartidos. Além disso, denotaremos por u, v e w as cardinalidades de U, V e W respectivamente. Vamos denotar por $\mathcal{G}_V$ o conjunto de todos os grafos simples não-dirigidos com conjunto de vértices V. Utilizaremos $\mathcal{G}_{U,W}$ para denotar o conjunto dos grafos bipartidos simples não-dirigidos com classes de vértices U e W.

Definição 1.1.1 *(i) Uma* **propriedade** P *sobre elementos de* $\mathcal{G}_V$ *é um subconjunto de* $\mathcal{G}_V$ *invariante por isomorfismo de grafos. Um elemento* G *de* $\mathcal{G}_V$ **tem** *ou* **satisfaz** *a propriedade* P *se e somente se* $G \in P$. *(ii) Uma* **propriedade** P *sobre elementos de* $\mathcal{G}_{U,W}$ *é um subconjunto de* $\mathcal{G}_{U,W}$ *invariante por isomorfismo de grafos bipartidos que respeitam as classes* U *e* W. *Um elemento* G *de* $\mathcal{G}_{U,W}$ **tem** *ou* **satisfaz** *a propriedade* P *se e somente se* $G \in P$.

As propriedades de grafos estudadas neste trabalho serão sempre invariantes por isomorfismo na categoria em questão. Ou mais simplesmente, se um grafo G satisfaz uma propriedade P então toda cópia isomorfa de G satisfaz P.

Seja X um conjunto de grafos. Dada uma propriedade P sobre X e um grafo $G \in X$, vamos estar interessados em descobrir se G satisfaz P. Evidentemente, se todo grafo do conjunto X satisfaz P (ou seja, $P = X$) ou nenhum grafo em X satisfaz P (i.e., $P = \varnothing$), então claramente é trivial descobrir se G satisfaz ou não P. Assim, estaremos interessados em propriedades P tais que pelo menos um grafo de X satisfaz P mas nem todo grafo de X satisfaz P.

Definição 1.1.2 *(i) Uma propriedade* P *sobre* $\mathcal{G}_V$ *é* **não-trivial** *se* $P \neq \varnothing$ *e* $P \neq \mathcal{G}_V$. *(ii) Uma propriedade* P *sobre* $\mathcal{G}_{U,W}$ *é* **não-trivial** *se* $P \neq \varnothing$ *e* $P \neq \mathcal{G}_{U,W}$.

Se G e H são dois grafos, escrevemos $G \subseteq_g H$ se G é um subgrafo gerador de H.

Definição 1.1.3 *Uma propriedade* P *é* **monotônica crescente** *se para quaisquer* G *e* H *tais que* $G \subseteq_g H$ *e* $G \in P$, *temos que* $H \in P$. *Dizemos que* P *é* **monotônica decrescente** *se para quaisquer* G *e* H *tais que* $H \subseteq_g G$ *e* $G \in P$, *temos que* $H \in P$.

Observe que se definirmos a propriedade **complementar** de P como sendo a propriedade P^c tal que $G \in P$ se e somente se $G^c \in P^c$ (onde G^c é o grafo complementar de G), então P é monotônica *decrescente* se e somente se P^c é monotônica *crescente*. Daqui para frente, vamos sempre supor que as propriedades monotônicas que consideramos são crescentes. Não haverá perda de generalidade nos casos que vamos considerar, pois se a propriedade monotônica P é decrescente podemos passar a considerar P^c no seu lugar e derivar os resultados para P a partir dos resultados que obtivermos para P^c. Omitiremos assim o termo crescente.

Neste trabalho estaremos particularmente interessados nas propriedades monotônicas não-triviais. Note que trivialmente propriedades monotônicas são univocamente determinadas pelos seus elementos minimais. Assim é conveniente fazer a seguinte definição.

Definição 1.1.4 *Dada uma propriedade P sobre $\mathcal{G}_V$ ou sobre $\mathcal{G}_{U,W}$, vamos denotar por* $\min(P)$ *o conjunto dos grafos que satisfazem P e são minimais segundo a relação $\subseteq_g$.*

Assim, dada uma propriedade P, o grafo G pertence a $\min(P)$ se e somente se $G \in P$ e, para qualquer subgrafo $H \subseteq_g G$ tal que $A(H) \subsetneq A(G)$, temos $H \notin P$.

Vamos indicar por $\mathcal{P}_V$ o conjunto das propriedades monotônicas não-triviais de grafos com conjunto de vértices V, e por $\mathcal{P}_{U,W}$ o conjunto das propriedades monotônicas não-triviais de grafos bipartidos com classes de vértices U e W.

1.2 Árvores de decisão

Dado um grafo G representado por sua matriz de adjacência, queremos verificar as entradas desta matriz para descobrir se G tem ou não uma certa propriedade P dada. Um algoritmo especificando quais entradas da matriz devem ser verificadas pode ser representado por uma "árvore de decisão".

Definição 1.2.1 *Seja X um conjunto. Uma* **árvore de decisão** *é uma árvore binária com raiz e pelo menos duas folhas tal que*

(i) todo nó interno é rotulado com um par não-ordenado $\{i, j\}$, onde $i, j \in X$;

(ii) todo nó interno tem uma aresta saindo com rótulo 1 e outra com 0;

(iii) em todo caminho da raiz até uma folha, os rótulos de quaisquer dois nós internos são distintos;

(iv) cada folha é rotulada com 0 *ou* 1.

Seja X um conjunto. Uma árvore de decisão T representa um algoritmo $A = A(T)$ da seguinte maneira. Suponha que um grafo G tem conjunto de vértices X, e que G é dado por uma matriz de adjacência $M \in \{0,1\}^{X \times X}$, ou seja $G \in \mathcal{G}_X$. Vejamos como o algoritmo A se comporta quando a entrada é G. Executamos o algoritmo A visitando os nós de T, começando pela raiz, e fazendo uma consulta à matriz M a cada nó interno de T visitado. O rótulo $\{i,j\}$ de um nó interno y de T representa a consulta "i e j são adjacentes?". Assim quando visitamos um nó interno y devemos fazer a pergunta que esse nó representa. As duas arestas que saem de y têm rótulos que correspondem aos dois resultados possíveis dessa consulta (0 se i e j *não* são adjacentes, e 1 se são adjacentes). Visitamos então o filho de y em T que é ligado a y através da aresta com rótulo que está em acordo com o resultado da consulta. O algoritmo A deve prosseguir dessa maneira até atingir uma folha. Para cada grafo $G \in \mathcal{G}_X$, esse procedimento determina um caminho único $C = C(G,T)$ entre a raiz e uma certa folha $f = f(G,T)$. O rótulo dessa folha f é a saída do algoritmo A para G.

Baseados na interpretação acima para T, fazemos $P(T) = \{G \in \mathcal{G}_X$: rótulo da folha $f(G,T)$ é $1\}$ e dizemos que T **computa** $P(T)$. No que segue, confundimos as noções de algoritmo e árvore de decisão; isto é, admitimos que todo algoritmo A que testa se um grafo G tem ou não uma propriedade P é tal que $A = A(T)$ para alguma árvore de decisão T tal que $P = P(T)$.

O caminho $C(G,T)$ acima fornece dois conjuntos: um de arestas que sabemos que G contém e um outro de arestas que sabemos que G não contém. Em geral, podemos pensar nesses conjuntos não apenas para caminhos que começam na raiz e terminam nas folhas, mas também para os caminhos que terminam em nós internos da árvore de decisão. Se um certo nó interno x de T está à distância i da raiz, podemos concluir que para chegar a esse nó o algoritmo A fez i consultas à matriz de adjacência M de G. Associado a esse nó x temos um **pré-conjunto** Y_x que é dado por um par ordenado de conjuntos (S_x, N_x) onde $S_x \cap N_x = \varnothing$ e $|S_x \cup N_x| = i$. O conjunto S_x contém os pares não-ordenados de vértices de G que $A(T)$ descobriu que pertencem a $A(G)$, e o conjunto N_x contém os pares de vértices que $A(T)$ descobriu

que *não* pertencem a $A(G)$. (As definições formais de S_x e N_x são dadas mais adiante.) Definimos então o **pré-grafo** G_x como sendo a tripla ordenada $(V(G); S_x, N_x)$. O pré-grafo G_x é uma imagem parcial de G que o algoritmo tem após fazer a i-ésima pesquisa na matriz de adjacência de G e é baseando-se em G_x que o algoritmo decide se G satisfaz ou não a propriedade P: se G_x contém algum $H \in \min(P)$ (ou mais precisamente, se $A(H) \subseteq S_x$ para algum $H \in \min(P)$), então certamente o algoritmo pode concluir que $G \in P$, já que P é monotônica crescente; se G_x é tal que $N_x \cap A(H) \neq \varnothing$ para todo $H \in \min(P)$, então nenhum grafo de $\min(P)$ pode ser subgrafo de G e portanto $G \notin P$. Se nenhum dos dois casos anteriores é válido, o algoritmo deve prosseguir fazendo consultas à matriz de adjacência.

Em termos de árvore de decisão, temos as definições a seguir. Dado um nó x de uma árvore de decisão A, considere o caminho $C = (x_0, a_0, x_1, a_1, \ldots, x_{i-1}, a_{i-1}, x_i)$ da raiz de A até x, onde x_0 é a raiz e $x_i = x$. Escrevemos S_x para o conjunto dos rótulos dos nós em C que são seguidos em C por uma aresta rotulada com 1, i.e. $S_x = \{\{k,l\} : \text{para algum } 0 \leq j < i,\ \{k,l\} \text{ é rótulo de } x_j \text{ e } a_j \text{ tem rótulo } 1\}$. Denotamos por N_x o conjunto dos rótulos dos nós de C seguidos por arestas 0, ou seja $N_x = \{\{k,l\} : \text{para algum } 0 \leq j < i,\ \{k,l\} \text{ é rótulo de } x_j \text{ e } a_j \text{ tem rótulo } 0\}$. Definimos o **pré-conjunto** Y_x associado a x em A como sendo o par ordenado (S_x, N_x). Note que pelo item (iii) da Definição 1.2.1, temos que $S_x \cap N_x = \varnothing$ e $|S_x \cup N_x| = i$.

Dada uma árvore de decisão A que testa uma certa propriedade, dizemos que **um grafo G está associado a uma folha** f de A se $S \subseteq A(G)$ e $N \cap A(G) = \varnothing$, onde (S, N) é o pré-conjunto Y_f associado a f.

Note que, na notação acima, o grafo G está associado a f se e somente se $f = f(G, T)$. Sendo assim, o lema a seguir é imediato.

Lema 1.2.2 *Dados um grafo G e uma árvore de decisão A, o grafo G está associado a uma única folha de A.*

Demonstração

Suponha que existam duas folhas distintas f_1 e f_2 em A associadas a G. Seja C_1 o caminho em A da raiz de A até f_1 e C_2 o caminho da raiz até f_2. Suponha que x seja o último nó em comum de C_1 e C_2, e que x tenha rótulo $\{i,j\}$. Se G tem a aresta $\{i,j\}$ então ambos C_1 e C_2 usam a aresta que sai de x com rótulo 1; se G não

tem $\{i, j\}$, então os dois caminhos tem que usar a aresta que sai de x com rótulo 0. Mas em ambos os casos isso é uma contradição, pois x era o último nó em comum de C_1 e C_2.

□

Daqui para frente não faremos distinção entre árvore de decisão e algoritmo, e freqüentemente chamaremos um mesmo objeto tanto de árvore de decisão como de algoritmo. Vamos então denotar por $\mathcal{A}_P$ o conjunto de árvores de decisão que computam P.

1.3 Complexidade determinística média de pior caso

Comecemos com a seguinte definição.

Definição 1.3.1 *Dados um grafo G e uma árvore de decisão A, seja $f = f(G, A)$ a folha de A associada a G. Definimos* $\text{custo}(A, G)$ *como sendo o comprimento do caminho de A que vai da raiz de A até f.*

Dado um algoritmo representado por uma árvore de decisão A, podemos interpretar $\text{custo}(A, G)$ como sendo o número de consultas que o algoritmo A precisa fazer à matriz de adjacência do grafo G para descobrir se G tem ou não a propriedade P.

No que segue, consideramos distribuições de probabilidade α sobre $\mathcal{G}_V$, no sentido de que cada α será uma função $\mathcal{G}_V \to [0, 1]$ tal que $\sum_G \alpha(G) = 1$, onde naturalmente a soma se estende sobre $\mathcal{G}_V$. Fixada α, denotaremos por $\mathbb{E}_\alpha$ e $\mathbb{P}_\alpha$ a esperança e a probabilidade no espaço de probabilidades $(\mathcal{G}_V, \alpha)$. Analogamente, vamos considerar também distribuições de probabilidade sobre $\mathcal{G}_{U,W}$; neste caso, $\mathbb{E}_\alpha$ e $\mathbb{P}_\alpha$ serão a esperança e a probabilidade no espaço de probabilidades $(\mathcal{G}_{U,W}, \alpha)$.

Definição 1.3.2 *(i) Dadas uma distribuição α sobre $\mathcal{G}_V$ e uma propriedade P sobre $\mathcal{G}_V$, definimos o* **custo médio** *$C(A, \alpha)$ de um algoritmo $A \in \mathcal{A}_P$ com relação a α como sendo a esperança de* $\text{custo}(A, G)$*, ou seja, $C(A, \alpha) = \mathbb{E}_\alpha(\text{custo}(A, G)) = \sum_{G \in \mathcal{G}_V} \alpha(G)\,\text{custo}(A, G)$. (ii) Analogamente para uma distribuição α sobre $\mathcal{G}_{U,W}$ e uma propriedade P sobre $\mathcal{G}_{U,W}$, definimos o* **custo médio** *$C(A, \alpha)$ de um algoritmo $A \in \mathcal{A}_P$ com relação a α como sendo $C(A, \alpha) = \mathbb{E}_\alpha(\text{custo}(A, G)) = \sum_{G \in \mathcal{G}_{U,W}} \alpha(G)\,\text{custo}(A, G)$.*

Definição 1.3.3 *Definimos a* **complexidade determinística média de pior caso** *de uma propriedade P sobre $\mathcal{G}_V$ (ou respectivamente sobre $\mathcal{G}_{U,W}$) como sendo $C^{\mathcal{D}}(P) = \sup_\alpha \min_{A \in \mathcal{A}_P} C(A, \alpha)$, onde o supremo é tomado sobre todas as distribuições de probabilidade α sobre $\mathcal{G}_V$ (respectivamente, sobre $\mathcal{G}_{U,W}$).*

Talvez a analogia com o jogo $J''(|V|, P)$ definido na introdução facilite a interpretação desta definição. De todo modo, podemos interpretar $C^{\mathcal{D}}(P)$ da seguinte forma. Suponha que queremos um algoritmo que, dado um grafo G, decide se G tem ou não a propriedade P. Ademais, suponha que usaremos este algoritmo com o dado de entrada (i.e. G) escolhido aleatoriamente de acordo com uma distribuição α conhecida. Então, pela definição de $C^{\mathcal{D}}(P)$, sabemos que existe um algoritmo $A = A(\alpha)$ que tem complexidade média no máximo $C^{\mathcal{D}}(P)$. Ainda, decorre da definição de $C^{\mathcal{D}}(P)$ que a distribuição α pode ser 'muito difícil', no sentido de que com esta distribuição de entrada não podemos achar nenhum algoritmo A que tenha complexidade média, relativa a α, menor que $C^{\mathcal{D}}(P)$. Esta segunda observação justifica o termo 'complexidade média de *pior caso*'. No que segue, entretanto, por simpicidade freqüentemente omitiremos a qualificação 'de pior caso', pois isto não causará qualquer confusão.

Podemos pensar numa distribuição α sobre $\mathcal{G}_V$ como um ponto do $\mathbb{R}^n$ onde $n = |\mathcal{G}_V|$ e as coordenadas de α são dadas por $\alpha(G)$ para cada $G \in \mathcal{G}_V$. Neste caso, o conjunto dos pontos α definidos pelas distribuições de probabilidade sobre $\mathcal{G}_V$ é fechado e limitado, pois é simplesmente um $(n-1)$-simplexo regular. Além disso, como $\varphi(\alpha) = \min_A C(A, \alpha)$ é uma função contínua em α, existe uma distribuição α_0 que atinge o valor $\varphi(\alpha_0) = C^{\mathcal{D}}(P)$. Em outras palavras, podemos definir a complexidade determinística média como $C^{\mathcal{D}}(P) = \max_\alpha \min_A C(A, \alpha)$.

1.4 Complexidade aleatória

Seja P uma propriedade.

Definição 1.4.1 *Um* **algoritmo aleatório** *R para testar P é um par ordenado (P, q) onde q é uma distribuição de probabilidade sobre $\mathcal{A}_P$, ou seja q é uma função $q : \mathcal{A}_P \to [0, 1]$ tal que $\sum_{A \in \mathcal{A}_P} q(A) = 1$.*

Um algoritmo aleatório $R = (P, q)$ pode ser interpretado como um algoritmo que tem probabilidade $q(A)$ de se comportar exatamente como A, onde $A \in \mathcal{A}_P$. A

definição mais usual de algoritmo aleatório entretanto é aquela em que o algoritmo (o Alginho) em certos instantes "joga uma moeda" para decidir o que fazer a seguir. É como se neste modelo mais usual, o algoritmo tivesse uma coleção de estratégias que, de acordo com as respotas obtidas até o momento, sugerissem a próxima pergunta a ser feita. O que o algoritmo faz então é sortear uma estratégia nova de tempos em tempos ao longo de sua execução. Mas claramente, podemos codificar esse processo de escolha aleatória de partes da estratégia a ser usada, em uma única escolha aleatória de uma estratégia global. Nesse sentido, o que a Definição 1.4.1 sugere é exatamente um algoritmo aleatório que sorteia uma estratégia logo no começo de sua execução e usa essa estratégia até o final. Portanto, a definição que vamos usar é apenas uma interpretação conveniente do modelo usual de algoritmos aleatórios.

Definição 1.4.2 *Dado um algoritmo aleatório $R = (P, q)$, o* **custo esperado** *de R para um grafo G é $E(R, G) = \mathbb{E}_q(\text{custo}(A, G)) = \sum_{A \in \mathcal{A}_P} q(A)\,\text{custo}(A, G)$.*

Definição 1.4.3 *(i) Seja $P \in \mathcal{P}_V$. A* **complexidade aleatória** *de P é $C^{Al}(P) = \inf_R \max_{G \in \mathcal{G}_V} E(R, G)$. (ii) Analogamente, seja $P \in \mathcal{P}_{U,W}$. A* **complexidade aleatória** *de P é $C^{Al}(P) = \inf_R \max_{G \in \mathcal{G}_{U,W}} E(R, G)$.*

Podemos interpretar $C^{Al}(P)$ como sendo o número médio de perguntas necessário no pior caso (i.e., na pior entrada) para que o 'melhor' algoritmo aleatório R teste P. Para esta definição também vale a observação feita para a complexidade determinística média, no sentido de que existe R_0 tal que $C^{Al}(P) = \max_G E(R_0, G)$, e portanto temos $C^{Al}(P) = \min_R \max_G E(R, G)$. É proveitoso também fazer uma analogia desta definição com o jogo $J'(|V|, P)$ definido na introdução.

Estudaremos limites inferiores para a complexidade aleatória de propriedades monotônicas não-triviais de grafos, tendo em vista a conjectura de Yao e Karp que damos a seguir.

Conjectura 1.4.4 *Para toda $P \in \mathcal{P}_V$, vale que $C^{Al}(P) = \Omega(v^2)$.*

Em virtude da dificuldade de se obter resultados diretamente da definição da complexidade aleatória, provaremos no próximo capítulo que $C^{Al}(P) = C^{D}(P)$ para qualquer propriedade não-trivial P de grafos. Com isso, estaremos na realidade estudando limites inferiores para $C^{D}(P)$, que graças a esse resultado também serão

válidos para $C^{AI}(P)$. Como já foi observado na introdução, todos os limites inferiores para a complexidade aleatória encontrados até hoje foram determinados a partir da complexidade determinística média de pior caso.

Capítulo 2

Primeiras técnicas

Iniciamos este capítulo com a definição de propriedade dual. Apesar de sua simplicidade aparente, será este conceito que permitirá o estabelecimento de uma relação muito estreita entre o problema de determinar a complexidade de propriedades de grafos e um outro problema sobre grafos, envolvendo a noção de empacotamento. O Lema 2.1.3 determina essa relação através da propriedade dual. Apesar de não envolver conceitos sofisticados, a importância desse lema será notável ao longo deste trabalho, pois graças a ele poderemos usar informações valiosas da propriedade dual no estudo da propriedade original. Esse lema fornece de imediato dois corolários, os Lemas 2.1.4 e 2.1.5, que usaremos nos capítulos seguintes. Enunciamos também dois resultados muito interessantes em empacotamentos. O primeiro se deve a Catlin [14] e o segundo, que na verdade é uma versão mais fina do primeiro, foi enunciado e provado por P. Hajnal [22]. Iremos provar parte do lema de Catlin ainda neste capítulo; deixaremos a demonstração da outra parte e a demonstração do resultado de Hajnal para o Capítulo 4, que reúne outros resultados devidos a este mesmo autor.

Na segunda parte deste capítulo, damos quatro resultados fundamentais devidos a Yao. O primeiro é um resultado minimax enunciado por Yao em 1977 [41] como um corolário da teoria dos jogos [32]. O segundo teorema fornece um limite inferior linear imediato para a complexidade aleatória de propriedades monotônicas não-triviais de grafos bipartidos. Os dois últimos resultados deste capítulo são provados através de técnicas probabilísticas. O segundo destes é uma peça crucial nos trabalhos de Yao [42], [44], King [27] e Hajnal [22].

2.1 Dualidade e empacotamento

Uma definição aparentemente ingênua mas que será muito útil é a seguinte. (Lembre que dado um grafo G, denotamos por G^c o seu grafo complementar.)

Definição 2.1.1 *A* **propriedade dual** P^* *de uma propriedade* P *é dada por* $G \in P^*$ *se e somente se* $G^c \notin P$.

Observe que se P é uma propriedade não-trivial monotônica *crescente*, então temos que P^* também é não-trivial monotônica *crescente*. Além disso, seja A uma árvore de decisão para P, e seja P^c a propriedade complementar de P (i.e., $G \in P$ se e somente se $G^c \in P^c$). Podemos obter uma árvore de decisão A^c para a propriedade P^c invertendo os rótulos das duas arestas que saem de cada nó interno de A. Mais ainda, se complementarmos os rótulos das folhas de A^c obteremos uma árvore de decisão A^* para P^*. Existe então uma bijeção natural entre as árvores que computam P e as que computam P^*. Sendo assim, as complexidades de P e P^*, tanto determinísticas como aleatórias, são as mesmas.

Vamos denotar por K_V o grafo completo com conjunto de vértices V, e por $K_{U,W}$ o grafo bipartido completo com classe U e W. Com isso podemos dar a seguinte definição.

Definição 2.1.2 *(i) Sejam dados* G *e* $H \in \mathcal{G}_V$. *Um* **empacotamento** *de* G *e* H *é um par ordenado* (G', H') *onde* G' *e* H' *pertencem a* $\mathcal{G}_V$ *e são respectivamente cópias isomorfas de* G *e* H *em* K_V *tais que* $A(G') \cap A(H') = \emptyset$. *(ii) Sejam dados* G *e* $H \in \mathcal{G}_{U,W}$. *Um* **empacotamento** *de* G *e* H **como grafos bipartidos** *é um par ordenado* (G', H') *onde* G' *e* H' *pertencem a* $\mathcal{G}_{U,W}$ *e são cópias isomorfas de* G *e* H *em* $K_{U,W}$ *que respeitam as classes (i.e., existem isomorfismos* $g : V(G') \to V(G)$ *e* $h : V(H') \to V(H)$ *tais que* g *e* h *restritas a* U *são permutações de* U, *bem como* g *e* h *restritas a* W *são permutações de* W*) tais que* $A(G') \cap A(H') = \emptyset$.

Freqüentemente, vamos dar um empacotamento de grafos G, $H \in \mathcal{G}_V$, simplesmente fornecendo uma função bijetora $f : V(G) \to V(H)$ tal que se $x, y \in V(G)$ e $xy \in A(G)$ então $f(x)f(y) \notin A(H)$. Analogamente, um empacotamento de G, $H \in \mathcal{G}_{U,W}$ como grafos bipartidos será dado por duas funções bijetoras $f : U \to U$ e $g : W \to W$ tais que se $x \in U$, $y \in W$ e $xy \in A(G)$ então $f(x)g(y) \notin A(H)$.

Os conceitos de dualidade e empacotamento vão ser especialmente importantes tendo em vista a simples observação a seguir.

Lema 2.1.3 *(i) Dada uma propriedade $P \in \mathcal{P}_V$, não existe empacotamento de G e H para quaisquer $G \in P$ e $H \in P^*$. (ii) Dada uma propriedade $P \in \mathcal{P}_{U,W}$, não existe empacotamento de G e H como grafos bipartidos para quaisquer $G \in P$ e $H \in P^*$.*

Demonstração

(i) Suponha que exista um empacotamento (G', H') de G e H.

Tome $G'' \in \mathcal{G}_V$ tal que $A(G'') = A(K_V) \setminus A(H')$, isto é $G'' = (H')^c$. Claramente G' é subgrafo de G'', e como P é monotônica crescente, temos que $G'' \in P$. Mas como $G'' = (H')^c$, isto é uma contradição, pois pela definição de P^* temos que $H' \in P^*$ se e somente se $(H')^c \notin P$.

(ii) Podemos provar este item como acima tomando o grafo G'' em $\mathcal{G}_{U,W}$ tal que $A(G'') = A(K_{U,W}) \setminus A(H')$.

□

A partir desse lema, obtêm-se imediatamente os dois resultados a seguir para grafos bipartidos. Seja dada uma propriedade $P \in \mathcal{P}_{U,W}$. Note que se existissem $G \in P$ e $G^* \in P^*$ tais que cada um tivesse pelo menos $u/2$ vértices isolados em U, então poderíamos identificar os vértices isolados de U de G com os não-isolados de U de G^* e vice-versa, obtendo assim um empacotamento de G e G^*, e contradizendo o lema. Podemos formalizar este primeiro resultado como no lema a seguir.

Lema 2.1.4 *Se $P \in \mathcal{P}_{U,W}$, então no máximo uma entre P e P^* contém um grafo em que a classe U tem $u/2$ ou mais vértices isolados.*

Demonstração

Suponha que ambos $G \in P$ e $G^* \in P^*$ tenham pelo menos $u/2$ vértices isolados em U. Seja então $f : V(G) \to V(G^*)$ uma bijeção tal que f restrita a W é a função identidade sobre W e tal que $d_G(x) > 0$ implica $d_{G^*}(f(x)) = 0$ para todo $x \in U \cap V(G)$. Claramente, a função f fornece naturalmente um empacotamento de G e G^* como grafos bipartidos. Mas tal empacotamento não pode existir pelo item (ii) do lema anterior, e essa contradição prova nosso lema.

□

Vamos enunciar agora o segundo resultado.

Lema 2.1.5 *Se $P \in \mathcal{P}_{U,W}$, então P ou P^* contém um grafo com pelo menos $\lfloor u/2 \rfloor$ vértices isolados em U.*

Demonstração

Seja X um conjunto com $\lfloor u/2 \rfloor$ vértices quaisquer de U e tome $Y = U \setminus X$. Seja $G = K_{X,W} + E_Y$. Então $G \in \mathcal{G}_{U,W}$ e G tem $\lceil u/2 \rceil$ vértices isolados em U. Se G satisfaz P, então temos a afirmação. Por outro lado, observe que G^c tem $\lfloor u/2 \rfloor$ vértices isolados. Assim se G não satisfaz P, pela definição da propriedade dual de P temos que G^c satisfaz P^* e obtemos o resultado desejado.

□

Seguem dois resultados sobre empacotamentos que serão úteis posteriormente. O primeiro é o teorema de Sauer e Spencer [38], enunciado pela primeira vez por Catlin [14]. A prova do item (i) será feita como em [9]. A prova do item (ii) bem como a prova do segundo resultado serão feitas no Capítulo 4.

Teorema 2.1.6 *(i) Se $G, H \in \mathcal{G}_V$ e $\Delta(G)\Delta(H) < v/2$, então G e H podem ser empacotados. (ii) Se $G, H \in \mathcal{G}_{U,W}$ e $\Delta_U(G)\Delta_W(H) + \Delta_W(G)\Delta_U(H) \leq \max\{u, w\}$, então G e H podem ser empacotados como grafos bipartidos.*

Demonstração

(i) A afirmação é trivial para $|V| = v \leq 2$, portanto vamos assumir que $v \geq 3$. Suponha que G e H não possam ser empacotados. Encontre G' e H' cópias isomorfas de G e H em K_V, respectivamente, de modo que G' e H' tenham o número mínimo possível de arestas em comum. Note que existe pelo menos uma aresta em comum já que G e H não podem ser empacotados. Sejam $V(G') = \{x_1, x_2, \ldots, x_v\}$, $V(H') = \{y_1, \ldots, y_v\}$ e suponha que identificamos x_i com y_i $(1 \leq i \leq v)$. Suponha ainda que x_1 e x_2 são adjacentes em G' e y_1 e y_2 são adjacentes em H'. Iremos provar então que existe $i > 2$ tal que invertendo as posições de x_2 e x_i, i.e., identificando x_2 com y_i e x_i com y_2, a intersecção dos conjuntos de arestas de G' e H' diminui.

Seja L o conjuntos de índices $l > 2$ para os quais existe j tal que vale que $x_2x_j \in A(G')$ e $y_jy_l \in A(H')$ ou vale que $y_2y_j \in A(H')$ e $x_jx_l \in A(G')$. Temos que

$$|L| \leq \sum_{x_j \in \Gamma_{G'}(x_2)} |\Gamma_{H'}(y_j)| + \sum_{y_j \in \Gamma_{H'}(y_2)} |\Gamma_{G'}(x_j)|.$$

Na primeira somatória estamos incluindo y_2 na contagem e na segunda somatória incluímos x_2. Como todos os elementos de L são estritamente maiores que 2, podemos excluir esses vértices e assim $|L| \leq \sum_{x_j \in \Gamma_{G'}(x_2)} |\Gamma_{H'}(y_j)| + \sum_{y_j \in \Gamma_{H'}(y_2)} |\Gamma_{G'}(x_j)| - 2 \leq 2\Delta(G')\Delta(H') - 2 < v - 2$. Portanto existe i $(3 \leq i \leq v)$ tal que $i \notin L$. Se identificarmos x_2 com y_i e x_i com y_2, temos que x_2 e y_i não têm vizinhos em comum assim como y_2 e x_i também não têm vizinhos em comum. Além disso, a aresta x_1x_2 não mais se sobrepõe sobre uma aresta de H' e portanto a intersecção de $A(G')$ e $A(H')$ diminui de uma aresta, contradizendo a maneira como G' e H' foram escolhidos.

(ii) A demonstração será feita no Capítulo 4.

□

Teorema 2.1.7 *Sejam dados $G, H \in \mathcal{G}_{U,W}$ e suponha que*

(i) $u \leq w \leq 2u$,

(ii) $\overline{d}_U(G)\Delta_W(H) \leq u/3$,

(iii) $\overline{d}_U(H)\Delta_W(G) \leq u/3$,

(iv) $\Delta_U(G), \Delta_U(H) \leq u/54(\log u + \log\log u)$.

Então G e H podem ser empacotados como grafos bipartidos, desde que $u \geq 55$.

A demonstração será feita no Capítulo 4.

□

2.2 Técnicas e resultados básicos

Em seu artigo [41], Yao enunciou o teorema fundamental que afirma que a complexidade aleatória e a complexidade determinística média de qualquer propriedade sobre grafos são iguais. Isto é, para qualquer propriedade P temos $C^{\mathcal{A}l}(P) = C^{\mathcal{D}}(P)$. (Observe que na realidade este resultado é apenas uma interpretação conveniente de um resultado mais genérico que extrapola o contexto aqui considerado.) Lembre que a complexidade determinística média $C^{\mathcal{D}}(P) = \max_\alpha \min_A C(A, \alpha)$ é o melhor custo médio que podemos esperar para um algoritmo determinístico que computa P, se a distribuição dos dados de entrada não é conhecida e assumimos a pior distribuição possível. Ademais, a complexidade aleatória $C^{\mathcal{A}l}(P) = \min_R \max_G E(R, G)$ é o custo mínimo necessário para um algoritmo aleatório computar P. Assim, a

desigualdade $C^{\mathcal{D}}(P) \geq C^{\mathcal{AI}}(P)$ significa que existe um algoritmo aleatório R_0 e uma distribuição de entrada α_0 tais que o custo de R_0 na pior instância G não é maior que o custo médio de qualquer algoritmo determinístico com distribuição de entrada α_0. Por outro lado, a desigualdade $C^{\mathcal{D}}(P) \leq C^{\mathcal{AI}}(P)$ significa que, para qualquer distribuição α dos grafos de entrada e qualquer algoritmo aleatório R, existem um algoritmo determinístico A e uma instância $G = G(R)$ tais que R com entrada G necessita em média de tanto tempo quanto A com as instâncias de entrada distribuídas de acordo com α. A desigualdade que mais utilizaremos ao longo deste trabalho é a segunda. Este resultado de Yao, não surpreendentemente, pode ser provado usando-se uma igualdade minimax apropriada. Yao sugere o uso do teorema minimax de von Neumann para a teoria dos jogos, veja [32]. Vamos dar uma demonstração supondo conhecido o teorema de dualidade forte de programação linear.

Teorema 2.2.1 *Se P é uma propriedade não-trivial sobre $\mathcal{G}_V$ ou sobre $\mathcal{G}_{U,W}$, então $C^{\mathcal{D}}(P) = C^{\mathcal{AI}}(P)$.*

Demonstração

Vamos apenas considerar o caso em que P é uma propriedade sobre $\mathcal{G}_V$. A demonstração para P sobre $\mathcal{G}_{U,W}$ é análoga.

A demonstração se resume em definir um problema de programação linear tal que o valor da solução do seu primal seja igual a $C^{\mathcal{D}}(P)$ e o do dual seja $C^{\mathcal{AI}}(P)$. Usando o Teorema de Dualidade de programação linear, teremos o resultado desejado.

Considere a matriz M indexada por elementos de $\mathcal{A}_P \times \mathcal{G}_V$, com a entrada $M(A, G)$ igual a custo(A, G), onde $A \in \mathcal{A}_P$ e $G \in \mathcal{G}_V$. Como estamos considerando propriedades não-triviais, todas as entradas de M são estritamente maiores que zero e portanto $C^{\mathcal{D}}(P) > 0$. Dada uma distribuição α sobre $\mathcal{G}_V$, vamos interpretá-la como um vetor real indexado por elementos de $\mathcal{G}_V$; cada entrada desse vetor está entre 0 e 1, e a soma de todas as suas entradas é igual a um. Temos então que o produto $M\alpha$ é um vetor indexado por elementos de $\mathcal{A}_P$, onde a posição indexada por $A \in \mathcal{A}_P$ é igual a $C(A, \alpha)$.

Vamos indicar por $\tilde{1}$ e por $\tilde{0}$ os vetores reais de $\mathbb{R}^{\mathcal{A}_P}$ com todas as posições iguais a 1 e a 0 respectivamente. Analogamente, usaremos $\overline{1}$ e $\overline{0}$ para $\mathbb{R}^{\mathcal{G}_V}$. Considere o seguinte problema de programação linear: $s = \min\{\langle x, \overline{1}\rangle : Mx \geq \tilde{1}, x \geq \overline{0}\}$. Note

que como toda entrada de M é maior que zero, este programa é viável. Além disso $0 < s < +\infty$. Vamos provar que s^{-1} é igual a $C^{\mathcal{D}}(P) = \max_{\alpha} \min_{A} C(A, \alpha)$.

Primeiramente, vamos provar que $C^{\mathcal{D}}(P) \leq s^{-1}$. Seja $b = C^{\mathcal{D}}(P) > 0$. Seja α_0 uma distribuição de probabilidade tal que $b = \min_A C(A, \alpha_0)$ e seja $x^* = b^{-1}\alpha_0$. Claramente $x^* \geq \overline{0}$ pois $b > 0$. Como observamos acima, a posição do vetor $M\alpha_0$ indexada por $A \in \mathcal{A}_P$ é igual a $C(A, \alpha_0)$ mas, pela definição de α_0, vale que $b \leq C(A, \alpha_0)$ para qualquer $A \in \mathcal{A}_P$. Logo $M\alpha_0 \geq b\tilde{1}$ e temos que $Mx^* = b^{-1}M\alpha_0 \geq \tilde{1}$. Assim x^* é candidato a solução do problema de programação linear acima e portanto $s \leq \langle x^*, \overline{1} \rangle = b^{-1}\langle \alpha_0, \overline{1} \rangle = b^{-1}$. Concluímos que $C^{\mathcal{D}}(P) = b \leq s^{-1}$.

A prova de que $C^{\mathcal{D}}(P) \geq s^{-1}$ é semelhante à anterior. Como o problema de programação linear acima é limitado, existe x^* satisfazendo $Mx^* \geq \tilde{1}$ e $x^* \geq \overline{0}$ tal que $s = \langle x^*, \overline{1} \rangle > 0$. Defina $\alpha_0 = s^{-1}x^*$. Claramente $\alpha_0 \geq \tilde{0}$. Além disso, $\langle \alpha_0, \overline{1} \rangle = s^{-1}\langle x^*, \overline{1} \rangle = 1$ e temos que α_0 é uma distribuição de probabilidade sobre $\mathcal{G}_V$. Observe que qualquer entrada do vetor $M\alpha_0$ é maior ou igual a s^{-1}, pois $M\alpha_0 = s^{-1}Mx^* \geq s^{-1}\tilde{1}$, mas $\min_A C(A, \alpha_0)$ é igual à entrada do vetor $M\alpha_0$ de valor mínimo e portanto temos $C^{\mathcal{D}}(P) \geq \min_A C(A, \alpha_0) \geq s^{-1}$. Com isso, provamos que $C^{\mathcal{D}}(P) = s^{-1}$.

Considere agora o dual do problema de programação linear acima dado por $t = \max\{\langle \tilde{1}, q \rangle : qM \leq \overline{1}, q \geq \tilde{0}\}$. De forma análoga à demonstração acima para a complexidade determinística média, podemos provar que $C^{\mathcal{AI}}(P) = t^{-1}$. Usando o teorema de dualidade, obtemos que $t = s$ e portanto $C^{\mathcal{AI}}(P) = C^{\mathcal{D}}(P)$.

□

Este teorema nos permite demonstrar o resultado a seguir, que por sua vez fornece um limite inferior linear imediato para a complexidade aleatória das propriedades que estamos estudando. A demonstração do próximo teorema, bem como a de outros resultados a seguir, usa a idéia algorítmica por trás da definição formal de árvores de decisão. Nestas demonstrações, quando possível, evitamos o uso explícito de árvores de decisão para que possamos simplificar a notação envolvida e assim possamos nos concentrar nas suas idéias básicas.

Teorema 2.2.2 *(i) Se $P \in \mathcal{P}_V$ e $G \in \min(P)$, então $C^{\mathcal{AI}}(P) \geq |A(G)|$. (ii) Se $P \in \mathcal{P}_{U,W}$ e $G \in \min(P)$, então $C^{\mathcal{AI}}(P) \geq |A(G)|$.*

Demonstração

(i) Afirmamos que para todo grafo $H \in \min(P)$ e todo algoritmo determinístico $A \in \mathcal{A}_P$ vale que $\text{custo}(A, H) \geq |A(H)|$. De fato, tome $H \in \min(P)$. Suponha que em um dado instante o algoritmo A tenha feito menos de $|A(H)|$ consultas à matriz de adjacência de H e que $H' = (V(H); S', N')$ seja o pré-grafo que A tem até o momento. Evidentemente, o algoritmo A não pode afirmar que $H \notin P$, já que $H \in \min(P)$. Mas para que A possa responder que $H \in P$, é necessário que exista $I \in \min(P)$ tal que $A(I) \subseteq S'$. Mas tal I não existe, pois $S' \subsetneq A(H)$ e $H \in \min(P)$. Sendo assim, qualquer algoritmo determinístico A precisa de pelo menos $|A(H)|$ consultas à matriz de adjacência de H para decidir se H satisfaz ou não P.

Pelo Teorema 2.2.1 anterior, temos $\mathcal{C}^{\mathcal{AI}}(P) = \max_\alpha \min_{A \in \mathcal{A}_P} C(A, \alpha)$, onde o máximo é tomado sobre todas as distribuições α sobre $\mathcal{G}_V$. Seja dado um grafo $G \in \min(P)$. Defina a distribuição β sobre $\mathcal{G}_V$ pondo

$$\beta(J) = \begin{cases} 1 & \text{se } J = G \\ 0 & \text{caso contrário.} \end{cases}$$

Para qualquer $A \in \mathcal{A}_P$, temos que

$$C(A, \beta) = \sum_{J \in \mathcal{G}_V} \beta(J)\,\text{custo}(A, J) = \text{custo}(A, G) \geq |A(G)|.$$

Assim, temos que $\mathcal{C}^{\mathcal{D}}(P) \geq |A(G)|$ e portanto $\mathcal{C}^{\mathcal{AI}}(P) \geq |A(G)|$.

(ii) Podemos provar este resultado modificando a demonstração do item (i), considerando as definições para $\mathcal{G}_{U,W}$ ao invés de $\mathcal{G}_V$.

□

Concluímos imediatamente do teorema anterior que $\mathcal{C}^{\mathcal{AI}}(P) \geq v\overline{d}(G)/2$ para toda propriedade $P \in \mathcal{P}_V$ e todo $G \in \min(P)$, onde $\overline{d}(G) = 2|A(G)|/v$ é o grau médio de G. Analogamente temos que $\mathcal{C}^{\mathcal{AI}}(P) \geq u\overline{d}_U(G)$ para toda $P \in \mathcal{P}_{U,W}$ e todo $G \in \min(P)$, onde $\overline{d}_U(G)$ é dado por $|A(G)|/u$. Além disso, como observado anteriormente, vale que $\mathcal{C}^{\mathcal{AI}}(P) = \mathcal{C}^{\mathcal{AI}}(P^*)$, e assim temos também que $\mathcal{C}^{\mathcal{AI}}(P) \geq |A(H)|$ para qualquer $H \in \min(P^*)$.

Podemos determinar facilmente um limite inferior trivial de $\max\{u/2, w/2\}$ para a complexidade aleatória de qualquer propriedade em $\mathcal{P}_{U,W}$, lembrando que $|U| = u$ e $|W| = w$. De fato, seja $P \in \mathcal{P}_{U,W}$ uma propriedade dada. Mostremos

que $C^{Al}(P) \geq u/2$. Se um grafo de $\min(P)$ tiver $u/2$ ou mais arestas, então o item (ii) do Teorema 2.2.2 anterior nos fornece o resultado desejado; caso contrário, todos os grafos de $\min(P)$ terão pelo menos $u/2$ vértices isolados, mas neste caso o Lema 2.1.4 garante que todo grafo de $\min(P^*)$ não têm essa quantidade de vértices isolados. Usando novamente o Teorema 2.2.2 (ii) para $G^* \in \min(P^*)$, concluímos que $C^{Al}(P) = C^{Al}(P^*) \geq |A(G^*)| \geq u/2$.

O seguinte exemplo ilustra uma outra aplicação elementar dos resultados que temos até o momento. Seja $2 \leq r \leq v = |V|$, e defina a propriedade $P_r \subset \mathcal{G}_V$ pondo $G \in P_r$ se e somente se G contém um clique com pelo menos r vértices. Ou seja $P_r = \{G \in \mathcal{G}_V : K_r \subseteq G\}$. Note que para $2 \leq r \leq v$, a propriedade P_r assim definida é monotônica não-trivial. Claramente $\min(P_r)$ é o conjunto com todos os grafos de $\mathcal{G}_V$ que têm exatamente um clique de tamanho r e mais nenhuma aresta, e portanto têm $\binom{r}{2}$ arestas. Além disso defina o grafo $G^* = E_{r-2} + K_{v-r+2}$, ou seja o grafo que contém $r-2$ vértices isolados e um clique de ordem $v-r+2$. Então não é difícil notar que G^* assim definido pertence a $\min(P_r^*)$, pois para qualquer aresta $a \in A(G^*)$ que retirarmos de G^*, teremos que $G^* \setminus \{a\}$ não pertence a P_r^* uma vez que $(G^* \setminus \{a\})^c$ pertence a P_r. Assim se $r \geq v/2$, pelo Teorema 2.2.2, temos que $C^{Al}(P_r) \geq \binom{r}{2} = \Omega(v^2)$. Por outro lado, se $r < v/2$, temos novamente pelo Teorema 2.2.2 que $C^{Al}(P_r) = C^{Al}(P_r^*) \geq |A(G^*)| = \Omega(v^2)$. Portanto para todo $2 \leq r \leq v$, vale que

$$C^{Al}(P_r) = \Omega(v^2).$$

Com um argumento análogo, podemos ver que a propriedade 'ter número cromático pelo menos r' ($2 \leq r \leq v$) também tem complexidade aleatória igual a $\Omega(v^2)$. A importância desses dois resultados pode ganhar relevância se notarmos que, no caso determinístico, a elusividade dessas propriedades (lembrando que uma proprieade é elusiva se no pior caso são necessárias $\binom{v}{2}$ perguntas para decidir se um grafo satisfaz ou não essa propriedade) é particularmente difícil de ser provada. A elusividade dessas duas propriedades foi provada numa demonstração específica, devida a Bollobás (ver [9] e [12]).

2.3 Dois resultados probabilísticos

Nesta seção provaremos dois resultados que usam técnicas probabilísticas nas suas demonstrações. O primeiro é extremamente simples e parte do princípio de que se a probabilidade de um certo evento é positiva, então certamente existe um elemento do espaço de probabilidade considerado que satisfaz tal evento. O segundo teorema encontra um limite inferior para $C^{\mathcal{A}l}(P)$ (ou melhor para $C^{\mathcal{D}}(P)$ e conseqüentemente para $C^{\mathcal{A}l}(P)$) para toda $P \in \mathcal{P}_{U,W}$, calculando a média de consultas que qualquer algoritmo em $\mathcal{A}_P$ precisa fazer se os grafos de entrada forem elementos aleatórios de um certo conjunto a ser especificado. O interesse maior da demonstração reside na maneira bastante sofisticada como esse conjunto de grafos é determinado. Ademais, este resultado e a técnica introduzida aqui serão muito importantes nos Capítulos 3 e 4.

O nosso primeiro resultado, que é apenas uma amostra do uso de técnicas probabilísticas, é o seguinte.

Lema 2.3.1 *(i) Seja $P \in \mathcal{P}_V$. Se $G \in P$ e $H \in P^*$, então $|A(G)|\,|A(H)| \geq \binom{v}{2}$. (ii) Seja $P \in \mathcal{P}_{U,W}$. Se $G \in P$ e $H \in P^*$, então $|A(G)|\,|A(H)| \geq uw$.*

Demonstração

(i) Suponha por absurdo que $|A(G)|\,|A(H)| < \binom{v}{2}$. Pegue aleatoriamente um grafo $G' \in \mathcal{G}_V$ isomorfo a G, de forma que todos os grafos isomorfos a G tenham a mesma probabilidade. Observe que $G' \in P$ pois P é invariante sob isomorfismos. Tome uma aresta $a \in A(G')$, e note que a probabilidade de a pertencer a $A(H)$ é igual a $|A(H)|/\binom{v}{2}$. Logo

$$\mathbf{P}\{|A(G') \cap A(H)| \geq 1\} \leq \mathbf{E}\{|A(G') \cap A(H)|\} = \frac{|A(G')|\,|A(H)|}{\binom{v}{2}} < 1.$$

Ou seja, existe um grafo $G'' \in \mathcal{G}_V$ isomorfo a G tal que $A(G'') \cap A(H) = \varnothing$. Neste caso G'' e H podem ser empacotados, mas isso contradiz o Lema 2.1.3, pois $H \in P^*$ e $G'' \in P$.

(ii) Claramente, uma demonstração análoga à do caso (i) vale aqui. Bastando substituituir $\binom{v}{2}$ por uw.

□

Passemos agora ao segundo resultado desta seção. Vamos primeiramente dar uma definição inicial e provar um lema probabilístico auxiliar.

Seja Y um conjunto de vértices de um grafo $G \in \mathcal{G}_{U,W}$. Suponha que Y tenha y vértices. Escrevemos $\mathbf{d}_Y(G)$ para a seqüência $\langle d_1, d_2, \ldots, d_y \rangle$ não-crescente de graus dos vértices de Y em G. Isto é, se $Y = \{x_1, \ldots, x_y\}$ e $d_G(x_1) \geq d_G(x_2) \geq \ldots \geq d_G(x_y)$, então $\mathbf{d}_Y(G) = \langle d_G(x_1), d_G(x_2), \ldots, d_G(x_y) \rangle$. Com isso vamos definir uma quase-ordem total sobre os grafos de $\mathcal{G}_{U,W}$.

Definição 2.3.2 *Dados grafos $G, H \in \mathcal{G}_{U,W}$, dizemos que $G \preccurlyeq_U H$ se e somente se a seqüência $\mathbf{d}_U(G)$ for menor ou igual à seqüência $\mathbf{d}_U(H)$ em ordem lexicográfica. Analogamente, definimos uma quase-ordem total $\preccurlyeq_W$ com relação à classe W.*

Sejam $1 \leq m \leq N$ inteiros dados. Considere $[N]^{(m)} = \{M \subseteq [N] : |M| = m\}$, o conjunto dos m-subconjuntos de $[N] = \{1, 2, \ldots, N\}$, como um espaço de probabilidades, supondo que todos os seus elementos M são equiprováveis. Então $Y = Y(M) = \min M$ é uma variável aleatória, e estaremos interessados em $\mathbf{E}(Y)$ a seguir. O seguinte lema é imediato.

Lema 2.3.3 *Se $1 \leq m \leq N$ e $M \in [N]^{(m)}$ é um m-subconjunto aleatório de $[N]$, então a variável aleatória $Y = Y(M) = \min M$ é tal que*

(i) $\mathbf{P}(Y = k) = \binom{N-k}{m-1} / \binom{N}{m}$ *para todo* $1 \leq k \leq N$,

(ii) $\mathbf{E}(Y) = (N+1)/(m+1)$.

Demonstração

A afirmativa (i) é imediata. Para verificar (ii), precisamos primeiro calcular a soma $S = S_{m,N} = \sum_{1 \leq k \leq N} k\binom{N-k}{m-1}$. Considere as funções geradoras $f(z)$ e $g(z)$ das seqüências $\langle 0, 1, 2, \ldots \rangle$ e $\langle \binom{0}{m-1}, \binom{1}{m-1}, \binom{2}{m-1}, \ldots \rangle$. (Veja [19].) Assim

$$f(z) = \frac{z}{(1-z)^2} = \sum_{k \geq 0} kz^k \quad \text{e} \quad g(z) = \frac{z^{m-1}}{(1-z)^m} = \sum_{l \geq 0} \binom{l}{m-1} z^l.$$

Ademais, claramente S é o coeficiente $[z^N]h(z)$ de z^N em $h(z) = f(z)g(z)$. Temos assim $S = [z^N]f(z)g(z) = [z^N]z^m/(1-z)^{m+2} = \binom{N+1}{m+1}$.

Daí temos $\mathbf{E}(Y) = \sum_{1 \leq k \leq N} k\binom{N-k}{m-1} / \binom{N}{m} = \binom{N}{m}^{-1} S = (N+1)/(m+1)$, como queríamos verificar.

□

O Teorema 2.3.4 a seguir, tirado de [27], é um resultado fundamental essencialmente devido a Yao [44], que provou o primeiro limite inferior superlinear para $c^{Al}(P)$ com $P \in \mathcal{P}_V$ usando, entre outros, os Teoremas 2.2.1, 2.2.2 e 2.3.4. A importância do Teorema 2.3.4 também pode ser notada nas demonstrações dos limites inferiores para propriedades de grafos bipartidos obtidas por King e P. Hajnal.

Note que um grafo $G \in \min(P)$ é minimal com relação a $\preccurlyeq_U$ se e somente se $H \preccurlyeq_U G$ e $H \in \min(P)$ implicam $G \preccurlyeq_U H$, ou seja $\mathbf{d}_U(G) = \mathbf{d}_U(H)$. Dado um grafo G, vamos denotar por $\overline{d}(G)$ o grau médio dos vértices de G e por $\Delta(G)$ o seu grau máximo. Além disso, para $\varnothing \neq S \subseteq V(G)$, tomamos $\Delta_S(G) = \max\{d_G(x) : x \in S\}$ e $\overline{d}_S(G) = |S|^{-1} \sum_{x \in S} d_G(x)$.

Teorema 2.3.4 *Seja dada uma propriedade $P \in \mathcal{P}_{U,W}$, e seja G um grafo de $\min(P)$ minimal com relação à quase-ordem $\preccurlyeq_U$. Então*

$$c^{Al}(P) \geq \max\left\{\frac{u}{2}, \frac{u}{2} \cdot \frac{\Delta_U(G) - 4\overline{d}_U(G) + 1}{4\overline{d}_U(G) + 1}\right\}.$$

Demonstração

Por simplicidade, ponha $\Delta = \Delta_U(G)$ e $\overline{d} = \overline{d}_U(G)$. Note que $(\Delta - 4\overline{d} + 1)/(4\overline{d} + 1) \geq 1$ se e somente se $\Delta \geq 8\overline{d}$. Assim analisamos os seguintes dois casos.

Caso 1. Vale que $\Delta \leq 8\overline{d}$.

Note que neste caso $(u/2)(\Delta - 4\overline{d} + 1)/(4\overline{d} + 1) \leq u/2$, e assim é suficiente mostrar que $c^{Al}(P) \geq u/2$. (Já demos a demonstração desse limite inferior como uma conseqüência imediata do Teorema 2.2.2. Vamos apenas repetir aqui os seus passos.) Pelo Teorema 2.2.2 (ii), se $|A(G)| \geq u/2$ temos o resultado. Assuma portanto que $|A(G)| < u/2$. Então claramente G tem pelo menos $u/2$ vértices isolados em U. Assim, pelo Lema 2.1.4, se $H \in \min(P^*)$ temos que $|A(H)| \geq u/2$. Mas então $c^{Al}(P) = c^{Al}(P^*) \geq |A(H)| \geq u/2$.

Caso 2. Temos que $\Delta > 8\overline{d}$.

Suponha que $U = \{x_0, x_1, \ldots, x_{u-1}\}$ e que $d(x_1) \leq d(x_2) \leq \ldots \leq d(x_{u-1}) \leq d(x_0) = \Delta$. Ponha $\Gamma_i = \Gamma_G(x_i)$ para $0 \leq i < u$ e seja $u' = \lceil u/2 \rceil$.

Temos que $d(x_i) < 2\overline{d}$ para $1 \leq i \leq u'$. De fato, basta observar que se $d(x_i) \geq 2\overline{d}$, para algum i $(1 \leq i \leq u')$, então pela escolha dos x_i $(1 \leq i \leq u')$ existem pelo menos $1 + \lfloor u/2 \rfloor$ vértices em U com grau maior ou igual a $2\overline{d}$. Mas então $|A(G)| \geq (1 + \lfloor u/2 \rfloor)2\overline{d} > u\overline{d} = |A(G)|$, obtendo-se assim uma contradição.

Definimos um grafo G' incluindo as seguintes arestas em G (se elas ainda não existirem): ligue x_i a todos os vértices de Γ_0 e de Γ_{i+1}, para $0 \leq i < u'$, e ligue $x_{u'}$ a todos os vértices de Γ_0. Note que assim $\Gamma_{G'}(x_i) = \Gamma_0 \cup \Gamma_i \cup \Gamma_{i+1}$ se $0 \leq i < u'$, e $\Gamma_{G'}(x_{u'}) = \Gamma_0 \cup \Gamma_{u'}$.

O grafo G' tem a propriedade P pois $G \subseteq G'$. Mais do que isso, observe que G' tem várias cópias isomorfas de G. De fato, considere σ_i a permutação cíclica $(0, 1, 2, \ldots, i)$, i.e., a permutação sobre $\{0, 1, \ldots, u-1\}$ tal que $\sigma_i(j) = j+1$ se $0 \leq j < i$, $\sigma_i(i) = 0$, e $\sigma_i(j) = j$ se $j > i$. Seja $G_i \in \mathcal{G}_{U,W}$ o grafo em que $\Gamma_{G_i}(x_j) = \Gamma_{\sigma_i(j)} = \Gamma_G(x_{\sigma_i(j)})$ para todo $0 \leq j < u$. Então $G_i \subseteq G'$, e G_i e G são isomorfos como grafos bipartidos. Ponha $m = \lfloor 4\overline{d} \rfloor$ e note que se $E_i = \Gamma_0 \setminus (\Gamma_i \cup \Gamma_{i+1})$ para $0 \leq i < u'$, então $m = \lfloor 4\overline{d} \rfloor \leq \Delta - 2\lfloor 2\overline{d} \rfloor \leq |E_i|$, para $0 \leq i < u'$. Ponha $E_{u'} = \Gamma_0 \setminus \Gamma_{u'}$. Claramente $m \leq |E_{u'}|$.

Seja $\Psi_i = \Psi_{G'}(x_i) = \{x_i y : y \in E_i\}$ $(0 \leq i \leq u')$ o conjunto das arestas com uma ponta em x_i e a outra em E_i. Definimos agora um subgrafo aleatório $H \subseteq G'$ a partir de G' apagando aleatoriamente m arestas em cada Ψ_i $(0 \leq i \leq u')$. Seja $W_i = W_i(H)$ o conjunto das arestas retiradas que são incidentes a x_i. Então W_i é um subconjunto aleatório de Ψ_i de cardinalidade m, onde todos esses m-subconjuntos W_i são equiprováveis e são escolhidos independentemente. Dessa maneira, o grafo aleatório H é tal que: (i) $H \notin P$ e (ii) para todo i $(0 \leq i \leq u')$ vale que $H \cup W_i \in P$. Provemos (i). Todo vértice x_i $(0 \leq i \leq u')$ tem grau menor que Δ em H, e os outros vértices x_j $(u' < j \leq u)$ têm o mesmo grau que em G, e assim concluímos que $H \preccurlyeq_U G$ e, mais do que isso, observamos que $H' \preccurlyeq_U G$ e $G \not\preccurlyeq_U H'$ para todo $H' \subseteq H$. Mas G é um grafo minimal de $\min(P)$ com relação a $\preccurlyeq_U$, e assim nenhum subgrafo gerador de H pertence a $\min(P)$ e portanto $H \notin P$. Para (ii), basta observar que G_i, como definido acima a partir de $\sigma_i = (0, 1, \ldots, i)$, é uma cópia de G em $H \cup W_i$.

Por (i) e (ii), temos que se entrarmos com H numa árvore de decisão A que computa P, então A tem que testar pelo menos uma aresta de cada W_i $(0 \leq i \leq u')$ e descobrir que elas não estão em H. De fato, mesmo que A fizesse perguntas sobre todas as arestas de H menos as em W_j para algum $0 \leq j \leq u'$, ainda assim sua resposta dependeria de W_j, já que H não satisfaz P, mas $H \cup W_j$ satisfaz. Formalmente, se num dado instante o pré-grafo que A conhece é $(V(H); S, N)$ e $N \cap W_j = \varnothing$ para algum $0 \leq j \leq u'$, então A não pode concluir que o grafo de entrada

não satisfaz P pois ele poderia ser $H \cup W_j \in P$. Assim, se $(V(H); S_f, N_f)$ é o prégrafo associado à folha $f = f(H, A)$, temos que $N_f \cap W_i \neq \varnothing$ para todo $0 \leq i \leq u'$, como afirmamos acima.

Lembre que $H \subseteq G'$ acima foi gerado aleatoriamente a partir de G'. Seja $\mathcal{H}$ o conjunto dos subgrafos H de G' que podem ser obtidos a partir de G' pelo método acima. Defina uma distribuição de probabilidade β sobre $\mathcal{G}_{U,W}$ pondo

$$\beta(J) = \begin{cases} 1/|\mathcal{H}| & \text{se } J \in \mathcal{H} \\ 0 & \text{caso contrário.} \end{cases}$$

Já sabemos pelo Teorema 2.2.1 que $C^{\mathcal{A}l}(P) = \max_\alpha \min_A C(A, \alpha)$, e assim certamente $C^{\mathcal{A}l}(P) \geq \min_A C(A, \beta)$. Para concluirmos a demonstração, basta provar que para todo $A \in \mathcal{A}_P$ temos que $C(A, \beta) = \mathsf{E}_\beta(\text{custo}(A, J)) \geq (u/2)(\Delta - 4\overline{d}+1)/(4\overline{d}+1)$. Seja portanto $A \in \mathcal{A}_P$ um algoritmo fixo e estimemos $C(A, \beta) = \mathsf{E}_\beta(\text{custo}(A, J))$. Lembre que E_β indica a esperança com relação a β, i.e., no espaço $(\mathcal{G}_{U,W}, \beta)$.

Seja $J \in \mathcal{G}_{U,W}$ e suponha que executamos A com entrada J. Lembre que $W_i \subseteq \Psi_{G'}(x_i)$ $(0 \leq i \leq u')$. No que segue queremos determinar o número médio de perguntas que A precisa fazer sobre arestas em $\Psi_{G'}(x_i)$ até fazer a primeira pergunta sobre alguma aresta em W_i. Com isso e com o fato acima que A precisa fazer pelo menos uma pergunta sobre cada W_i $(0 \leq i \leq u')$, encontraremos o limite inferior desejado. Assim, suponha que $k \geq 1$ é o menor inteiro tal que a k-ésima consulta à matriz de adjacência de J por A sobre um par $\{x_i, y\}$ em $\Psi_{G'}(x_i)$ é tal que $\{x_i, y\} \in W_i(J)$. Então pomos $Z_{i,A}(J) = k$. Note que se $Z(J) = \sum_{0 \leq i \leq u'} Z_{i,A}(J)$, então temos que $\text{custo}(A, J) \geq Z(J)$. Assim $Z = Z(J)$, $Z_i = Z_{i,A} = Z_{i,A}(J)$ $(1 \leq i \leq u')$ são variáveis aleatórias sobre $(\mathcal{G}_{U,W}, \beta)$ tais que $C(A, \beta) = \mathsf{E}_\beta(\text{custo}(A, J)) \geq \mathsf{E}_\beta(Z) = \sum_{1 \leq i \leq u'} \mathsf{E}_\beta(Z_i)$. É fácil verificar que Z_i tem a mesma distribuição que a variável aleatória Y do Lema 2.3.3, com $N = |\Psi_{G'}(x_i)| \geq \Delta - 4\overline{d}$. Assim temos que

$$C(A, \beta) \geq u' \frac{|\Psi_{G'}(x_i)| + 1}{m + 1} \geq \frac{u}{2} \cdot \frac{\Delta - 4\overline{d} + 1}{4\overline{d} + 1},$$

completando a nossa demonstração.

□

Note que vale um resultado análogo se considerarmos a classe W ao invés de U.

Capítulo 3

Técnicas utilizando grau máximo de grafos

O produto final deste capítulo será um limite inferior de $\Omega(v^{5/4})$ para a complexidade aleatória de propriedades monotônicas não-triviais de grafos com v vértices. Entretanto esse não é o principal objetivo deste capítulo, já que, como veremos no Capítulo 4, o melhor limite inferior conhecido atualmente é $\Omega(v^{4/3})$. Queremos neste capítulo estudar algumas das técnicas introduzidas por King e outras introduzidas por Yao e revisadas por King, pois elas serão essenciais para a compreensão dos resultados de P. Hajnal, dados no próximo capítulo. Além disso, acreditamos que algumas das técnicas aqui apresentadas poderão ser o ponto de partida para outros resultados futuros.

Inicialmente, consideramos a redução do problema de determinar a complexidade aleatória de uma propriedade P ao problema de determinar a complexidade aleatória de outra propriedade P', onde P e P' são ambas monotônicas não-triviais. Dentre essas reduções se destaca a redução de propriedades de grafos genéricos a de grafos bipartidos, essencialmente devida a Yao [44]. Em seguida, provaremos dois teoremas, dos quais o Teorema 3.3.3 é especialmente interessante, pois fornece um limite inferior imediato para a complexidade de um conjunto considerável de propriedades de grafos. Finalmente, encerraremos o capítulo demonstrando o limite inferior de $\Omega(v^{5/4})$.

3.1 Preliminares

Dado um grafo G, vamos denotar por $|G|_+$ o número de vértices de grau positivo em G.

Definição 3.1.1 *Dada uma propriedade P, definimos* $c_1(P) = \min\{|G|_+ : G \in P\}$ *e* $c_0(P) = \min\{|G|_+ : G \in P^*\}$.

Ou seja, denotamos por $c_1(P)$ a ordem do menor clique que contém todas as arestas de algum grafo em P. Analogamente, tomamos $c_0(P)$ como sendo a ordem do menor clique que contém todas as arestas de algum grafo em P^*. Claramente, temos que $c_1(P) = c_0(P^*)$ e $c_0(P) = c_1(P^*)$.

Para uma propriedade monotônica crescente P, podemos notar o seguinte. Se acharmos um clique de ordem $c_1(P)$ em um grafo G então certamente G satisfaz P. Por outro lado, se acharmos em G um conjunto independente de cardinalidade $c_0(P)$, então G não satisfaz P já que o seu complemento satisfaz P^*.

3.2 Propriedades auxiliares

Vamos definir aqui propriedades que vão auxiliar na redução de problemas de complexidade de propriedades sobre $\mathcal{G}_V$ a problemas de complexidade de propriedades sobre outros conjuntos de grafos.

Lembre que dada uma propriedade P, denotamos por $\mathcal{A}_P$ o conjunto das árvores de decisão, como na Definição 1.2.1, que computam P. No que segue, estaremos procurando limites inferiores para propriedades monotônicas não-triviais P sobre o conjunto Y de grafos, onde Y será $\mathcal{G}_V$ ou $\mathcal{G}_{U,W}$. Grosseiramente falando, iremos tomar um conjunto X de grafos tal que todo $G \in X$ pode ser facilmente transformado num grafo $G' \in Y$. Definiremos então uma propriedade P' sobre X tal que $G \in P'$ se e somente se $G' \in P$. Intuitivamente, deve ser claro que qualquer algoritmo que compute P também computa P', bastando fazer algumas modificações: mais especificamente, se temos $G \in X$ e queremos saber se G satisfaz P', basta submeter G' (o grafo correspondente a G em Y) a algum algoritmo $A \in \mathcal{A}_P$. O lema a seguir prova que nessas condições temos que a complexidade aleatória de P' é menor ou igual à complexidade aleatória de P.

Lema 3.2.1 *Sejam X e Y dois conjuntos de grafos tais que $|X| \leq |Y|$. Seja $f : X \to Y$ uma função injetora. Se P é uma propriedade sobre X, e Q é uma propriedade sobre Y tais que existe uma função $g : \mathcal{A}_Q \to \mathcal{A}_P$ tal que* custo$(g(B), G) \leq$ custo$(B, f(G))$ *para todo $G \in X$ e todo $B \in \mathcal{A}_Q$, então $C^{\mathcal{A}}(P) \leq C^{\mathcal{A}}(Q)$.*

Demonstração

Tome uma distribuição β sobre X tal que $C^{\mathcal{D}}(P) = \min C(A, \beta)$, onde o mínimo é tomado sobre todos os algoritmos $A \in \mathcal{A}_P$. Conside γ a distribuição sobre Y que corresponde a β da maneira natural, i.e.

$$\gamma(H) = \begin{cases} \beta(G) & \text{se existe } G \in X \text{ tal que } H = f(G) \\ 0 & \text{se não existe } G \in X \text{ tal que } H = f(G). \end{cases}$$

Pelo Teorema 2.2.1, temos que $C^{\mathcal{A}l}(Q) = C^{\mathcal{D}}(Q)$ e que $C^{\mathcal{A}l}(P) = C^{\mathcal{D}}(P)$. Além disso, temos que $C^{\mathcal{D}}(Q) \geq \min_{B \in \mathcal{A}_Q} C(B, \gamma)$ e que $C^{\mathcal{D}}(P) = \min_{A \in \mathcal{A}_P} C(A, \beta) \leq \min_{B \in \mathcal{A}_Q} C(g(B), \beta)$. Mas por hipótese custo$(B, f(G)) \geq$ custo$(g(B), G)$ para todo $G \in X$ e todo $B \in \mathcal{A}_Q$, e portanto fixando um algoritmo $B \in \mathcal{A}_Q$ temos que

$$\begin{aligned} C(B, \gamma) &= \mathbb{E}_\gamma(\text{custo}(B, H)) = \sum_{H \in Y} \gamma(H)\,\text{custo}(B, H) \\ &= \sum_{G \in X} \gamma(f(G))\,\text{custo}(B, f(G)) = \sum_{G \in X} \beta(G)\,\text{custo}(B, f(G)) \\ &\geq \sum_{G \in X} \beta(G)\,\text{custo}(g(B), G) = \mathbb{E}_\beta(\text{custo}(g(B), G)) \\ &= C(g(B), \beta). \end{aligned}$$

Segue que $\min_{B \in \mathcal{A}_Q} C(B, \gamma) \geq \min_{B \in \mathcal{A}_Q} C(g(B), \beta)$ e portanto $C^{\mathcal{A}l}(Q) \geq C^{\mathcal{A}l}(P)$. □

No que segue, daremos dois exemplos de reduções que podem ser usadas no estudo da complexidade de propriedades em $\mathcal{P}_V$.

Definição 3.2.2 *Seja $P \in \mathcal{P}_V$. Fixamos conjuntos U e W tais que $V = U \cup W$ e $U \cap W = \varnothing$. Dado um grafo $G \in \mathcal{G}_{U,W}$, tome $\tilde{G} \in \mathcal{G}_V$ como sendo o grafo que se obtém adicionando a G todas as arestas com ambas as pontas em W, ou seja $\tilde{G} = G \cup K_W$. Defina $\tilde{P} \subseteq \mathcal{G}_{U,W}$ tal que $G \in \tilde{P}$ se e somente se $\tilde{G} \in P$.*

Definição 3.2.3 *Seja $P \in \mathcal{P}_V$. Fixamos conjuntos U e W tais que $V = U \cup W$ e $U \cap W = \emptyset$. Dado $G \in \mathcal{G}_U$, tome $\hat{G}$ como sendo a união disjunta de G e K_W com todas as arestas entre G e K_W, ou seja $\hat{G} = G \times K_W = (G + K_W) \cup K_{U,W}$. Defina $\hat{P} \subseteq \mathcal{G}_U$ tal que $G \in \hat{P}$ se e somente $\hat{G} \in P$.*

Com essas definições os dois lemas a seguir são triviais.

Lema 3.2.4 *Seja $P \in \mathcal{P}_V$. Dada $\tilde{P}$ como na Definição 3.2.2, temos que*

(i) se $c_1(P) > |W|$ e $c_0(P) > |U|$ então $\tilde{P}$ é propriedade monotônica não-trivial;

(ii) vale a relação $C^{\mathcal{A}l}(P) \geq C^{\mathcal{A}l}(\tilde{P})$.

Demonstração

(i) Claramente $\tilde{P}$ é uma propriedade sobre $\mathcal{G}_{U,W}$ invariante por isomorfismos. Vamos provar que $\tilde{P}$ é monotônica. Tome $G, H \in \mathcal{G}_{U,W}$ tais que $G \subseteq H$ e $G \in \tilde{P}$. Queremos mostrar que $H \in \tilde{P}$. Mas isso é imediato pois $G \subseteq H$ implica $\tilde{G} \subseteq \tilde{H}$ (onde $\tilde{G}, \tilde{H} \in \mathcal{G}_V$ são dados como na Definição 3.2.2), e por definição $\tilde{G} \in P$ pois $G \in \tilde{P}$. Usando a monotonicidade de P, temos que $\tilde{H} \in P$ e conseqüentemente $H \in \tilde{P}$.

Vamos provar agora que $\tilde{P}$ é não-trivial. Por hipótese, temos que $c_1(P) > |W| = w$, logo o grafo vazio $E_{U,W} \notin \tilde{P}$. Além disso, temos que $c_0(P) > |U| = u$ e portanto $K_{U,W} \in \tilde{P}$. Assim, tem-se que $\tilde{P} \neq \mathcal{G}_{U,W}$ e $\tilde{P} \neq \emptyset$, logo por definição $\tilde{P}$ é de fato não-trivial.

(ii) Primeiramente, observe que todo algoritmo determinístico que computa P, ou seja todo algoritmo em $\mathcal{A}_P$, pode ser trivialmente adaptado para computar $\tilde{P}$. De fato, seja $A \in \mathcal{A}_P$, descrevamos um algoritmo $\tilde{A}$ que computa $\tilde{P}$. Suponha que $\tilde{A}$ recebeu $G \in \mathcal{G}_{U,W}$ como entrada. Então tomamos $\tilde{G} \in \mathcal{G}_V$ acrescentando em G todas as arestas com ambas as pontas em W, e submetemos $\tilde{G}$ assim obtido a A. O algoritmo $\tilde{A}$ responde que G satisfaz $\tilde{P}$ se e somente se A responde que $\tilde{G}$ satisfaz P. Claramente custo$(\tilde{A}, G) \leq$ custo$(A, \tilde{G})$ para todo $G \in \mathcal{G}_{U,W}$.

Usando o Lema 3.2.1, obtemos o resultado desejado.

□

Lema 3.2.5 *Seja $P \in \mathcal{P}_V$. Dada $\hat{P}$ como na Definição 3.2.3, temos que*

(i) se $c_0(P) \leq |U|$ então $\hat{P}$ é propriedade monotônica não-trivial;

(ii) vale a relação $C^{\mathcal{A}l}(P) \geq C^{\mathcal{A}l}(\hat{P})$.

Demonstração

(i) É imediato que $\hat{P}$ é uma propriedade sobre $\mathcal{G}_U$ invariante por isomorfismo. A demonstração de que $\hat{P}$ é monotônica é análoga àquela do Lema 3.2.4 (i). Quanto a $\hat{P}$ ser não-trivial, observe primeiramente que $c_0(P) \leq |U|$ equivale a $c_1(P) > |W| = w$, ou seja, temos que $E_U \notin \hat{P}$. Além disso, pela definição de $\hat{P}$ (Definição 3.2.3) temos que $\hat{K}_U$ é igual a $(K_U + K_W) \cup K_{U,W}$ que é simplesmente K_V, mas $K_V \in P$ pois P é monotônica não-trivial. Logo $K_U \in \hat{P}$

(ii) A demonstração do Lema 3.2.4 (ii) pode ser facilmente adaptada para que valha aqui também.

□

3.3 Propriedades com grafos minimais de grau baixo

Nesta seção, vamos inicialmente dar um lema sobre propriedades de grafos bipartidos. A prova deste resultado faz uso de uma redução a uma propriedade auxiliar semelhante às propriedades $\hat{P}$ e $\tilde{P}$ da seção anterior. Após essa demonstração concluiremos que na realidade uma demonstração semelhante pode ser feita para o caso de grafos genéricos. Em seguida demonstraremos o Teorema 3.3.3, que permitirá a demonstração de um limite inferior quadrático para um conjunto considerável de propriedades de grafos. É talvez conveniente aqui ressaltar a importância que o grau máximo dos grafos considerados terá nas demonstrações subseqüentes.

Seja G um grafo e $M \subset V(G)$ um conjunto de vértices de G. Dizemos que M é um conjunto **admissível** em G, ou simplesmente **G-admissível**, se (i) M é um conjunto independente em G, (ii) para todo $x \in M$ temos $\Gamma_G(x) \neq \varnothing$, e (iii) para todo $x, y \in M$, $x \neq y$, temos $\Gamma_G(x) \cap \Gamma_G(y) = \varnothing$.

Lema 3.3.1 *Seja $P \in \mathcal{P}_{U,W}$ e tome $G \in \min(P)$. Se existem t conjuntos dois-a-dois disjuntos $M_1, M_2, \ldots, M_t$ em U tais que, para cada i, vale que M_i é G-admissível e $|M_i| = m$, então $C^{AI}(P) \geq m^2 t/32$.*

Demonstração

Nesta demonstração, vamos denotar as arestas por xy supondo sempre que o primeiro vértice pertence à classe U e o segundo à W.

Seja $G \in \min(P)$ satisfazendo o enunciado. Denote por M a união dos conjuntos M_i $(1 \leq i \leq t)$ e por T o conjunto das arestas xy de G tais que x não pertence a M, i.e. $T = \{xy \in A(G) : x \in U \setminus M\}$. Para quaisquer $x, x' \in M$, definimos a *superaresta* $(x, \Gamma_G(x'))$ como o conjunto de arestas $\{xy : y \in \Gamma_G(x')\} \subseteq A(K_{U,W})$.

Considere o conjunto $\mathcal{G}_{M,\Psi(G)}$ de grafos bipartidos com classes M e $\Psi(G)$, onde $\Psi(G) = \{\Gamma_G(x) : x \in M\}$, ou seja $\Psi(G)$ é o conjunto das vizinhanças dos vértices de M no grafo G. Se $H \in \mathcal{G}_{M,\Psi(G)}$, então uma aresta de H pode ser interpretada como uma superaresta.

Seja σ_i uma permutação de M_i $(1 \leq i \leq t)$. Para cada seqüência $\sigma_1, \sigma_2, \ldots, \sigma_t$ de tais permutações, definimos o grafo $G_{\sigma_1,\sigma_2,\ldots,\sigma_t} \in \mathcal{G}_{U,W}$ que contém exatamente as seguintes arestas: as arestas do conjunto T definido acima, e as arestas de cada superaresta $(x, \Gamma_G(\sigma_i(x)))$ para todo $x \in M_i$ $(1 \leq i \leq t)$. Todo $G_{\sigma_1,\ldots,\sigma_t}$ assim definido é isomorfo a G, e portanto satisfaz P. Além disso, todo $G_{\sigma_1,\ldots,\sigma_t}$ 'contém' um conjunto distinto de exatamente mt superarestas. De fato, tal conjunto para $G_{\sigma_1,\ldots,\sigma_t}$ é $\{(x, \Gamma_G(\sigma_i(x))) : x \in M_i,\ 1 \leq i \leq t\}$, que claramente tem cardinalidade mt.

Defina a operação † que a cada grafo de $\mathcal{G}_{M,\Psi(G)}$ fornece um grafo de $\mathcal{G}_{U,W}$ da seguinte maneira: se $H \in \mathcal{G}_{M,\Psi(G)}$, então $H^\dagger \in \mathcal{G}_{U,W}$ é tal que o conjunto de arestas de $H^\dagger$ contém exatamente as arestas de T e as arestas de cada superaresta $S \in A(H)$. Defina, então, a propriedade $P^\dagger$ sobre $\mathcal{G}_{M,\Psi(G)}$ tal que $H \in P^\dagger$ se e somente se $H^\dagger \in P$. Observe que $E_{M,\Psi(G)} \notin P^\dagger$ pois $(E_{M,\Psi(G)})^\dagger = E_{U,W} \cup T \subsetneq G$ e $G \in \min(P)$. Além disso $K_{M,\Psi(G)} \in P^\dagger$ pois $G \subseteq (K_{M,\Psi(G)})^\dagger$, e portanto $P^\dagger$ é não-trivial. Claramente $P^\dagger$ é monotônica já que P é monotônica.

Dado um algoritmo determinístico A que testa a propriedade P, podemos obter um algoritmo determinístico $A^\dagger$ para verificar $P^\dagger$ tal que o número de perguntas que $A^\dagger$ precisa para decidir sobre um grafo $H \in \mathcal{G}_{M,\Psi(G)}$ qualquer nunca é maior do que o número de perguntas que A faz sobre o grafo $H^\dagger$ correspondente. Vamos descrever tal $A^\dagger$. Suponha que A faça uma pergunta sobre uma aresta xy. Se $xy \in T$ ou se y não pertence a nenhum $Y \in \Psi(G)$, então $A^\dagger$ não faz nenhuma pergunta. Caso contrário, ou seja se $x \in M$ e y pertence a algum $Y \in \Psi(G)$, então $A^\dagger$ pergunta se $(x, Y) \in A(H)$, caso ainda não tenha feito tal pergunta. Ou seja, o algoritmo $A^\dagger$ faz uma pergunta sobre uma superaresta (x, Y) quando A fizer a primeira pergunta sobre alguma aresta xy de (x, Y). A próxima vez que A perguntar sobre alguma aresta de (x, Y), o algoritmo $A^\dagger$ não faz nenhuma pergunta.

Intuitivamente, podemos dizer que é como se $A^\dagger$, ao invés de fazer uma pergunta sobre uma superaresta (x, Y), fizesse uma pergunta diretamente a $H^\dagger$ sobre uma aresta xy com $y \in Y$: se $xy \in A(H^\dagger)$ então $A^\dagger$ conclui que (x, Y) pertence a $A(H)$, caso contrário conclui que (x, Y) não pertence a $A(H)$. Essa interpretação é coerente pois qualquer superaresta (x, Y) pertence a $A(H)$ se e somente se toda a aresta de (x, Y) pertence a $A(H^\dagger)$. Assim, usando o Lema 3.2.1, podemos concluir que a complexidade aleatória de P é maior ou igual à complexidade aleatória de $P^\dagger$. Basta então achar um limite inferior para a complexidade aleatória de $P^\dagger$.

Note que entre os elementos de $\min(P^\dagger)$, existem $(m!)^t$ grafos com exatamente mt superarestas, cada um correspondendo a um grafo $G_{\sigma_1,\ldots,\sigma_t}$. Chame de N esse subconjunto de grafos de $\min(P^\dagger)$. Qualquer árvore de decisão para $P^\dagger$ que receba um grafo $H \in N$ como entrada, terá que achar todas as mt superarestas de H. Assim, toda árvore de decisão para $P^\dagger$ terá um folha distinta associada a cada grafo em N, sendo que o caminho da raiz da árvore até qualquer uma dessas folhas terá exatamente mt arestas com rótulo 1. Mas, numa árvore de decisão, não existem mais do que $\binom{h}{mt}$ folhas no final de caminhos que tenham comprimento menor ou igual a h e tenham exatamente mt arestas com rótulo 1. Se tomarmos um h tal que $\binom{h}{mt} \leq (m!)^t/2$, então pelo menos metade das folhas associadas aos grafos de N estarão a uma profundidade maior do que h e, conseqüentemente, definindo a distribuição β sobre $\mathcal{G}_{M,\Psi(G)}$ tal que

$$\beta(H) = \begin{cases} 1/|N| & \text{se } H \in N \\ 0 & \text{caso contrário,} \end{cases}$$

teremos que $C(A^\dagger, \beta) \geq h/2$ para qualquer árvore de decisão $A^\dagger$ que compute $P^\dagger$. Seja então $h = \lfloor m^2t/2^{1/mt}\mathrm{e}^2 \rfloor \geq m^2t/16$. Note que

$$\binom{h}{mt} \leq \left(\frac{\mathrm{e}h}{mt}\right)^{mt} \leq \frac{1}{2}\left(\frac{m}{\mathrm{e}}\right)^{mt} \leq \frac{1}{2}(m!)^t,$$

e assim $C(A^\dagger, \beta) \geq h/2 \geq m^2t/32$ para qualquer $A^\dagger \in \mathcal{A}_{P^\dagger}$. Pelo Teorema 2.2.1, temos que $\mathcal{C}^{\mathcal{AI}}(P^\dagger) = \mathcal{C}^{\mathcal{D}}(P^\dagger) \geq \min_A C(A, \beta) \geq m^2t/32$.

□

Claramente, as técnicas usadas na demonstração anterior também são válidas

para provar um resultado análogo para propriedades de grafos genéricos, que enunciamos a seguir.

Lema 3.3.2 *Seja $P \in \mathcal{P}_V$ e tome $G \in \min(P)$. Se existem t conjuntos dois-a-dois disjuntos $M_1, M_2, \ldots, M_t \subset V(G)$ de tamanho m tais que $M = \bigcup M_i$ é um conjunto independente em G e cada M_i $(1 \leq i \leq t)$ é G-admissível, então $C^{Al}(P) \geq m^2 t/32$.*

□

Com este resultado, podemos provar o seguinte teorema.

Teorema 3.3.3 *Para toda propriedade $P \in \mathcal{P}_V$ e todo grafo $G \in P$ tal que $\Delta(G) = d$ vale*

$$C^{Al}(P) \geq \frac{v^2}{512(d^3 + d^2 + d + 1)}.$$

Demonstração

Seja $G \in P$ com grau máximo $\Delta(G)$ igual a d. Provaremos que existe um grafo em $\min(P)$ que satisfaz as condições do Lema 3.3.2 para algum $t \geq (d^2+1)/(d+1)$ e algum $m \geq v/4(d^2+1)$, ou que senão existe um grafo em $\min(P^*)$ com $v^2/16d$ ou mais arestas.

Tome $G' \in \min(P)$ tal que $G' \subseteq G$ e considere os dois casos a seguir.

Caso 1. G' tem pelo menos $v/2$ vértices isolados.

Suponha que exista um grafo $H \in \min(P^*)$ com menos de $v^2/16d$ arestas. Neste caso, vamos provar que existe empacotamento de H e G', chegando a uma contradição. De fato, identifique os $\lceil v/2 \rceil$ vértices de maior grau de H com vértices isolados de G' e sejam G'_1 e H_1 os subgrafos induzidos pelos vértices restantes de G e H respectivamente. Observe que se $\Delta(H_1) \geq v/4d$ então $2|A(H)| \geq \lceil v/2 \rceil v/4d \geq v^2/8d$, o que contradiz a maneira como H foi escolhido. Portanto $\Delta(H_1) < v/4d$. Além disso, como $G'_1 \subseteq G' \subseteq_g G$, temos que $\Delta(G'_1) \leq d$. Assim $\Delta(H_1)\Delta(G'_1) \leq v/4$ e podemos aplicar o Teorema 2.1.6 (i), obtendo um empacotamento de G'_1 e H_1 e, conseqüentemente, um empacotamento de G' e H. Mas isso contradiz o Lema 2.1.3. Concluímos portanto que se G' tem pelo menos $v/2$ vértices isolados então qualquer grafo em $\min(P^*)$ tem pelo menos $v^2/16d$ arestas. Sendo assim, pelo Teorema 2.2.2 vem que $C^{Al}(P) = C^{Al}(P^*) \geq v^2/16d \geq v^2/512(d^3 + d^2 + d + 1)$.

Caso 2. G' tem pelo menos $v/2$ vértices de grau positivo.

Seja S um conjunto com $\lceil v/2 \rceil$ vértices de grau positivo em G'. Vamos provar as duas afirmações a seguir.

Afirmação (1) Qualquer conjunto $S' \subseteq S$ com $|S'| \geq v/4$ contém um conjunto G'-admissivel com pelo menos $v/4(d^2+1)$ vértices.

Seja $M \subseteq S'$ um conjunto G'-admissível. Um vértice $y \in S'$ é tal que $M \cup \{y\}$ não é G'-admissível se e somente se $y \in \Gamma(M)$ ou $y \in \Gamma(\Gamma(M)) \setminus M$. Suponha agora que M é um conjunto G'-admissível maximal contido em S'. Assim, pela observação acima, temos que $S' \setminus M \subset \Gamma(M) \cup \Gamma(\Gamma(M)) \setminus M$. Como $|\Gamma(\Gamma(M)) \setminus M| \leq |\Gamma(M)|(d-1) \leq d|M|(d-1)$, temos que $|S'| \leq |M| + d|M| + d|M|(d-1) = |M|(d^2+1)$. Se M tem menos de $v/4(d^2+1)$ elementos, então $|S'| \leq |M|(1+d^2) < (1+d^2)v/4(d^2+1) = v/4$, o que é uma contradição. Concluímos que $|M| \geq v/4(d^2+1)$.

Afirmação (2) O conjunto S contém pelo menos $(d^2+1)/(d+1)$ conjuntos M_i tais que cada M_i é um conjunto G'-admissivel, $|M_i| \geq v/4(d^2+1)$, e $M = \bigcup M_i$ é independente.

De fato, como S tem $\lceil v/2 \rceil$ vértices, podemos usar a Afirmação (1) e concluir que enquanto tivermos $v/4$ vértices em S é possível extrair um conjunto M_i com as propriedades acima. Assim, a quantidade de conjuntos M_i's que podem ser retirados de S junto com suas respectivas vizinhanças é maior ou igual a

$$\frac{v/4}{(1+d)\{v/4(d^2+1)\}} = \frac{d^2+1}{d+1}.$$

Com essas duas afirmações e usando o Lema 3.3.2, obtemos o limite inferior desejado:

$$C^{Al}(P) \geq \frac{1}{32}\left(\frac{v}{4(d^2+1)}\right)^2 \frac{d^2+1}{d+1} = \frac{v^2}{512(d^3+d^2+d+1)}.$$

□

Este teorema é particularmente interessante se considerarmos uma propriedade monotônica não-trivial $P \in \mathcal{P}_V$ cujo conjunto $\min(P)$ contém um grafo G tal que $\Delta(G) = O(1)$, ou seja cujo grau máximo é uma constante. Alguns exemplos de tais propriedades são os seguintes: ser hamiltoniano, ser conexo, conter um k-fator para um $k = O(1)$ fixo, e não ser planar. (Para as definições destas propriedades, ver [10], [13].) Note que para uma propriedade P assim, o Teorema 3.3.3 anterior garante que $C^{Al}(P) = \Omega(v^2)$, lembrando que $v = |V|$.

3.4 O limite inferior de $\Omega(v^{5/4})$

Aqui, inicialmente demonstraremos a redução do caso de propriedades sobre $\mathcal{G}_V$ para o de grafos bipartidos, e em seguida daremos o limite inferior para a complexidade aleatória de propriedades de grafos bipartidos. Observe que a redução a grafos bipartidos se faz necessária para que possamos usar o Teorema 2.3.4. Este teorema é essencial na demonstração de todos os limites inferiores superlineares conhecidos atualmente, e não tem equivalente para propriedades em $\mathcal{P}_V$.

Vamos denotar por $b_{u,w}$ o mínimo dentre as complexidades aleatórias de propriedades monotônicas não-triviais de grafos bipartidos com classes que tenham ordens u e w. Ou seja, se como sempre temos $|U| = u$ e $|W| = w$,

$$b_{u,w} = \min\{C^{Al}(P) : P \in \mathcal{P}_{U,W}\}.$$

Teorema 3.4.1 *Seja $P \in \mathcal{P}_V$ uma propriedade de grafos. Para $|V| = v \geq 4100$, temos que $C^{Al}(P) \geq \min\{v^{5/4}/32, b_{\lceil v/2\rceil,\lfloor v/2\rfloor}\}$.*

Demonstração

Suponha que $U, W \subset V$ são tais que $V = U \cup W$, $u = |U| = \lceil v/2\rceil$ e $w = |W| = \lfloor v/2\rfloor$. Pelo Teorema 2.2.2, podemos supor que o número de arestas de qualquer grafo em $\min(P)$ ou em $\min(P^*)$ é estritamente menor que $v^{5/4}/32$.

Vamos considerar dois casos.

Caso 1. Vale que $c_1(P) > \lceil v/2\rceil$ e $c_0(P) > \lceil v/2\rceil$.

Seja $\tilde{P}$ como na Definição 3.2.2. Temos então que $C^{Al}(P) \geq C^{Al}(\tilde{P}) \geq b_{\lceil v/2\rceil,\lfloor v/2\rfloor}$.

Caso 2. Vale que $c_1(P) \leq \lceil v/2\rceil$.

Como $c_0(P^*) = c_1(P)$ e as complexidades aleatórias de P e P^* são iguais, a demonstração deste caso é análoga à do Caso 3 a seguir.

Caso 3. Vale que $c_0(P) \leq \lceil v/2\rceil$.

Seja $\hat{P}$ como na Definição 3.2.3. Vamos provar que existe $H \in \hat{P}$ tal que $\Delta(H) < v^{1/4}/8$. Seja $G \in \min(P)$. Coloque os $\lfloor v/2\rfloor$ vértices de G de maior grau na classe W e o restante em U. Observe que $\Delta_U(G) < v^{1/4}/8$, pois caso contrário teríamos que $2|A(G)| \geq (v^{1/4}/8)(1 + \lfloor v/2\rfloor) \geq v^{5/4}/16$, contradizendo nossas suposições iniciais.

Seja então $H = G[U]$. Claramente, vale que $H \in \hat{P}$, pois $G \in P$ e $G \subseteq K_W \times H = \hat{H}$, onde $\hat{H}$ é dado na Definição 3.2.3. Usando o Teorema 3.3.3 e o fato de que $\Delta(H) \leq \Delta_U(G) < v^{1/4}/8$, obtemos

$$\begin{aligned} C^{Al}(P) &\geq \frac{\lceil v/2 \rceil^2}{512(v^{3/4}/512 + v^{1/2}/64 + v^{1/4}/8 + 1)} \\ &\geq \frac{v^2}{16v^{3/4}(1/4 + 2v^{-1/4} + 16v^{-1/2} + 128v^{-3/4})} \\ &\geq \frac{v^{5/4}}{16}. \end{aligned}$$

A última desigualdade valendo para $v \geq 4096$.

□

O próximo resultado mostra que a complexidade aleatória de qualquer propriedade monotônica não-trivial de grafos bipartidos com classes de vertices de cardinalidades u e w é $\Omega(u^{3/4}w^{1/2})$.

Teorema 3.4.2 *Vale que* $b_{u,w} \geq u^{3/4}w^{1/2}/18$ *para todo* $u, w \geq 2600$.

Demonstração

Sejam U e W dois conjuntos disjuntos com $|U| = u$ e $|W| = w$, e seja $P \in \mathcal{P}_{U,W}$ uma propriedade monotônica não-trivial de grafos bipartidos com classes de vértices U e W. Seja G_1 um grafo minimal de $\min(P)$ com relação a $\preccurlyeq_W$.

Vamos supor sem perda de generalidade que $|U| = u \geq w = |W|$. Pelo Teorema 2.2.2, podemos supor que os grafos em $\min(P)$ e em $\min(P^*)$ têm menos de $u^{3/4}w^{1/2}/18$ arestas e portanto $\overline{d}_W(G_1) \leq u^{3/4}w^{1/2}/18w = u^{3/4}w^{-1/2}/18$.

Consideremos dois casos.

Caso 1. Vale que $\Delta_W(G_1) \geq u^{3/2}w^{-1}/32$.

Usando o Teorema 2.3.4, temos:

$$C^{Al}(P) \geq \frac{w}{2}\left(\frac{\Delta_W(G_1)+2}{1+4\overline{d}_W(G_1)} - 1\right) \geq \frac{w}{2}\left(\frac{2+u^{3/2}w^{-1}/32}{1+4u^{3/4}w^{-1/2}/18} - 1\right) =$$

$$\frac{u^{3/4}w^{1/2}}{18}\left(\frac{162u^{-3/4}w^{1/2} + (81/32)u^{3/4}w^{-1/2}}{9+2u^{3/4}w^{-1/2}} - 9u^{-3/4}w^{1/2}\right) \geq \frac{u^{3/4}w^{1/2}}{18},$$

a última desigualdade valendo para u, w suficientemente grandes (note que u, $w \geq 2600$ já são o suficente).

Caso 2. Vale que $\Delta_W(G_1) < u^{3/2}w^{-1}/32$.

Sabemos pelo Lema 2.1.4 que $\min(P)$ ou $\min(P^*)$ tem um grafo com $\lfloor u/2 \rfloor$ ou mais vértices isolados em U. Suponha que $\min(P^*)$ tenha um grafo assim (se não fosse este o caso, poderíamos ter tomado P^* ao invés de P desde o início desta demonstração, já que $\mathcal{C}^{\mathcal{AI}}(P) = \mathcal{C}^{\mathcal{AI}}(P^*)$), Então pelo Lema 2.1.5 todos os grafos em $\min(P)$ têm pelo menos $u/2$ vértices com grau positivo na classe U. Em particular G_1 tem essa quantidade de vértices de grau positivo em U. Além disso, entre esses $u/2$ vértices de grau positivo de G_1, existem pelo menos $\lceil u/4 \rceil$ vértices de grau menor que $u^{-1/4}w^{1/2}/4$. Se isso não fosse verdade, teríamos que $|A(G_1)| \geq (u/4)(u^{-1/4}w^{1/2}/4) > u^{3/4}w^{1/2}/18$, o que contraria nossa suposição do início desta demonstração. Seja então $S \subseteq U$ com $\lceil u/4 \rceil$ vértices de grau positivo menor que $u^{-1/4}w^{1/2}/4$.

Vamos provar duas afirmações para o grafo G_1, semelhantes às que provamos para o Caso 2 da demonstração do Teorema 3.3.3.

Afirmação (1) Todo conjunto $S' \subseteq S \subseteq U$ com $|S'| \geq u/8$ contém um conjunto G_1-admissível de cardinalidade $\lceil 16u^{-1/4}w^{1/2} \rceil$.

De fato, tome $M \subseteq S'$ conjunto G_1-admissível maximal contido em S'. Afirmamos que $|M| \geq 16u^{-1/4}w^{1/2}$. Para verificar esta desigualdade, assuma o contrário, e observe que um vértice $x \in S_1$ é tal que $M \cup \{x\}$ não é G_1-admissível se e somente se x é adjacente a algum vértice em $\Gamma(y)$ para algum $y \in M$. Mas então, o número de vértices $x \in S_1$ é limitado superiormente por

$$\sum_{y \in M} |\Gamma(y)|\Delta_W(G_1) \leq |M|\Delta_S(G_1)\Delta_W(G_1) < \frac{16w^{1/2}}{u^{1/4}} \cdot \frac{w^{1/2}}{4u^{1/4}} \cdot \frac{u^{3/2}}{32w} = u/8,$$

e portanto existe $x \in S_1$ tal que $M \cup \{x\}$ é G_1-admissível, contrariando a escolha de M. Concluímos que se M é maximal então $|M| \geq 16u^{-1/4}w^{1/2}$.

Afirmação (2) Existem no mínimo $u^{5/4}w^{-1/2}/128$ conjuntos disjuntos $M_i \subseteq S$ tais que cada M_i é G_1-admissível e $|M_i| = \lceil 16u^{-1/4}w^{1/2} \rceil$.

Para provar a Afirmação (2), observe que pela Afirmação (1) podemos remover conjuntos M_i de S enquanto houver pelo menos $u/8$ vértices em S. Sendo $|S| = \lceil u/4 \rceil$, o número de conjuntos M_i que podemos retirar de S é maior ou igual a $(u/8)/16u^{-1/4}w^{1/2} = u^{5/4}w^{-1/2}/128$, e assim temos a afirmação.

Pelo Lema 3.3.1, segue que

$$C^{AI}(P) \geq \frac{1}{32}(\frac{16w^{1/2}}{u^{1/4}})^2 \frac{u^{5/4}}{128w^{1/2}} = \frac{u^{3/4}w^{1/2}}{16}.$$

□

Com os Teoremas 3.4.1 e 3.4.2, obtemos o seguinte limite inferior.

Teorema 3.4.3 *Para toda* $P \in \mathcal{P}_V$ *tal que* $|V| = v \geq 5200$, *vale que* $C^{AI}(P) \geq v^{5/4}/50$.

□

Capítulo 4

Aperfeiçoamento de resultados anteriores

Neste capítulo demonstraremos o melhor limite inferior conhecido atualmente para a complexidade aleatória de propriedades monotônicas não-triviais, devido a P. Hajnal. Iniciamos enunciando uma modificação do Teorema 2.3.4. Em seguida demonstraremos o item (ii) do Teorema 2.1.6 e o Teorema 2.1.7 de empacotamento devido a Hajnal. Deixamos a demonstração do item (ii) do Teorema 2.1.6 para este capítulo, pois a demonstração do Teorema 2.1.7 está intimamente ligada a ela. Vamos então definir o conceito de pré-empacotamento que nos permitirá concentrar nossa atenção em alguns subgrafos de grafos minimais das propriedades consideradas. Vamos também definir dois grafos e provar algumas de suas propriedades. Munidos desses grafos, provaremos na última sessão o limite inferior de $\Omega(u^{4/3})$ para a complexidade aleatória de propriedades monotônicas não-triviais de grafos bipartidos com classes U e W, onde $|U| = u$ e $|W| = w$. Finalmente, demonstraremos a redução do caso de grafos genéricos para o de grafos bipartidos, determinando assim um limite inferior de $\Omega(v^{4/3})$ para a complexidade aleatória de propriedades monotônicas não-triviais de grafos de ordem v.

O mérito de Hajnal reside no seu Teorema 2.1.7 de empacotamento, que por si só já é uma contribuição interessante, e no fato de ter visualisado o uso conjunto desse teorema com a nova versão do Teorema 2.3.4 para provar o limite inferior para grafos bipartidos citado acima. Com isso a demonstração para grafos genéricos fica extremamente simplificada.

Se nos resultados do capítulo anterior havia um uso intenso do grau máximo

dos grafos considerados, este Capítulo 4 pode ser caracterizado pela sua ênfase no uso do grau médio dos vértices dos grafos.

4.1 Modificação de uma técnica anterior

O Teorema 2.2.2, essencialmente devido a Yao [44], concerne grafos em $\min(P)$ minimais com relação a $\preccurlyeq_U$, mas esse lema pode ser modificado de forma que a sua conclusão seja também válida para um outro tipo de grafo de $\min(P)$. Esta modificação foi proposta e provada por P. Hajnal em [22]. Este novo lema é um dos ingredientes em sua prova do melhor limite inferior conhecido atualmente para a complexidade aleatória de propriedades monotônicas não-triviais. Vamos provar aqui nesta seção inicial o lema de Hajnal.

Sabemos por um resultado anterior bastante simples (Lema 2.1.5) que, dada uma propriedade $P \in \mathcal{P}_{U,W}$, existe pelo menos um grafo em $\min(P) \cup \min(P^*)$ com $\lfloor u/2 \rfloor$ ou mais vértices isolados em U. (Lembre que sempre temos $u = |U|$ e $w = |W|$.)

Lema 4.1.1 *Seja dada uma propriedade $P \in \mathcal{P}_{U,W}$. Suponha que exista pelo menos um grafo em $\min(P)$ com $\lfloor u/2 \rfloor$ ou mais vértices isolados em U. Seja G um grafo minimal com relação a $\preccurlyeq_U$ dentre os grafos de $\min(P)$ com $\lfloor u/2 \rfloor$ ou mais vértices isolados. Então*

$$C^{AI}(P) \geq \max\left\{\frac{u}{2}, \frac{u}{4} \cdot \frac{\Delta_U(G) - 8\bar{d}_U(G) + 1}{8\bar{d}_U(G) + 1}\right\}.$$

Demonstração

Basta modificar a demonstração do Teorema 2.3.4 convenientemente. A seguir assumimos familiaridade com a demonstração do Teorema 2.3.4, e apenas descrevemos as modificações necessárias. O caso em que $\Delta_U(G) \leq 16\bar{d}_U(G)$ é imediato pois sabemos que $C^{AI}(P) \geq u/2$. Assim supomos que $\Delta_U(G) \geq 16\bar{d}_U(G)$. Aqui começamos pondo $u' = \lceil u/4 \rceil$. Suponha que $U = \{x_0, x_1, \ldots, x_{u-1}\}$ com os graus dos vértices x_i $(0 \leq i < u)$ satisfazendo $d(x_1) \leq d(x_2) \leq \ldots \leq d(x_{u'}) \leq d(x_0) = \Delta_U(G)$, $d(x_{u'}) \leq d(x_i)$ para $u' < i < \lceil u/2 \rceil$, e $d(x_i) = 0$ para $\lceil u/2 \rceil \leq i < u$. Desta forma $x_1, \ldots, x_{u'}$ têm os menores graus possíveis dentre os vértices em $\{x_0, x_1, x_2, \ldots, x_{\lceil u/2 \rceil - 1}\}$.

Note que $d(x_i) < 4\overline{d}_U(G)$ para todo $1 \leq i \leq u'$. Pomos agora $m = \lfloor 8\overline{d}_U(G) \rfloor$. Obtemos G' a partir de G incluindo arestas incidentes nos vértices $x_0, x_1, \ldots, x_{u'}$, como na demonstração do Teorema 2.3.4, para que todos esses vértices tenham grau maior ou igual a $\Delta_U(G)$ e estritamente menor que $\Delta_U(G)+m$. Geramos um subgrafo aleatório H de G' tirando de G' os conjuntos de arestas aleatórios W_i $(0 \leq i \leq u')$. Esses conjuntos aleatórios W_i's devem ser tais que $|W_i| = m$, como na demonstração original. Note que não alteramos as vizinhanças dos vértices x_i $(\lceil u/2 \rceil \leq i < u)$, portanto G' e H também têm pelo menos $\lfloor u/2 \rfloor$ vértices isolados. Na demonstração do Teorema 2.3.4, o único ponto em que usamos o fato de que G é minimal com relação a $\preccurlyeq_U$ é quando provamos que $H \notin P$. Isto continua valendo neste caso, pois para qualquer $J \in \min(P)$ temos que $J \not\subseteq H$. De fato, se $J \in \min(P)$ e J não tem $\lfloor u/2 \rfloor$ vértices isolados, então como H tem pelo menos $\lfloor u/2 \rfloor$ vértices isolados, claramente $J \not\subseteq H$. Suponha então que exista $J \in \min(P)$ com $\lfloor u/2 \rfloor$ vértices isolados tal que $J \subseteq H$. Segue que $J \preccurlyeq_U H$. Mas, por construção, temos que $G \not\preccurlyeq_U H$ e portanto $G \not\preccurlyeq_U J$, o que é uma contradição pois G é minimal com relação a $\preccurlyeq_U$ dentre os grafos em $\min(P)$ com $\lfloor u/2 \rfloor$ vértices isolados. Assim, como P é monotônica crescente e para todo $J \in \min(P)$ vale $J \not\subseteq H$, temos que $H \notin P$. O restante da demonstração é análogo à demonstração do teorema original.

□

Note que, na demonstração, os vértices $v_1, \ldots, v_{u'}$ não precisam necessariamente ter grau positivo. Mais do que isso, se pudermos garantir um número substancialmente grande de vértices isolados (além dos $\lceil u/2 \rceil$ da hipótese), então podemos eliminar o grau médio do denominador do enunciado. Ou seja, dada uma propriedade P, se existir um grafo $G_0 \in \min(P)$ com $\lfloor 3u/4 \rfloor$ ou mais vértices isolados tal que G_0 é minimal com relação a $\preccurlyeq_U$ dentre os grafos de $\min(P)$ com $\lfloor u/2 \rfloor$ ou mais vértices isolados, então podemos facilmente modificar a demonstração anterior e obter que $\mathcal{C}^{\mathcal{AI}}(P) \geq \max\{u/2, u(\Delta_U(G)+1)/8\}$. Ou seja, este resultado nos permite achar um limite inferior para algumas propriedades que tenham grafos minimais com grau máximo alto, fazendo um contraponto interessante com as técnicas do capítulo anterior. Nossa intenção ao fazermos esta observação foi mostrar uma modificação possível do Teorema 2.3.4. Acreditamos que a obtenção de melhores limites inferiores para a complexidade aleatória de propriedades em $\mathcal{P}_{U,W}$ (e eventualmente

em $\mathcal{P}_V$), necessariamente passa por versões mais fortes desse teorema.

4.2 Um resultado em empacotamento de grafos

Vamos demonstrar agora o item (ii) do Teorema 2.1.6 enunciado no Capítulo 2. Para esta demonstração seguiremos a sugestão dada por Hajnal em [22] baseada em resultados obtidos por Catlin [14]. Em seguida daremos uma demonstração do Teorema 2.1.7 que melhora esse Teorema 2.1.6 (ii) com a introdução de grau médio.

Teorema 2.1.6 *(ii) Se* $G, H \in \mathcal{G}_{U,W}$ *e* $\Delta_U(G)\Delta_W(H) + \Delta_W(G)\Delta_U(H) \leq \max\{u, w\}$, *então* G *e* H *podem ser empacotados como grafos bipartidos.*

Demonstração

Sem perda de generalidade, podemos supor que $u \leq w$. Sejam U' e W' conjuntos disjuntos com $|U'| = |U| = u$ e $|W'| = |W| = w$, e considere H como um grafo em $\mathcal{G}_{U',W'}$ da forma natural. Para provar que G e H podem ser empacotados, vamos mostrar que existem bijeções $f : U \to U'$ e $g : W \to W'$ que fornecem tal empacotamento. Na verdade, montraremos que qualquer bijeção f entre U e U' admite uma bijeção g conveniente.

Fixe uma bijeção $f : U \to U'$ arbitrária. Defina o grafo bipartido B_f com bipartição (W, W') pondo a aresta xy em $A(B_f)$ onde $x \in W$ e $y \in W'$ se e somente se $\Gamma_G(x) \cap f^{-1}(\Gamma_H(y)) = \varnothing$. Note que se $\{x_i y_i \in A(B_f) : x_i \in W, y_i \in W', 1 \leq i \leq w\}$ é um emparelhamento perfeito em B_f, então $g : W \to W'$ dada por $g(x_i) = y_i$ $(1 \leq i \leq w)$ é uma bijeção como a procurada. Provemos portanto que B_f contém um emparelhamento perfeito.

Lembrando que $\delta_W(B_f)$ é o grau mínimo dos vértices de W no grafo B_f, e analogamente $\delta_{W'}(B_f)$ é o grau mínimo dos vértices de W' em B_f, temos que $\delta_W(B_f) \geq w - \Delta_W(G)\Delta_{U'}(H)$ e $\delta_{W'}(B_f) \geq w - \Delta_{W'}(H)\Delta_U(G)$. Assim, nossa hipótese fornece $\delta_W(B_f) + \delta_{W'}(B_f) \geq w$. Mas, pelo teorema de Hall (ver Apêndice A para o enunciado deste teorema) isso garante a existência de um emparelhamento perfeito em B_f. De fato, seja $\varnothing \neq S \subseteq W$, e suponha por absurdo que $|\Gamma_{B_f}(S)| < |S|$. Então, se $y \in W'$ é um vértice qualquer tal que $y \in W' \setminus \Gamma_{B_f}(S) \neq \varnothing$, então $w \leq \delta_W(B_f) + \delta_{W'}(B_f) \leq |\Gamma_{B_f}(S)| + |\Gamma_{B_f}(y)| < |S| + |\Gamma_{B_f}(y)| \leq w$, o que é uma contradição.

□

O seguinte lema probabilístico será fundamental abaixo. Este resultado segue facilmente da desigualdade de Hoeffding e é um lema extremamente útil da teoria dos grandes desvios em probabilidade. Uma boa referência para este resultado e outros correlatos com aplicações combinatórias e à teoria da computação é [31]. No que segue os logaritmos são todos na base e.

Lema 4.2.1 *Sejam $Y_1, Y_2, \ldots, Y_n$ variáveis aleatórias independentes satisfazendo $0 \leq Y_i \leq 1$ $(1 \leq i \leq n)$. Seja $Y = \sum_1^n Y_i$. Então para todo $0 < \epsilon \leq 1$, temos*

(i) $\mathbf{P}(Y \geq (1+\epsilon)\mathbf{E}(Y)) \leq \exp\{-\epsilon^2\mathbf{E}(Y)/3\}$,

(ii) $\mathbf{P}(Y \leq (1-\epsilon)\mathbf{E}(Y)) \leq \exp\{-\epsilon^2\mathbf{E}(Y)/2\}$.

□

Um corolário imediato do Lema 4.2.1 (i) é o seguinte resultado.

Lema 4.2.2 *Suponha que $0 \leq d_1, d_2, \ldots, d_n \leq L = n/54(\log n + \log\log n)$, e que $\overline{d} = n^{-1}\sum_1^n d_i > 0$. Seja $p \leq 1/3\overline{d}$. Então, se $X_1, X_2, \ldots, X_n$ são variáveis aleatórias independentes $0-1$ com $\mathbf{E}(X_i) = p$ $(1 \leq i \leq n)$, temos*

$$\mathbf{P}\left\{\sum_1^n X_i d_i \geq n/2\right\} \leq (n\log n)^{-3/2}. \tag{1}$$

Demonstração

Podemos supor que $p = 1/3\overline{d}$, pois o lado esquerdo de (1) é claramente crescente em p. Ponha $Y_i = X_i d_i/L$ $(1 \leq i \leq n)$ e seja $Y = \sum_1^n Y_i$. Então $\mathbf{E}(Y) = L^{-1}\sum_1^n \mathbf{E}(X_i)d_i = n/3L$. Note que se $\epsilon = 1/2$ e $\sum_1^n X_i d_i \geq n/2$, então $Y - \mathbf{E}(Y) \geq n/2L - n/3L = \epsilon\,\mathbf{E}(Y)$. Assim

$$\begin{aligned}\mathbf{P}\left\{\sum_1^n X_i d_i \geq n/2\right\} &\leq \mathbb{P}\{Y - \mathbb{E}(Y) \geq \epsilon\,\mathbb{E}(Y)\}\\ &\leq \exp\{-\epsilon^2\mathbb{E}(Y)/3)\}\\ &= \exp\{-n/36L\} = (n\log n)^{-3/2}.\end{aligned}$$

□

Para demonstrar o resultado de empacotamento de Hajnal, precisamos de uma pequena variante do Lema 4.2.2, que segue facilmente deste resultado. Denotamos

por $[n]$ o conjunto $\{1,2,\ldots,n\}$. No que segue, se $0 \leq k \leq n$, escrevemos S_k para um subconjunto aleatório de $[n]$ de cardinalidade k, onde tais conjuntos são todos considerados equiprováveis.

Corolário 4.2.3 *Suponha que* $0 \leq d_1, d_2, \ldots, d_n \leq L = n/54(\log n + \log\log n)$ *e que* $\overline{d} = n^{-1}\sum_1^n d_i > 0$. *Seja* $D \geq 1$ *um inteiro com* $3\overline{d}D \leq n$. *Então*

$$\mathbf{P}\Big\{ \sum_{i \in S_D} d_i \geq n/2 \Big\} \leq 2/n(\log n)^{3/2}.$$

Demonstração

Sejam $X_1, X_2, \ldots, X_n$ variáveis aleatórias independentes $0-1$ com $\mathbf{E}(X_i) = p = D/n$. Seja $S = S(X_1, X_2, \ldots, X_n)$ o subconjunto aleatório de $[n]$ dado por $S = \{i : X_i = 1\}$, e ponha $X_S = \sum_{i \in S} d_i = \sum_1^n X_i d_i$. Então

$$\begin{aligned}
\mathbf{P}\Big\{ \sum_1^n X_i d_i \geq n/2 \Big\} &= \sum_{k=0}^n \mathbf{P}\Big\{ X_S \geq n/2 \;\Big|\; |S| = k \Big\} \mathbf{P}(|S| = k) \\
&= \sum_{k=0}^n \mathbf{P}(X_{S_k} \geq n/2) \binom{n}{k} p^k (1-p)^{n-k} \\
&\geq \mathbf{P}(X_{S_D} \geq n/2) \binom{n}{D} p^D (1-p)^{n-D} \\
&\geq \mathrm{e}^{-1/6D} (2/\pi n)^{1/2} \mathbf{P}(X_{S_D} \geq n/2),
\end{aligned}$$

onde a última desigualdade segue da fórmula de Stirling. Pelo Lema 4.2.2, temos assim que

$$\mathbf{P}(X_{S_D} \geq n/2) \leq \mathrm{e}^{1/6D} (\pi n/2)^{1/2} (n \log n)^{-3/2} \leq 2/n(\log n)^{3/2}.$$

□

Agora podemos provar o teorema de empacotamento de Hajnal. Observe que a demonstração deste teorema é na realidade a aplicação dos resultados probabilísticos anteriores à demonstração do item (ii) do Teorema 2.1.6.

Teorema 4.2.4 *Sejam dados* $G, H \in \mathcal{G}_{U,W}$ *e suponha que*

(i) $u \leq w \leq 2u$,

(ii) $\overline{d}_U(G)\Delta_W(H) \leq u/3$,

(iii) $\overline{d}_U(H)\Delta_W(G) \leq u/3$,

(iv) $\Delta_U(G)$, $\Delta_U(H) \leq u/54(\log u + \log\log u)$.

Então G e H podem ser empacotados como grafos bipartidos, desde que $u \geq 55$.

Demonstração

Consideramos U', W' como na demonstração do Teorema 2.1.6 (ii), e para uma bijeção $f : U \to U'$ consideramos novamente o grafo bipartido B_f. Desta vez mostramos que a *maioria* das bijeções f são tais que B_f contém um emparelhamento perfeito. Mais precisamente, consideramos o conjunto $B(U, U')$ de todas as bijeções $f : U \to U'$ como um espaço de probabilidades onde todas as bijeções são equiprováveis. Afirmamos então que

$$\mathbb{P}(f \text{ é tal que } B_f \text{ tem um emparelhamento perfeito}) \geq 1 - 8(\log u)^{-3/2} > 0,$$

de onde seguirá o nosso resultado. (Observe que $8(\log u)^{-3/2} < 1$ para $u \geq 55$.) Para $x \in W \cup W'$, seja E_x o evento $\{d_{B_f}(x) < w/2\}$. Pela demonstração do Teorema 2.1.6 (ii), é suficiente mostrar que

$$\mathbb{P}(E_x \text{ ocorre para algum } x \in W \cup W') \leq 8(\log u)^{-3/2},$$

ou ainda que para todo $x \in W \cup W'$ temos $\mathbb{P}(E_x) \leq 2/u(\log u)^{3/2}$. Para provar esta última desigualdade, seja $x \in W$ um vértice fixo. Suponha que $U' = \{y'_1, y'_2, \ldots, y'_u\}$ e seja $d_i = d_H(y'_i)$ $(1 \leq i \leq u)$. Se $d_{B_f}(x) < w/2$ então $s(x, f) = \sum\{d_i : y'_i \in f(\Gamma_G(x))\} > w/2 \geq u/2$. Mas, pelo Corolário 4.2.3, temos que

$$\mathbb{P}(E_x) = \mathbb{P}(d_{B_f}(x) < w/2) \leq \mathbb{P}(s(x, f) \geq u/2) \leq 2/u(\log u)^{3/2},$$

como queríamos. O caso onde $x \in W'$ é análogo, e assim concluímos o resultado.

□

As constantes na hipótese do lema acima são melhores que aquelas na versão original deste lema e ainda podem ser melhoradas. Por exemplo, podemos substituir (i) por $u \leq w = O(u)$, e os limites em (ii) e (iii) por $(1 - \epsilon)u/2$ para qualquer $\epsilon > 0$ fixo. Para tanto, basta assumir que $\Delta_U(G)$, $\Delta_U(H) \leq u/C\log u$ para C suficientemente grande.

4.3 Preliminares para o resultado mais recente

Nesta seção e na próxima, os grafos bipartidos considerados pertencerão a $\mathcal{G}_{U,W}$ onde $|U| = \lceil v/2 \rceil$, $|W| = \lfloor v/2 \rfloor$ e $v \in \mathbb{N}$. Observe que, neste caso, tem-se $\overline{d}_U(G) \leq \overline{d}(G) \leq \overline{d}_W(G)$.

Vamos definir a seguir dois grafos que serão usados ao longo desta seção.

Definição 4.3.1 *Seja dada uma propriedade $P \in \mathcal{P}_{U,W}$, e suponha que* $\min(P)$ *tenha um grafo com pelo menos $\lfloor u/2 \rfloor$ vértices isolados em U. Então*

(a) escrevemos G_P para um grafo de $\min(P)$ *que é minimal com relação a $\preccurlyeq_U$ dentre os grafos de* $\min(P)$ *com pelo menos $\lfloor u/2 \rfloor$ vértices isolados em U,*

(b) escrevemos H_P para um grafo de $\min(P^*)$ *minimal com relação a $\preccurlyeq_W$.*

Apesar de G_P e H_P pertencerem a $\mathcal{G}_{U,W}$, quando necessário, para maior clareza vamos denotar as classes de G_P por U_G e W_G e as de H_P por U_H e W_H. Introduzimos agora uma nova noção envolvendo empacotamentos. No que segue fixamos uma propriedade $P \in \mathcal{P}_{U,W}$, onde U e W são como acima.

Sejam $G_P = G(P)$ e $H_P = H(P)$ como na definição anterior. Vamos definir conjuntos $U_0 \subseteq U_G$, $W_0 \subseteq W_G$, $U_0' \subseteq U_H$, $W_0' \subseteq W_H$ e um empacotamento entre os subgrafos que esses conjuntos induzem em G_P e H_P. (Veja as figuras na próxima página.) Tome $U_0 \subseteq U_G$ como um conjunto de $\lfloor u/2 \rfloor$ vértices isolados em G_P. Tome $W_0 \subseteq W_G$ com $\lfloor w/8\overline{d}(H) \rfloor$ vértices em W_G de grau maior possível em G_P. Tome $W_0' \subseteq W_H$ com $\lfloor w/8\overline{d}(H) \rfloor$ vértices de W_H de grau menor possível em H_P. Tome $U_0' \subseteq U_H$ como o conjunto dos vizinhos de W_0' em H_P mais uma certa quantidade de vértices de U_H de maior grau possível em H_P, de forma que U_0' tenha $\lfloor u/2 \rfloor$ vértices. (Iremos provar que a vizinhança de W_0' tem no máximo $\lfloor u/4 \rfloor$ elementos, e portanto U_0' terá pelo menos $\lfloor u/4 \rfloor$ vértices de grau alto.)

Seja G_0 o subgrafo $G_P[U_0 \cup W_0]$ de G_P induzido pelo conjunto de vértices $U_0 \cup W_0$ em G_P. Seja H_0 o subgrafo $H_P[U_0' \cup W_0']$ de H_P induzido por $U_0' \cup W_0'$ em H_P. Observe que $A(G_0)$ é vazio, logo G_0 e H_0 podem ser empacotados como grafos bipartidos, onde consideramos H_0 e G_0 como grafos bipartidos com a bipartição natural.

Definição 4.3.2 *Seguindo a notação acima, chamamos de* **pré-empacotamento** *de G_P e H_P a um empacotamento de G_0 e H_0.*

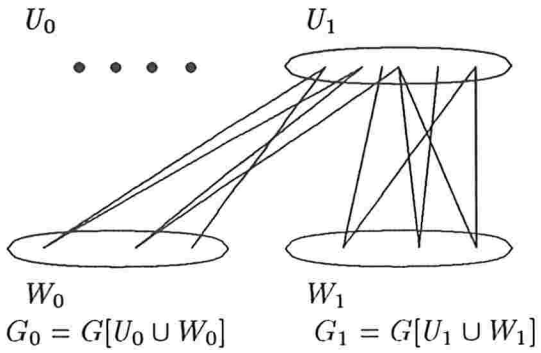

Figura 4.1: Grafo $G = G_P$

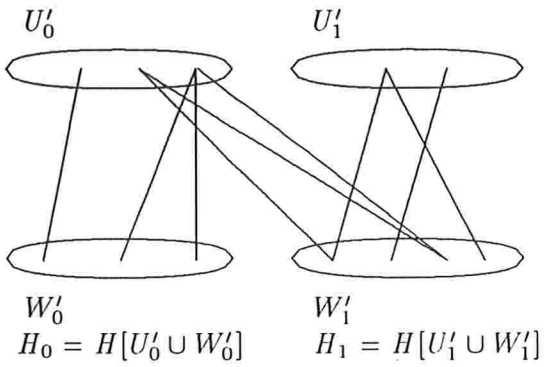

Figura 4.2: Grafo $H = H_P$

Sejam $U_1 = U_G \setminus U_0$, $W_1 = W_G \setminus W_0$, $U_1' = U_H \setminus U_0'$ e $W_1' = W_H \setminus W_0'$. Denotamos por G_1 o subgrafo $G_P[U_1 \cup W_1]$ de G_P induzido por $U_1 \cup W_1$ em G_P. Finalmente, seja H_1 o subgrafo $H_P[U_1' \cup W_1']$ de H_P induzido por $U_1' \cup W_1'$ em H_P.

Na demonstração do Caso 2 do Teorema 3.3.3, utilizamos um artifício semelhante ao do pré-empacotamento: tínhamos dois grafos $G' \in \min(P)$ e $H \in \min(P^*)$, identificávamos $\lceil u/2 \rceil$ vértices isolados de G' com os de grau maior possível de H, definíamos os subgrafos G_1' e H_1 induzidos pelos vértices restantes e encontrávamos um empacotamento de G_1' e H_1, e conseqüentemente um empacotamento de G' e H. O procedimento adotado aqui será semelhante, com a diferença de que definimos pré-empacotamento apenas para os grafos bipartidos G_P e H_P como os da Definição 4.3.1.

Lema 4.3.3 *Seja dada uma propriedade $P \in \mathcal{P}_{U,W}$, e tome $G = G(P)$ e $H = H(P)$ como na Definição 4.3.1. Construa G_0, H_0, G_1 e H_1 como acima.*

(i) Se existe um empacotamento de G_1 e H_1 então este empacotamento mais um pré-empacotamento de G e H formam um empacotamento de G e H.

(ii) Temos que $\overline{d}(H) > 1/4$, e portanto $w/8\overline{d}(H) < w/2$.

(iii) Temos que $\Delta_{U_1'}(H_1) \leq 4\overline{d}(H)$.

(iv) Temos que $\Delta_{W_1}(G_1) \leq 8\overline{d}(G)\overline{d}(H)$.

Demonstração

(i) Como não existem arestas entre U_0 e W_1, nem tão pouco entre U_1' e W_0', a afirmação é imediata.

(ii) Suponha o contrário. Então vale que $1/4 \geq \overline{d}(H) \geq \overline{d}_U(H) = |A(H)|/u$ e portanto $|A(H)| \leq u/4$. Como $|U_0'| = \lfloor u/2 \rfloor$, todas as arestas de H têm neste caso uma ponta em U_0'. Assim, os vértices em U_1' são isolados e o grafo H_1 não tem arestas. Logo, existe empacotamento de H_1 e G_1. Mas pelo item anterior, também existirá empacotamento de G e H, o que contradiz o Lema 2.1.3 (ii) uma vez que $G \in P$ e $H \in P^*$. Assim $w/8\overline{d}(H) < w/2$.

(iii) Vamos primeiramente provar que $\Delta_{W_0'}(H) \leq 2\overline{d}(H)$. Suponha que exista $x_0 \in W_0'$ tal que $d_H(x_0) > 2\overline{d}(H)$. Como W_0' tem os vértices de menor grau em H, todos os vértices de W_1' terão grau maior que $2\overline{d}(H)$. Observe ainda que $|W_1'| = w - \lfloor u/8\overline{d}(H) \rfloor$ e portanto pelo item anterior temos que $|W_1'| > w/2$.

Mas isso é uma contradição, bastando notar que $|A(H)| \geq d_H(x_0) + \sum_{x \in W_1'} d_H(x) > 2\overline{d}(H)(1 + |W_1'|) > 2\overline{d}(H)(1 + w/2) > |A(H)|$.

Tendo provado que $\Delta_{W_0'}(H) \leq 2\overline{d}(H)$, temos que $|\Gamma_H(W_0')| \leq 2\overline{d}(H)|W_0'| = 2\overline{d}(H)\lfloor w/8\overline{d}(H)\rfloor \leq w/4 \leq u/4$. Ou seja, a vizinhança de W_0' tem no máximo $\lfloor u/4\rfloor$ vértices e portanto U_0' tem pelo menos $\lfloor u/4\rfloor$ dos vértices em U_H de maior grau possível em H.

Finalmente, podemos demonstrar que $\Delta_{U_1'}(H_1) \leq 4\overline{d}(H)$. De fato, observe que se existisse algum $x \in U_1'$ tal que $d(x) > 4\overline{d}(H)$, então, como $U_0' \setminus \Gamma(W_0')$ tem pelo menos $\lfloor u/4\rfloor$ dos vértices de U_H com grau maior que o de x em H, vale que $|A(H)| \geq d_H(x) + \sum_{y \in U_0' \setminus \Gamma(W_0')} d_H(y) > 4\overline{d}(H)(1 + |U_0' \setminus \Gamma(W_0')|) \geq 4\overline{d}(H)(1 + \lfloor u/4\rfloor) \geq |A(H)|$. Temos assim uma contradição.

(iv) Suponha por absurdo que $\Delta_{W_1}(G_1) > 8\overline{d}(G)\overline{d}(H)$. Seja $y \in W_1$ tal que $d_{G_1}(y) = \Delta_{W_1}(G_1)$. Como W_0 tem $\lfloor w/8\overline{d}(H)\rfloor$ dos vértices em W_G com grau maior possível em G, teremos $|A(G)| \geq d_G(y) + \sum_{x \in W_0} d_G(x) > 8\overline{d}(G)\overline{d}(H)(1 + \lfloor w/8\overline{d}(H)\rfloor) \geq \overline{d}(G)(1 + w) \geq |A(G)|$, o que é uma contradição.

□

4.4 O resultado mais recente: $C^{Al}(P) = \Omega(v^{4/3})$

Vamos agora demonstrar o melhor limite inferior conhecido para a complexidade aleatória de propriedades monotônicas não-triviais de grafos bipartidos. Em geral, e aqui em particular, procura-se obter primeiro a demonstração do caso de grafos bipartidos, pois assim a demonstração para grafos genéricos fica restrita a uma simples redução. É compreensível portanto a observação de Yao em [44] que o próximo passo no estudo deste assunto seria a demonstração de que $C^{Al}(P) = \Omega(v^2)$ para qualquer $P \in \mathcal{P}_{U,W}$.

Teorema 4.4.1 *Seja $P \in \mathcal{P}_{U,W}$ uma propriedade monotônica não-trivial de grafos bipartidos com bipartição (U, W), onde $|U| = u = \lceil v/2\rceil$ e $|W| = w = \lfloor v/2\rfloor$. Então vale que $C^{Al}(P) \geq u^{4/3}/18$ para $u \geq 6 \times 10^9$.*

Demonstração

Podemos supor que $\min(P)$ tem um grafo com pelo menos $\lfloor u/2\rfloor$ vértices isolados em U. De fato, pelo Lema 2.1.4, se $\min(P)$ não tivesse tal grafo, então $\min(P^*)$

teria um grafo desse tipo e poderíamos fazer a demonstração com P^*, já que P e P^* têm as mesmas complexidades. Sejam $G = G(P)$ e $H = H(P)$ como na Definição 4.3.1. Vamos verificar três casos.

Caso 1. $\overline{d}(G)$ ou $\overline{d}(H)$ é maior ou igual a $u^{1/3}/4$.

Neste caso, podemos usar o Teorema 2.2.2 (ii) e concluímos que $C^{Al}(P) \geq u^{4/3}/4$.

Caso 2. O Caso 1 não vale, e $\Delta_U(G)$ ou $\Delta_W(H)$ é maior ou igual a $2u^{2/3}/3$.

Suponha que $\Delta_U(G) \geq 2u^{2/3}/3$. Podemos usar o Lema 4.1.1 e obtemos

$$C^{Al}(P) \geq \frac{u}{4} \cdot \frac{2u^{2/3}/3 - 2u^{1/3} + 1}{2u^{1/3} + 1} = \frac{u^{4/3}}{18} \cdot \frac{3u^{1/3} - 9 + 9/2u^{1/3}}{2u^{1/3} + 1} \geq u^{4/3}/18,$$

a última desigualdade valendo para $u \geq 10^3$.

A demonstração é análoga para o caso em que $\Delta_W(H) \geq 2u^{2/3}/3$, bastando usar o Teorema 2.3.4 com relação à quase-ordem $\preccurlyeq_W$.

Caso 3. Temos que $\overline{d}(G), \overline{d}(H) < u^{1/3}/4$ e $\Delta_U(G), \Delta_U(H) < 2u^{2/3}/3$.

Sejam G_0, H_0, G_1 e H_1 como na Definição 4.3.2 de pré-empacotamento. Vamos verificar as condições do Teorema 4.2.4 para G_1 e H_1, que têm classes de vértices U_1, W_1 e U_1', W_1' respectivamente. Observemos agora os seguintes fatos.

(i) Por hipótese, temos que $|U_1| = \lceil u/2 \rceil$ e $|W_1| = w - \lfloor w/8\overline{d}(H) \rfloor$. Pelo Lema 4.3.3 (ii), temos $w/8\overline{d}(H) < w/2 \leq u/2$. Portanto vale que $|U_1| \leq |W_1| \leq 2|U_1|$.

(ii) Observe que $\overline{d}_U(G) \leq \overline{d}(G) < u^{1/3}/4$ e portanto $|A(G)| < u^{4/3}/4$. Sendo assim, temos que $\overline{d}_{U_1}(G_1) < u^{4/3}/4\lceil u/2 \rceil \leq u^{1/3}/2$. Além disso, como H_1 é subgrafo de H, vale que $\Delta_{W_1'}(H_1) \leq \Delta_W(H) < 2u^{2/3}/3$. Portanto vale que $\overline{d}_{U_1}(G_1)\Delta_{W_1'}(H_1) \leq u/3$.

(iii) Analogamente ao item anterior, obtemos que $\overline{d}_{U_1'}(H_1) < u^{1/3}/2$. Pelo Lema 4.3.3 (iv), vale que $\Delta_{W_1}(G_1) \leq 8\overline{d}(G)\overline{d}(H)$, e portanto por hipótese temos que $\Delta_{W_1}(G_1) < u^{2/3}/2$. Obtemos então que $\overline{d}_{U_1'}(H_1)\Delta_{W_1}(G_1) \leq u/4$.

(iv) Por hipótese, temos $\Delta_{U_1}(G_1) \leq \Delta_U(G) \leq 2u^{2/3}/3 \leq \lceil u/2 \rceil / 54(\log\lceil u/2 \rceil + \log\log\lceil u/2 \rceil)$, a última desigualdade sendo válida para $u \geq 6 \times 10^9$.

Pelo Lema 4.3.3 (iii), temos $\Delta_{U_1'}(H_1) \leq 4\overline{d}(H) \leq u^{1/3} \leq \lceil u/2 \rceil / 54(\log\lceil u/2 \rceil + \log\log\lceil u/2 \rceil)$, a última desigualdade sendo válida para $u \geq 5 \times 10^4$.

Por (i), (ii), (iii) e (iv), provamos que G_1 e H_1 satisfazem as condições do Teorema 4.2.4 e portanto podem ser empacotados. Mas pelo Lema 4.3.3 (i) temos

que neste caso G e H também podem ser empacotados, contradizendo o fato de que $G \in P$ e $H \in P^*$.

□

Vamos demonstrar aqui a existência de um certo grafo para a propriedade $\hat{P}$ dada na Definição 3.2.3, que auxiliará na redução do problema de propriedades de grafos genéricos a um problema de grafos bipartidos. A demonstração faz uso de um algoritmo do tipo guloso que dado um certo grafo G tem como objetivo obter um conjunto independente em G. Um conjunto de vértices contendo esse conjunto independente irá induzir um subgrafo em G com as propriedades necessárias para que esse subgrafo possa ser usado na demonstração do Teorema 4.4.3.

Lema 4.4.2 *Seja $P \in \mathcal{P}_V$ uma propriedade tal que $c_0(P) \leq \lceil v/2 \rceil$. Seja $G \in \min(P)$. Então existe conjunto $U \subseteq V$ com $|U| = \lceil v/2 \rceil$ tal que se tomarmos $\hat{P}$ como na Definição 3.2.3, então existe $H \in \min(\hat{P})$ tal que*

(i) $\Delta(H) \leq 4\overline{d}(G)$;

(ii) H tem pelo menos $\lfloor v/16\overline{d}(G) \rfloor$ vértices isolados.

Demonstração

Qualquer subconjunto $U = V_0$ de V de tamanho $u = \lceil v/2 \rceil$ é tal que o subgrafo $G[V_0]$ de G induzido por V_0 tem a propriedade $\hat{P} \subseteq \mathcal{G}_{V_0}$. Portanto $\min(\hat{P})$ contém um elemento que é subgrafo de $G[V_0]$. Sendo assim, é suficiente mostrar que para um V_0 apropriado, vale que $G[V_0]$ tem as propriedades (i) e (ii), pois claramente neste caso, qualquer $J \in \min(\hat{P})$ tal que $J \subseteq G[V_0]$ será tal que $\Delta(J) \leq 4\overline{d}(G)$ e J conterá pelo menos $\lfloor v/16\overline{d}(G) \rfloor$ vértices isolados. Por simplicidade, ponha $\overline{d} = \overline{d}(G)$.

Construa um conjunto $X \subseteq V$, escolhendo $\lfloor v/16\overline{d} \rfloor$ vértices em V através do algoritmo guloso a seguir. Inicialmente, ponha $X_0 = \varnothing$ e $G_0 = G$. Pegue um vértice $x_1 \in V(G_0)$ de grau mínimo em G_0 e ponha-o em X_0, obtendo o conjunto $X_1 = X_0 \cup \{x_1\}$. Tire esse vértice e os vértices em sua vizinhança de G_0, obtendo o grafo $G_1 = G_0 \setminus (\{x_1\} \cup \Gamma_{G_0}(x_1))$. Escolha um vértice x_2 de grau mínimo em G_1 e coloque em X_1, obtendo $X_2 = X_1 \cup \{x_2\} = \{x_1, x_2\}$. Retire esse vértice e os vértices em sua vizinhança de G_1, obtendo o grafo $G_2 = G_1 \setminus (\{x_2\} \cup \Gamma_{G_1}(x_2))$. Suponha que continuando este processo, obtemos uma seqüência de conjuntos de vértices $\varnothing = X_0 \subsetneq X_1 \subsetneq X_2 \subsetneq \ldots \subsetneq X_{i-1}$ com $X_j = \{x_1, x_2, \ldots, x_j\} \subseteq V$ $(0 \leq j < i)$, e uma seqüência

de grafos $G = G_0 \supsetneq G_1 \supsetneq G_2 \supsetneq \ldots \supsetneq G_{i-1}$ com $G_j = G_{j-1} \setminus (\{x_j\} \cup \Gamma_{G_{j-1}}(x_j)) = G \setminus (X_j \cup \Gamma_G(X_j))$ $(1 \leq j < i)$. Caso $|X_{i-1}| = \lfloor v/16\overline{d} \rfloor$, terminamos este processo e pomos $X = X_{i-1}$. Caso $|X_{i-1}| < \lfloor v/16\overline{d} \rfloor$ e $|V(G_{i-1})| = 0$, abortamos este processo. Caso $|V(G_{i-1})| \geq 1$ e $|X_{i-1}| < \lfloor v/16\overline{d} \rfloor$, escolhemos um vértice $x_i \in V(G_{i-1})$ de grau mínimo em G_{i-1}, pomos $X_i = X_{i-1} \cup \{x_i\}$, fazemos $G_i = G_{i-1} \setminus (\{x_i\} \cup \Gamma_{G_{i-1}}(x_i))$, e continuamos o procedimento.

Afirmamos agora o seguinte.

Afirmativa (1) O algoritmo acima sempre termina com $X \subset V$ tal que $|X| = \lfloor v/16\overline{d} \rfloor$ e $|\Gamma_G(X)| \leq v/4$.

Para provar a afirmativa acima, observe primeiramente que $v/16\overline{d} < v/2$. De fato, se $v/16\overline{d}(G) \geq v/2$, então $\overline{d}(G) \leq 1/8$ e portanto $|A(G)| \leq v/16$. Ou seja, o grafo G teria no máximo $v/8$ vértices com grau positivo, ou ainda, o grafo G teria pelo menos $7v/8$ vértices isolados. Isso contradiz o Lema 2.1.3, já que nossa hipótese garante que $c_0(P) \leq \lceil v/2 \rceil$.

Suponha que o procedimento acima encontrou até um certo instante $X_0 \subsetneq X_1 \subsetneq \ldots \subsetneq X_k$ e $G_0 \supsetneq G_1 \supsetneq \ldots \supsetneq G_k$, e que $|X_k| \leq \lfloor v/16\overline{d} \rfloor$. Ademais, ponha $J = \{1 \leq j \leq k : |\Gamma_G(X_j)| > v/4\}$ e suponha por absurdo que $J \neq \varnothing$. Seja $i = \min J$. Então claramente $|\Gamma_G(X_{i-1})| \leq v/4$, $|X_{i-1}| \leq \lfloor v/16\overline{d} \rfloor < v/2$, e assim $|V(G_{i-1})| = |V \setminus (X_{i-1} \cup \Gamma_G(X_{i-1}))| \geq v - v/2 - v/4 = v/4$. Assim, se no algoritmo acima μ_i é o número médio de vértices novos que se inclui em $\Gamma_G(X_j)$ $(0 \leq j < i)$, i.e.

$$\mu_i = \frac{1}{i} \sum_{j=0}^{i-1} |\Gamma_G(X_{j+1}) \setminus \Gamma_G(X_j)|,$$

então $\mu_i = |\Gamma_G(X_i)|/i > (v/4)/|X_i| \geq 4\overline{d}$. Assim para algum i_0 $(1 \leq i_0 \leq i)$ temos que $d_{G_{i_0-1}}(x_{i_0}) > 4\overline{d}$. Mas no algoritmo guloso acima x_{i_0} foi escolhido de forma que $d_{G_{i_0-1}}(x_{i_0}) \leq d_{G_{i_0-1}}(y)$ para qualquer $y \in V(G_{i_0-1})$, assim temos $2|A(G)| > 4\overline{d}|V(G_{i_0-1})| \geq 4\overline{d}|V(G_{i-1})| \geq v\overline{d} = 2|A(G)|$, o que é uma contradição. Assim concluímos que $J = \varnothing$ e portanto $|\Gamma_G(X_j)| \leq v/4$ para $1 \leq j \leq k$. Caso $|X_k| = \lfloor v/16\overline{d} \rfloor$, temos que o algoritmo pára com $X = X_k$, e vale que $|\Gamma_G(X)| \leq v/4$, como afirmamos em (1). Suponha que $|X_k| < \lfloor v/16\overline{d} \rfloor$. Então $|V(G_k)| = |V(G) \setminus (X_k \cup \Gamma_G(X_k))| \geq v/4 > 0$, e o algoritmo prosseguirá, achando $x_{k+1} \in G_k$, X_{k+1}, e G_{k+1}. Assim concluímos (1).

Suponha que terminamos o algoritmo acima com saída X. Lembre que $|X| = \lfloor v/16\overline{d} \rfloor < v/2$. Seja $Y_0 \subseteq V(G) \setminus (X \cup \Gamma_G(X))$ um conjunto de vértices cujos graus

em G são os maiores possíveis e tal que $Y = \Gamma_G(X) \cup Y_0$ tem cardinalidade $\lfloor v/2 \rfloor$. Note que $|Y_0| \geq \lfloor v/2 \rfloor - \lfloor v/4 \rfloor$. Seja $V_0 = V(G) \setminus Y$ e note que $|V_0| = \lceil v/2 \rceil$. Observe ainda que $X \subseteq V_0$ e portanto $G[V_0]$ contém pelo menos $\lfloor v/16\overline{d} \rfloor$ vértices isolados. Vamos mostrar agora que $\Delta(G[V_0]) \leq 4\overline{d}$. Suponha que exista $x \in V_0$ tal que o grau de x em $H = G[V_0]$ é maior que $4\overline{d}$. Como os vértices em X são isolados em H, temos que $x \in V_0 \setminus X$. Mas pela escolha dos vértices de $Y_0 = Y \setminus \Gamma(X)$, temos que $2|A(G)| \geq d_H(x) + \sum_{y \in Y_0} d_G(y) > 4\overline{d}(1 + |Y_0|) \geq \overline{d}v = 2|A(G)|$, o que é uma contradição. Assim $\Delta(G[V_0]) \leq 4\overline{d}$. Para completar a demonstração do lema, é suficiente escolher um subgrafo $J \subseteq G[V_0]$ que pertence a $\min(\hat{P})$.

□

Lembre que denotamos por $b_{u,w}$ a complexidade aleatória mínima possível para uma propriedade monotônica não-trivial sobre grafos bipartidos com classes de vértices que tenham ordens u e w.

Teorema 4.4.3 *Seja dada uma propriedade $P \in \mathcal{P}_V$. Temos que $\mathcal{C}^{Al}(P) \geq \min\{v^{4/3}/40, b_{\lceil v/2 \rceil, \lfloor v/2 \rfloor}\}$.*

Demonstração

Tome U satisfazendo o Lema 4.4.2 e $W = V \setminus U$. Vamos considerar três casos.

Caso 1. Temos que $c_1(P) > w$ e $c_0(P) > u$.

Neste caso, tome $\tilde{P}$ como na Definição 3.2.2. Então pelo Lema 3.2.4 (ii) temos que $\mathcal{C}^{Al}(P) \geq \mathcal{C}^{Al}(\tilde{P}) \geq b_{u,w}$.

Caso 2. Vale que $c_1(P) \leq w$.

Como $c_0(P^*) = c_1(P)$ e as complexidades aleatórias de P e P^* são iguais, a demonstração deste caso é análoga à do Caso 3 a seguir.

Caso 3. Vale que $c_0(P) \leq u$.

Considere $\hat{P}$ como na Definição 3.2.3. Para $G \in \min(P)$ construa $H \in \min(\hat{P})$ como no Lema 4.4.2 acima. Tome $F \in \min((\hat{P})^*)$. Pelo Lema 2.1.3 (i), sabemos que não existe empacotamento de F e H. Seja $X_0 \subset V(F)$ um conjunto com $\lfloor v/16\overline{d}(G) \rfloor$ vértices de maior grau possível em F. Identifique arbitrariamente os vértices em X_0 com $\lfloor v/16\overline{d}(G) \rfloor$ vértices isolados de H. Sejam $F_1 \subset F$ e $H_1 \subset H$ os subgrafos induzidos pelos vértices restantes de F e H, respectivamente.

Vamos terminar a prova considerando dois subcasos.

Subcaso 3.1. $\overline{d}(F) \geq v^{1/3}/10$ ou $\overline{d}(G) \geq v^{1/3}/10$.

Se $\overline{d}(F) \geq v^{1/3}/10$, então pelo Lema 3.2.5 (iv) e Teorema 2.2.2 (i), temos que

$$C^{Al}(P) \geq C^{Al}(\hat{P}) = C^{Al}((\hat{P})^*) \geq \frac{u}{2} \cdot \frac{v^{1/3}}{10} \geq \frac{v^{4/3}}{40}.$$

Se $\overline{d}(G) \geq v^{1/3}/10$, então pelo Teorema 2.2.2 (i), temos que $C^{Al}(P) \geq v^{4/3}/20$.

Subcaso 3.2. $\overline{d}(F) < v^{1/3}/10$ e $\overline{d}(G) < v^{1/3}/10$.

Observe que $\Delta(F_1) \leq 16\overline{d}(G)\overline{d}(F)$, pois caso contrário teríamos $2|A(F)| > 16\overline{d}(G)\overline{d}(F)(1 + |X_0|) \geq 2|A(F)|$, o que é uma contradição. Portanto $\Delta(F_1) \leq 4v^{2/3}/25$. Além disso, pelo Lema 4.4.2 anterior temos que $\Delta(H_1) \leq 4\overline{d}(G) \leq 2v^{1/3}/5$. Podemos assim usar o Teorema 2.1.6 (i), obtendo um empacotamento de F_1 e H_1 e conseqüentemente um empacotamento de F e H, o que contradiz o Lema 2.1.3, já que $H \in \min(\hat{P})$ e $F \in \min(\hat{P}^*)$.

□

Combinando o teorema anterior e o Teorema 4.4.1, obtém-se o resultado a seguir.

Teorema 4.4.4 *Para toda* $P \in \mathcal{P}_V$, *com* $|V| = v \geq 6 \times 10^9$, *vale que* $C^{Al}(P) \geq v^{4/3}/40$.

□

Capítulo 5

Problemas de seleção

A Conjectura 1.4.4 de Yao e Karp afirma que a complexidade aleatória de qualquer propriedade monotônica não-trivial sobre grafos de ordem v é $\Omega(v^2)$. Caso esta conjectura seja verificada, o ganho que se terá com a introdução de aleatoriedade em algoritmos que verificam propriedades monotônicas não-triviais de grafos se restringirá a no máximo uma constante. Uma pergunta que ocorre naturalmente é se a introdução de aleatoriedade em procedimentos computacionais implica em algum ganho *assintótico*. De fato, esta pergunta fundamental já era posta por Yao [41] em 1977.

Neste capítulo, iremos considerar algoritmos pararelos para os problemas de ordenação, seleção do mínimo e, mais genericamente, do k-ésimo menor elemento de um conjunto $X \subset \mathbb{Z}$, onde $|X| = n$ e $1 \leq k \leq n$. Vamos inicialmente fornecer um limite inferior de $\log\log n$, devido a Valiant [39], para a complexidade de tempo de *qualquer* algoritmo determinístico paralelo que utiliza n processadores para selecionar o mínimo de X no pior caso. Na seção seguinte, daremos uma versão melhorada do algoritmo paralelo aleatório de Reischuk [33] que seleciona o k-ésimo menor elemento de X (e, portanto, em particular o mínimo de X) em tempo médio *constante*. Observamos com isso um ganho substancial em relação aos algoritmos paralelos determinísticos com a introdução de aleatoriedade. Temos assim um caso concreto onde o poder da aleatoriedade é comprovado.

Na Seção 5.4, apresentaremos um teorema enunciado pela primeira vez por Yao em [41] e que nada mais é do que o análogo do Teorema 2.2.1 para ordenação. (Na realidade, esse teorema será válido para toda uma classe de problemas que definiremos como problemas de seleção.) Esse teorema será essencial na demonstração do

limite inferior para a complexidade do problema de ordenação aleatória em paralelo devido a Alon e Azar [4], que daremos na última seção deste capítulo.

5.1 Definição dos problemas e do modelo de computação

Sejam dados um conjunto finito não-vazio $X \subset \mathbb{Z}$ e um inteiro k tais que $1 \leq k \leq n = |X|$. Vamos definir formalmente os problemas MÍNIMO, SELEÇÃO$_k$ e ORDENAÇÃO, sendo que em MÍNIMO queremos o menor elemento do conjunto dado X, em SELEÇÃO$_k$ queremos selecionar o k-ésimo menor elemento de X, e em ORDENAÇÃO queremos ordenar X em ordem crescente. Vamos adotar o modelo sugerido por Yao em [41]. Primeiramente defina o posto $\rho_X(x)$ de um elemento $x \in X$ como sendo a quantidade de elementos de X que são menores ou iguais a x, ou seja $\rho_X(x) = |\{y \in X : y \leq x\}|$. Defina $I_{n,k}$ como sendo o conjunto de todas as k-uplas de inteiros distintos em $[n] = \{1, 2, \ldots, n\}$, ou seja $I_{n,k} = \{\langle i_1, i_2, \ldots, i_k\rangle : i_j \in [n],\ 1 \leq j \leq k,\ \text{e}\ i_j \neq i_l,\ 1 \leq j < l \leq k\}$. Um **problema de seleção** sobre X é dado por um subconjunto $P \subset I_{n,k}$ da maneira a seguir. Usando comparações binárias (i.e., comparações entre pares de elementos de X), queremos determinar uma seqüência de k elementos de X tal que a seqüência de seus postos pertença a P, ou seja queremos uma k-upla $\langle x_1, x_2, \ldots, x_k\rangle$ ($x_i \in X$, $1 \leq i \leq k$) cuja seqüência de postos $\langle \rho_X(x_1), \rho_X(x_2), \ldots, \rho_X(x_n)\rangle$ pertença a P. Com isso, temos que o problema ORDENAÇÃO é dado pelo conjunto que contém exatamente a n-upla $\langle 1, 2, \ldots, n\rangle$, e neste caso k evidentemente é igual a n. Por sua vez, o problema SELEÇÃO$_k$ é dado pelo seguinte conjunto de k-uplas: $\{\langle i_1, \ldots, i_k\rangle : i_j \in [n]\,(1 \leq j \leq k)\ \text{e}\ i_k = k\}$. Obviamente o problema MÍNIMO é o caso particular de SELEÇÃO$_k$ para $k = 1$.

Todas as nossas observações neste capítulo poderiam ser feitas considerando X como sendo um conjunto munido de uma ordem linear tal que a relação entre quaisquer dois elementos de X pudesse ser efetivamente computada através de uma comparação. Vamos nos restringir a subconjuntos V de $\mathbb{Z}$ por uma questão de comodidade e facilidade na notação. Observe que estamos supondo que os elementos da entrada são distintos dois-a-dois, não há perda de generalidade pois se não fossem poderíamos acrescentar uma informação extra ao final de cada elemento da entrada e considerar a ordem lexicográfica para fins de comparação.

Estaremos particularmente interessados em algoritmos *paralelos* que solucionam esses problemas (SELEÇÃO$_k$ e ORDENAÇÃO) e dentre esses algoritmos gostaríamos de comparar a complexidade dos determinísticos com a dos aleatórios. Observe que consideraremos apenas os algoritmos que se baseiam exclusivamente em comparações binárias.

O modelo de computação seqüencial usualmente utilizado para o estudo da complexidade de algoritmos para problemas de seleção é aquele em que se toma como base o número de comparações binárias feitas pelo algoritmo. Ignoramos aqui outros fatores que possam influenciar o tempo de execução como, por exemplo, a movimentação e o endereçamento dos dados, relegando inclusive o tempo gasto para tirar conclusões a partir de comparações já feitas. Neste caso estamos supondo que são as comparações que dominam todo o tempo de processamento. Essa suposição é razoável para os fins desejados, já que com este modelo um limite inferior encontrado será válido para qualquer algoritmo que se baseie em comparações, independentemente da máquina seqüencial que vá ser utilizada para executá-lo.

O modelo paralelo que adotaremos aqui foi introduzido por Valiant [39] e também só considera relevante o número de comparações feitas, ignorando fatores como a comunicação entre processadores e a comunicação entre um processador e a memória central (se houver uma). Assim se há m processadores disponíveis, iremos supor que eles estão sincronizados de modo que em um certo intervalo de tempo, que passaremos a chamar de **iteração**, cada um deles consegue executar uma comparação. Observe que um mesmo elemento do conjunto de entrada pode participar em mais de uma comparação numa mesma iteração. Ao final de uma iteração, o algoritmo verifica se já tem informação suficiente para fornecer a solução final do problema, e em caso negativo determina quais as comparações que deverão ser executadas na próxima iteração. Claramente, um limite inferior neste modelo também implica um limite inferior para qualquer algoritmo baseado em comparações para qualquer máquina paralela, mesmo para as mais poderosas que permitem leitura e escrita em uma mesma posição de memória por vários processadores simultaneamente (i.e., as usualmente conhecidas por CRCW PRAM, ver [24]). Definimos assim a **complexidade de tempo de pior caso** de um algoritmo como sendo o número máximo de iterações que esse algoritmo precisa executar para solucionar um problema P, ou seja, qualquer que seja a entrada do universo considerado, o algoritmo não precisa

mais do que esse número de iterações para resolvê-la. Iremos nos referir também à complexidade média de um algoritmo, ou seja ao número médio de iterações que o algoritmo precisa executar para solucionar um problema dado, supondo que as entradas estão distribuídas de acordo com uma certa distribuição de probabilidade. Em ambos os casos (pior caso ou caso médio), a complexidade de um *problema* é a complexidade do melhor algoritmo para resolvê-lo na categoria considerada.

No modelo seqüencial, o objetivo é minimizar o número de comparações que, neste caso, também corresponde ao número de iterações. No modelo paralelo, queremos minimizar o número de iterações bem como o número de processadores utilizados. De fato, se não tivéssemos a restrição sobre o número de processadores, poderíamos facilmente resolver qualquer problema de seleção em uma única iteração. Suponha que $|X| = n$ e que temos $m \geq \binom{n}{2}$ processadores disponíveis. Neste caso, podemos achar o posto de cada elemento de X simplesmente comparando cada elemento de X com todos os outros. Estaríamos fazendo um número total de comparações igual a $\binom{n}{2}$. Mas esse número de comparações é muito maior do que é geralmente necessário. De fato, sabemos por exemplo que para solucionar SELEÇÃO$_k$ e ORDENAÇÃO em seqüencial são suficientes $O(n)$ e $O(n \log n)$ comparações respectivamente (ver [1] e [29]).

Vamos agora introduzir um conceito que nos ajudará a avaliar a eficiência de um algoritmo paralelo. Dizemos que um algoritmo paralelo que resolve um problema P dado atinge **speed-up ótimo** se seu tempo de execução é $O(\text{Seq}(n)/m)$, onde n é o tamanho da instância do problema, m é o número de processadores utilizados, e $\text{Seq}(n)$ é a complexidade de tempo de pior caso do melhor algoritmo seqüencial que resolve P.

No problema SELEÇÃO$_k$, sabemos por extensa literatura no assunto (ver [1], [29]) que $\text{Seq}(n) = \Theta(n)$, onde n é o número de elementos do conjunto de entrada. Demonstraremos nas seções seguintes que *qualquer* algoritmo paralelo *determinístico* que utilize n processadores *não* atinge speed-up ótimo, pois terá tempo de execução $\Omega(\log \log n)$; enquanto que existe um algoritmo paralelo *aleatório* que usa n processadores e atinge speed-up ótimo, tendo tempo de execução igual a $O(1)$.

5.2 Um limite inferior para selecionar o mínimo em paralelo

Nesta seção, vamos demonstrar o limite inferior devido a Valiant [39] para a complexidade de pior caso do problema de se achar em paralelo o menor elemento de um conjunto $X \subset \mathbb{Z}$. Primeiramente, vamos enunciar um resultado clássico da teoria extremal de grafos, o teorema de Turán (ver [9]). Utilizaremos um corolário deste resultado, o Corolário 5.2.2 abaixo, na demonstração do teorema de Valiant.

Teorema 5.2.1 *Sejam n e $r \geq 2$ dois números naturais. Então todo grafo com n vértices e mais de $t_{r-1}(n)$ arestas contém um K^r, i.e., um grafo completo com r vértices, onde $t_q(n) = \binom{n}{2} - \sum_{i=0}^{q-1} \binom{n_i}{2}$, e $n_i = \lfloor (n+i)/q \rfloor$ para todo $0 \leq i < q$.*

□

Em particular, note que se o resto da divisão de n por q é igual a k então, com os n_i definidos acima,

$$A = \sum_{i=0}^{q-1} \binom{n_i}{2} = \frac{1}{2} \sum_{i=0}^{q-1} \left\lfloor \frac{n+i}{q} \right\rfloor \left(\left\lfloor \frac{n+i}{q} \right\rfloor - 1 \right) = \frac{q}{2} \left\lfloor \frac{n}{q} \right\rfloor \left(\left\lfloor \frac{n}{q} \right\rfloor - 1 \right) + k \left\lfloor \frac{n}{q} \right\rfloor,$$

enquanto que se definirmos, como de usual, $\binom{x}{2} = x(x-1)/2$ para $x \in \mathbb{R}$, então para q, $n \in \mathbb{N}$ temos que

$$\begin{aligned} B = q\binom{n/q}{2} &= \frac{q}{2} \left(\left\lfloor \frac{n}{q} \right\rfloor + \frac{k}{q} \right) \left(\left\lfloor \frac{n}{q} \right\rfloor + \frac{k}{q} - 1 \right) \\ &= \frac{q}{2} \left\lfloor \frac{n}{q} \right\rfloor \left(\left\lfloor \frac{n}{q} \right\rfloor - 1 \right) + k \left\lfloor \frac{n}{q} \right\rfloor + \frac{k}{2} \left(\frac{k}{q} - 1 \right). \end{aligned}$$

Portanto, temos que $A \geq B$ já que $0 \leq k < q$. Assim $t_q(n)$, como definido acima, é menor ou igual a $\binom{n}{2} - q\binom{n/q}{2}$ e, como uma conseqüência trivial do teorema de Turán, podemos garantir que todo grafo com n vértices e mais de $\binom{n}{2} - (r-1)\binom{n/(r-1)}{2}$ arestas $(r \geq 2)$ contém um K^r.

Corolário 5.2.2 *Todo grafo com n vértices e m arestas contém um conjunto independente com pelo menos $\lceil n^2/(2m+n) \rceil$ vértices.*

Demonstração

É imediato da observação acima que todo grafo G com n vértices e menos de $(q-1)\binom{n/(q-1)}{2}$ arestas contém um conjunto independente com q vértices.

Assim, basta verificar que $m < (r-1)\binom{n/(r-1)}{2}$ onde $r = \lceil n^2/(2m+n) \rceil$. Mas de fato,

$$\begin{aligned}(r-1)\binom{n/(r-1)}{2} &= \frac{n}{2}\left(\frac{n}{\lceil n^2/(2m+n)\rceil - 1} - 1\right) \\ &> \frac{n}{2}\left(\frac{n}{n^2/(2m+n)} - 1\right) \\ &= m.\end{aligned}$$

□

Podemos agora demonstrar o resultado de Valiant.

Teorema 5.2.3 *A complexidade de pior caso de qualquer algoritmo paralelo determinístico que acha o mínimo de um conjunto $X \subset \mathbb{Z}$, $|X| = n$, usando $m = n$ processadores, é maior ou igual a* $\log_2 \log_2 m - 0{,}67$.

Demonstração

Vamos considerar um algoritmo paralelo arbitrário que utiliza m processadores para o problema em questão. Seja C_i o conjunto de todos os elementos $x \in X$ para os quais, até o início da i-ésima iteração ($i > 0$), não foi encontrado um elemento $y \in X$ tal que $y < x$. Ou seja C_i é o conjunto dos candidatos a mínimo na i-ésima iteração. Claramente $C_1 = X$. Suponha que $|C_i| = c_i$ para todo $i \geq 1$.

Se durante a iteração i ocorrer uma comparação entre dois não-candidatos, ou seja, entre dois elementos em $X \setminus C_i$, trivialmente temos que o resultado dessa comparação não contribuirá para diminuir o número de candidatos. Como estamos interessados num limite inferior para o número mínimo de iterações, vamos supor que o algoritmo em questão não faz esse tipo de comparação. Note ainda que as comparações entre um não-candidato $x \in X \setminus C_i$ e um candidato $y \in C_i$, também podem ser evitadas pelo algoritmo, pois não diminuem o número de candidatos. De fato, afirmamos que se $C_i \neq \varnothing$ ($i \geq 1$) então para todo $x \in X \setminus C_i$ existe um elemento de C_i menor que x. A afirmação é trivial se notarmos que se $x_1 \in C_1 = X$ e $x_1 \notin C_i$ ($i > 1$), então numa iteração $1 \leq i_1 < i$, temos que x_1 foi comparado com um elemento x_2 tal que $x_2 < x_1$ e portanto, nessa iteração x_1 deixou de ser

candidato. Se x_2 pertence a X_i, então encontramos o elemento que afirmamos existir, caso contrário numa iteração $1 \leq i_2 < i$, temos que x_2 foi comparado com um elemento x_3 tal que $x_3 < x_2$. Se x_3 também não pertence a C_i, prosseguimos com o mesmo raciocínio. Como estamos supondo que a entrada do algoritmo é um conjunto finito de inteiros, vamos necessariamente encontrar um x_k $(1 \leq k \leq |X \setminus C_i| + 1)$ tal que $x_k < x_{k-1} < \ldots < x_2 < x_1$ e $x_k \in C_i$.

Assim, podemos supor que na i-ésima iteração o algoritmo faz comparacões apenas entre os elementos de C_i. Para obter o limite inferior desejado, vamos provar que mesmo após serem feitas m comparações entre apenas c_i elementos, existirá um conjunto suficientemente grande de candidatos que não foram comparados entre si e que, portanto, continuam a ser candidatos, desde que i seja suficientemente pequeno. Para isso, vamos representar as comparações feitas durante uma iteração $i \geq 1$ em termos de um grafo dirigido G_i. Cada vértice de G_i representa um elemento distinto de C_i, ou seja $V(G_i) = C_i$. A aresta dirigida $\vec{xy}$ pertence a $A(G_i)$ $(x, y \in V(G_i))$ se e somente se x e y foram comparados na i-ésima iteração e $x < y$. Claramente, para todo $i \geq 1$, vale que $|A(G_i)| \leq m$. Nesta formulação, o conjunto C_{i+1} de candidatos a mínimo na $(i+1)$-ésima iteração são representados por vértices de G_i que não têm arestas entrando, i.e. $x \in C_{i+1}$ se e somente se $\vec{yx} \notin A(G_i)$ para qualquer $y \in V(G_i)$. Note que os vértices que representam os elementos de C_{i+1} em G_i formam um conjunto independente em G_i.

Se G é um grafo, seja $\alpha(G)$ a cardinalidade de um conjunto independente máximo em G. Neste caso, temos que

$$c_{i+1} \geq \min\{\alpha(G) : G \text{ tem } c_i \text{ vértices e } m \text{ arestas}\}.$$

De acordo com o Corolário 5.2.2, temos então que $c_{i+1} \geq c_i^2/(2m + c_i) \geq c_i^2/3m$. Como $c_0 = n$, temos que $c_{i+1} \geq n^{2^i}/(3m)^{2^i-1}$ para todo i.

Obviamente, enquanto $c_j > 1$ $(j \geq 1)$ o algoritmo não pode decidir qual é o mínimo de X. Podemos portanto obter um limite inferior para o número de iterações determinando os $j \geq 1$ para os quais vale que $c_j > 1$, ou seja, para os quais $n^{2^j} = m^{2^j} > (3m)^{2^j-1}$. É fácil verificar que $c_j > 1$ enquanto $j \leq \log_2 \log_2 m - 0{,}67$. Assim, qualquer algoritmo precisa de pelo menos $\log_2 \log_2 m - 0{,}67$ iterações para determinar o mínimo de um conjunto com n elementos usando $m = n$ processadores.

□

5.3 Um algoritmo paralelo aleatório para SELEÇÃO$_k$

Nesta seção, vamos provar que existe um algoritmo *paralelo aleatório* que resolve o problema SELEÇÃO$_k$ num número constante de iterações. Inicialmente, vamos descrever um algoritmo paralelo aleatório SELPARAL que usa m processadores e que ao receber um conjunto X de entrada, com $|X| = n \leq m$, *quase* sempre devolve o k-ésimo menor elemento de X num número constante de iterações. Vamos provar que a probabilidade de SELPARAL falhar é $o(m^{-2})$. Nesses casos em que o algoritmo abortar sem nada devolver, vamos usar um outro algoritmo qualquer de seleção que resolve SELEÇÃO$_k$ em tempo $O(n^2)$. Chamando de SELPARAL$^+$ a esse algoritmo SELPARAL modificado que, ao invés de abortar, chama outro algoritmo, teremos que a complexidade de SELPARAL$^+$ será $O(1)$.

Vamos então agora descrever SELPARAL que utiliza m processadores e devolve com probabilidade $1 - o(m^{-2})$ o k-ésimo menor elemento de um conjunto $X \subset \mathbb{Z}$ dado, onde $|X| = n \leq m$, em tempo médio constante. Vamos neste algoritmo supor que $1 \leq k \leq \lceil n/2 \rceil$. Note que se quisermos achar o k'-ésimo menor elemento de X tal que $k' > \lceil n/2 \rceil$, basta simplesmente determinar o $(n - k' + 1)$-ésimo menor elemento de X na ordem reversa usando o nosso algoritmo, i.e. podemos redefinir a função posto como $\rho'_X(x) = |\{y \in X : y \geq x\}|$ para $x \in X$ e chamar o algoritmo que descreveremos a seguir com $k = n - k' + 1$.

A idéia do algoritmo é bastante simples. Primeiro tomamos um subconjunto aleatório $S \subset X$ tal que S seja suficientemente pequeno para que possamos ordená-lo em uma iteração. Mais precisamente, pegamos um S tal que $\binom{|S|}{2} \leq m$ pois neste caso podemos comparar cada elemento de S com todos os outros, descobrindo assim o posto de cada um deles. Esse conjunto S vai ser uma espécie de guia para que possamos decidir como dividir o conjunto X convenientemente. Se $k \leq 25n^{1/2} \log n$, então tomamos um conjunto A com os menores elementos de X e chamamos o algoritmo recursivamente com o mesmo k e o conjunto A no lugar de X. Se $25n^{1/2} \log n < k \leq \lceil n/2 \rceil$, então tomamos um conjunto A com os menores elementos de X e um conjunto $B \subseteq X \setminus A$ com uma certa quantidade de elementos de forma que o k-ésimo menor elemento de X esteja em B. Neste último caso, chamamos o algoritmo recursivamente com B no lugar de X e $k - |A|$ no lugar de k. Uma escolha "ruim" de S pode fazer com que o algoritmo aborte sem devolver o k-ésimo menor elemento. Escolheremos os tamanhos dos conjuntos envolvidos

convenientemente de forma que a probabilidade de S ser ruim seja pequena. (No algoritmo abaixo, iremos colocar um índice entre parênteses na frente de algumas desigualdades às quais faremos referência posteriormente.)

Algoritmo 5.3.1 (SELPARAL(X,k)) *Descrição: Recebe um conjunto $X \subset \mathbb{Z}$ com n elementos e um inteiro k $(1 \leq k \leq \lceil n/2 \rceil)$. Usa $m \geq n$ processadores. Devolve o k-ésimo menor elemento de X, ou aborta com probabilidade $o(m^{-2})$.*

Caso 1: $n \leq (2m)^{1/2}$

Neste caso, ordene X em um passo comparando cada $x \in X$ com todos os outros elementos de X. Devolva o k-ésimo menor elemento de X.

Caso 2: $n > (2m)^{1/2}$

Tome $p = 1/(2n)^{1/2}$.

Escolha $S \subseteq X$ aleatoriamente, pondo cada $x \in X$ em S com probabilidade p independentemente.

Se S assim escolhido for tal que (1) $\binom{|S|}{2} > m$ então aborte.

Caso contrário, ordene S em um passo comparando cada $y \in S$ com todos os outros elementos de S. Suponha que $S = \{y_1, y_2, \ldots, y_s\}$, onde $y_1 \leq y_2 \leq \ldots \leq y_s$, e considere os dois subcasos a seguir.

Caso 2.1 $k \leq 25n^{1/2} \log n$

Tome $i = \lceil n^{1/4}/4 \rceil$. Se (2) $|S| < i$, aborte.

Tome $A = \{x \in X : x \leq y_i\}$.

Se (3) $|A| > l = \lceil n^{3/4} \rceil$ ou (4) $|A| < k$, aborte.

Caso contrário, chame recursivamente o algoritmo com o mesmo k e com A no lugar de X.

Caso 2.2 $k > 25n^{1/2} \log n$

Tome $\mu = kp$ e $d = ((25/2) \log n)^{1/2}$.

Tome $i = \lceil \mu - d\mu^{1/2} \rceil$ e $j = \lceil \mu + d\mu^{1/2} \rceil$. Se (5) $|S| < j$, aborte.

Tome $A = \{x \in X : x < y_i\}$. Tome $B = \{x \in X : y_i \leq x \leq y_j\}$.

Se (6) $|B| > l = \lceil 2^{5/2} d(n\mu)^{1/2} \rceil$ ou (7) o k-ésimo menor elemento de X não pertence a B, aborte.

Caso contrário, chame recursivamente o algoritmo com B no lugar do conjunto X e $k - |A|$ no lugar de k.

□

Provaremos inicialmente que o algoritmo SELPARAL sempre pára em tempo constante. Como o número de elementos do conjunto de entrada é sempre menor ou igual ao número de processadores, note que cada chamada do algoritmo pode ser executada em tempo constante. Vamos verificar então quantas chamadas recursivas são necessárias para que o algoritmo pare. Suponha que inicialmente chamamos SELPARAL com um conjunto X_1 tal que $|X_1| = n_1$ e com $k = k_1$. Se o Caso 2.1 ocorrer numa chamada $i \geq 1$, com um conjunto de entrada X_i tal que $|X_i| = n_i$, então na próxima chamada o conjunto de entrada X_{i+1} será tal que $|X_{i+1}| \leq \lceil n_i^{3/4} \rceil$. Caso contrário, i.e. no Caso 2.2, teremos $X_{i+1} \leq \lceil 2^{5/2}(n_i\mu_i)^{1/2}d_i \rceil \leq 20(\log n_i)^{1/2}n_i^{3/4}$. Ou seja, se numa chamada i o conjunto de entrada X_i tem tamanho n_i, então o conjunto de entrada da próxima chamada recursiva terá seu tamanho reduzido a pelo menos $20(\log n_i)^{1/2}n_i^{3/4}$. Portanto já no início da quarta chamada o conjunto de entrada terá tamanho igual a $o(n_1^{1/2})$, onde $n_1 = |X_1|$, e portanto valerá o Caso 1. Ao atingir o Caso 1, o algoritmo encontra o k-ésimo elemento de X e pára. São portanto necessárias no máximo quatro chamadas para que o algoritmo termine sua execução. Claramente, se o algoritmo abortar serão necessárias menos de quatro chamadas.

Vamos provar agora que o algoritmo SELPARAL deixa de encontrar o k-ésimo menor elemento de X com probabilidade $o(m^{-2})$, analisando cada um dos casos em que o algoritmo é obrigado a abortar. Vamos usar o Lema 4.2.1 para achar limites superiores para as probabilidades relevantes.

Lema 5.3.2 *Se o algoritmo* SELPARAL *acima recebe um conjunto de entrada X com $|X| = n \leq m$, então ele encontra o k-ésimo elemento de X com probabilidade $1 - o(m^{-2})$.*

Demonstração

Para esta prova, suponha que o conjunto de entrada é inicialmente $X = \{x_1, x_2, \ldots, x_n\}$ onde $x_1 \leq x_2 \leq \ldots \leq x_n$.

Vamos analisar os sete casos em que o algoritmo aborta e concluir que a probabilidade de pelo menos um deles acontecer é $o(m^{-2})$. Lembre que, para os casos considerados, vale que $n > (2m)^{1/2}$.

Em (1), temos que $\mathbf{P}\left\{\binom{|S|}{2} > m\right\} \leq \mathbf{P}\left\{|S| \geq (2m)^{1/2}\right\} = \mathbf{P}\left\{|S| \geq 2(m/2)^{1/2}\right\}$. Como $\mathbf{E}(|S|) = np = (n/2)^{1/2} \leq (m/2)^{1/2}$, temos que

$$\mathbf{P}\left\{\binom{|S|}{2} > m\right\} \leq \mathbf{P}\{|S| \geq 2\mathbf{E}(|S|)\} \leq \exp\{-(n/2)^{1/2}/3\} = \exp\{-\Omega(m^{1/4})\},$$

onde a penúltima desigualdade é dada pelo Lema 4.2.1.

Vamos calcular a probabilidade de S ser menor que i em (2). Lembre que $\mathbf{E}(|S|) = np = (n/2)^{1/2}$, assim com o Lema 4.2.1, temos que

$$\begin{aligned}\mathbf{P}(|S| < i) &\leq \mathbf{P}\{|S| \leq n^{1/4}/4\} \leq \mathbf{P}\{|S| \leq \mathbf{E}(|S|)/2\} \\ &\leq \exp\{(n/2)^{1/2}/8\} \leq \exp\{-\Omega(m^{-1/4})\}.\end{aligned}$$

Para calcular a probabilidade de $|A|$ ser maior que l em (3), defina $Z = |S \cap \{x_1, x_2, \ldots, x_l\}|$. Temos assim que $\mathbf{E}(Z) = lp \geq n^{3/4}(2n)^{-1/2} \geq n^{1/4}/2$. De onde vem que

$$\mathbf{P}(|A| > l) = \mathbf{P}(x_l < y_i) = \mathbf{P}(Z < i) \leq \mathbf{P}(Z \leq n^{1/4}/4) \leq \mathbf{P}(Z \leq \mathbf{E}(Z)/2).$$

Usando novamente o Lema 4.2.1, concluímos que $\mathbf{P}(|A| > l) \leq \exp(-n^{1/4}/16) \leq \exp(-\Omega(m^{1/8}))$.

Para (4), defina $Y = |S \cap \{x_1, x_2, \ldots, x_k\}|$. Vamos supor que $k = \lfloor 25n^{1/2}\log n \rfloor$, já que isso só aumenta a probabilidade de $|A|$ ser menor que k e queremos um limite superior para tal probabilidade. Assim, temos que

$$\mathbf{P}(|A| < k) = \mathbf{P}(y_i < x_k) \leq \mathbf{P}(Y \geq i) = \mathbf{P}(Y \geq \lceil n^{1/4}/4 \rceil) \leq \mathbf{P}(Y \geq 2\mathbf{E}(Y)),$$

já que $\mathbf{E}(Y) = kp \leq 2^{-1/2}25\log n \leq n^{1/4}/8$, a última desigualdade valendo para $n \geq 2 \times 10^{16}$. Observe ainda que, tomando $k = \lfloor 25n^{1/2}\log n \rfloor$, temos $\mathbf{E}(Y) \geq \lfloor (25/2^{1/2})\log n \rfloor \lfloor (2n)^{1/2} \rfloor (2n)^{-1/2} \geq (25/2)\log n$. Usando o Lema 4.2.1, temos que

$$\mathbf{P}(|A| < k) \leq \exp\{-(25/6)\log n\} \leq \exp\{-(25/6)\log(2m)^{1/2}\} = o(m^{-2}).$$

Para (5), (6) e (7), note que $k > 25n^{1/2}\log n$ e portanto $\mu \geq 2^{-1/2}25\log n$.

Note que em (5) para $k \geq 25n^{1/2}\log n$, temos que $i = \lceil \mu(1 - d\mu^{-1/2})\rceil \geq \lceil \mu(1-2^{-1/4})\rceil \geq 1$. Portanto y_i sempre pertence a S se y_j pertence a S. Vamos então calcular a probabilidade de termos $y_j \notin S$, ou seja a probabilidade de valer $|S| < j$ em (5). Note que

$$j \leq \lceil \mu(1 + 2^{-1/4})\rceil \leq \lceil 1{,}84 \cdot kp\rceil \leq \left\lceil 1{,}84 \left\lceil \frac{n}{2} \right\rceil \frac{1}{(2n)^{1/2}} \right\rceil \leq 0{,}93 \cdot \left(\frac{n}{2}\right)^{1/2}.$$

Portanto, usando o Lema 4.2.1, temos

$$\mathbf{P}(|S| < j) \leq \mathbf{P}\left\{|S| \leq 0{,}93\mathbf{E}(|S|)\right\} \leq \exp\{0{,}0049(n/2)^{1/2}/2) = \exp\{-\Omega(m^{1/4})\}.$$

Para calcular a probabilidade de $|B|$ ser menor que l em (6), suponha que $y_i = x_{i'}$. Equivalentemente, i' é o posto de y_i no conjunto X. Ponha $W = |S \cap \{x_q \in X : i' \leq q < i' + l\}|$. Note que $|B| > l$ se e somente se $y_i = x_{i'}$ e $y_j \geq x_{i'+l}$, mas isto acontece se e somente se $W \leq j - i$. Assim, temos que $\mathbb{P}(|B| > l) = \mathbf{P}(W \leq j - i) \leq \mathbf{P}(W \leq 2d\mu^{1/2})$. Como $\mathbb{E}(W) = lp \geq 4d\mu^{1/2} \geq 50\log n$, usando novamente o Lema 4.2.1, temos que

$$\begin{aligned} \mathbf{P}(|B| > l) &\leq \mathbf{P}\{W \leq \mathbf{E}(W)/2\} \leq \exp\{-4d\mu^{1/2}/8\} \\ &\leq \exp\{-(25/4)\log n\} \leq \exp\{-(25/4)\log(2m)^{1/2}\} \\ &\leq m^{-3}. \end{aligned}$$

Finalmente, para calcular a probabilidade de x_k não pertencer a B em (7), tome $U = |S \cap \{x_1, x_2, \ldots, x_k\}|$. Claramente, temos que $x_k \in B$ se e somente se $y_i \leq x_k \leq y_j$, o que ocorre se e somente se $i \leq U \leq j$. Temos então que

$$\begin{aligned} \mathbf{P}(x_k \notin B) &\leq \mathbf{P}\{U \leq \mu - d\mu^{1/2}\} + \mathbb{P}\{U \geq \mu + d\mu^{1/2}\} \\ &= \mathbf{P}\{U \leq \mu(1 - d\mu^{-1/2})\} + \mathbb{P}\{U \geq \mu(1 + d\mu^{-1/2})\}. \end{aligned}$$

Como $\mathbf{E}(U) = kp = \mu \geq 2^{-1/2}25\log n$ e $0 \leq d\mu^{-1/2} \leq 2^{-1/4}$, com o Lema 4.2.1 temos que

$$\begin{aligned} \mathbf{P}(x_k \notin B) &\leq \exp\{-(d\mu^{-1/2})^2\mu/2\} + \exp\{-(d\mu^{-1/2})^2\mu/3\} \\ &= \exp\{-(25/4)\log n\} + \exp\{-(25/6)\log n\} \\ &\leq \exp\{-(25/4)\log(2m)^{1/2}\} + \exp\{-(25/6)\log(2m)^{1/2}\} \\ &= o(m^{-2}). \end{aligned}$$

Assim a probabilidade de qualquer um dentre (1) a (7) ocorrer numa chamada recursiva é $o(m^{-2})$. Como o algoritmo nunca faz mais do que quatro chamadas, temos que a probabilidade total do algoritmo abortar antes de encontrar o k-ésimo elemento é também $o(m^{-2})$. □

Com esse resultado, podemos agora facilmente demonstrar o seguinte teorema.

Teorema 5.3.3 *Existe um algoritmo aleatório paralelo que soluciona o problema* SELEÇÃO$_k$ *para entradas com n elementos usando $m = n$ processadores em tempo constante.*

Demonstração

Tome um algoritmo determinístico A que utiliza no máximo m processadores e soluciona o problema SELEÇÃO$_k$ com no máximo n^2 iterações se o conjunto de entrada tem n elementos. Observe que podemos obter facilmente um algoritmo A assim, pois até mesmo um algoritmo seqüencial trivial que compare cada elemento da entrada com todos os outros consegue resolver SELEÇÃO$_k$ em $\binom{n}{2}$ iterações.

Considere o algoritmo SELPARAL$^+$ que funciona exatamente como o algoritmo SELPARAL acima mas que, nos casos em que SELPARAL aborta, chama o algoritmo A.

O tempo médio de execução de SELPARAL$^+$ será então igual ao tempo de execução de SELPARAL, ou seja $\Theta(1)$, vezes a probabilidade de SELPARAL não abortar mais o tempo de execução de A vezes a probabilidade de SELPARAL abortar, ou seja $\Theta(1)(1 - o(m^{-2})) + n^2 o(m^{-2}) = O(1)$.

□

Por completude, enunciamos o seguinte corolário do resultado acima.

Teorema 5.3.4 *Existe um algoritmo aleatório paralelo que soluciona o problema* ORDENAÇÃO *para entradas com n elementos usando $m \geq n$ processadores em tempo $O(\log n)$.*

Demonstração

Apenas esboçamos um argumento. Pelo Teorema 5.3.3 acima, podemos achar a mediana de um conjunto dado X usando no máximo $|X|$ processadores em tempo

constante. Mas então claramente existe um algoritmo de divisão e conquista que ordena um conjunto com n elementos em $O(\log n)$ iterações.

□

5.4 Ordenação

Sabemos por resultados anteriores (ver [5] e [6]) que todo algoritmo determinístico que usa $m \geq n$ processadores para ordenar um conjunto de entrada com n elementos, baseado em comparações, precisa de $\Omega\{(\log n)/\log(1 + m/n)\}$ iterações. Ou seja, qualquer algoritmo determinístico que use $m \gg n$ processadores não atinge speed-up ótimo. Observe que para $m \leq n$ o circuito de ordenação de Ajtai, Komlós e Szemerédi (ver [2], [3]) estabelece que $\Theta((n/m)\log n)$ iterações são suficientes para solucionar esse problema.

Devido à semelhança dos objetivos envolvidos nos problemas de ordenação e seleção do mínimo, ocorre naturalmente a pergunta sobre a existência de um algoritmo aleatório que usa $m \gg n$ processadores e atinge speed-up ótimo. Vamos provar nesta seção e na próxima que este infelizmente não é o caso.

Por comodidade na notação, iremos supor que o conjunto de entrada X para o problema ORDENAÇÃO é dado na forma de um vetor, ou mais precisamente na forma de uma n-upla que apresenta os elementos de X numa ordem arbitrária. Vamos então fixar um conjunto $X \subset \mathbb{Z}$, $X = \{x_1, x_2, \ldots, x_n\}$, de entrada e verificar quantas comparações são necessárias para ordená-lo. Dada uma permutação σ sobre $[n]$, denotaremos por X_σ o vetor $\langle X_{\sigma(i)}\rangle_{i=1}^n = \langle x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(n)}\rangle$. Assim, dado um vetor $V = X_{\sigma_0}$ (onde σ_0 é evidentemente uma permutação), queremos obter uma permutação σ_1 tal que $\sigma_1(V) = X_{\sigma_1\sigma_0} = \langle x_{\sigma_1(\sigma_0(i))}\rangle_1^n$ tenha seqüência de postos igual a $\langle 1, 2, \ldots, n\rangle$. Vamos denotar por S_n o conjunto de todas as permutações sobre $[n]$. Denote por $\mathcal{V}[n]$ todas as n-uplas possíveis com os elementos de X, ou seja $\mathcal{V}_{[n]} = \{X_\sigma : \sigma \in S_n\}$.

Inicialmente, vamos redefinir o conceito de árvores de decisão da Definição 1.2.1 para adaptá-las ao problema de ordenação seqüencial.

Definição 5.4.1 *Seja $n \geq 2$ um inteiro. Uma* **árvore de decisão** *é uma árvore binária com raiz e pelo menos duas folhas tal que*

(i) todo nó interno é rotulado com um par ordenado (i, j)*, onde* $1 \leq i < j \leq n$*;*

(ii) todo nó interno tem uma aresta saindo com rótulo '$<$' *e outra com* '$>$' *(vamos convencionar que a aresta da "esquerda" tem rótulo* '$<$' *e a da "direita" tem rótulo* '$>$'*);*

(iii) em todo caminho da raiz até uma folha, os rótulos de quaisquer dois nós internos são distintos;

(iv) cada folha é rotulada com uma permutação $\sigma_1 \in S_n$*.*

A interpretação dessa definição é bastante simples. Suponha que a árvore de decisão A representa um certo algoritmo determinístico seqüencial de ordenação que recebe um vetor $V = \langle V_i \rangle_1^n = X_{\sigma_0} = \langle x_{\sigma_0(i)} \rangle_1^n$ como entrada. Simulamos o algoritmo visitando os nós da árvore de decisão, começando pela raiz, até atingirmos uma folha. Ao visitarmos um nó interno da árvore com rótulo (i, j), comparamos $x_{\sigma_0(i)}$ com $x_{\sigma_0(j)}$. Se $x_{\sigma_0(i)} < x_{\sigma_0(j)}$, então vamos para o filho esquerdo desse nó na árvore A; caso contrário, visitamos o filho direito. Ao chegarmos numa folha, temos uma permutação $\sigma_1 \in S_n$ tal que $\sigma_1(V) = X_{\sigma_1\sigma_0}$ tem a seqüência de postos $\langle 1, 2, \ldots, n \rangle$. Claramente, todo algoritmo determinísitco de ordenação seqüencial baseado em comparações que não faz comparações redundantes, tem uma árvore de decisão equivalente.

Vamos agora dar uma definição generalizando o conceito de árvore de decisão para algoritmos paralelos determinísticos de ordenação.

Definição 5.4.2 *Sejam dados dois inteiros* $m > 0$ *e* $n \geq 2$*. Uma* m**-árvore de decisão** *é uma árvore com raiz e pelo menos* 2^m *folhas tal que*

(i) todo nó interno tem exatamente 2^m *filhos e é rotulado com uma seqüência de* m *pares ordenados* $(i_1, j_1), (i_2, j_2), \ldots, (i_m, j_m)$*, onde* $i_k, j_k \in \mathbb{Z}$ *e* $1 \leq i_k < j_k \leq n$ *para* $1 \leq k \leq m$*;*

(ii) cada aresta que sai de um nó interno tem uma seqüência em $\{'<', '>'\}^m$ *como rótulo;*

(iii) em todo caminho da raiz até uma folha, quaisquer pares ordenados dos rótulos de quaisquer dois nós internos distintos são distintos;

(iv) cada folha é rotulada com uma permutação $\sigma_1 \in S_n$*.*

Neste caso, a interpretação é análoga à do caso seqüencial, mas ao invés de fazermos uma comparação a cada nó da árvore que visitamos, fazemos m comparações. Suponha que um algoritmo determinístico paralelo que utiliza m processadores recebe um vetor de inteiros $V = \langle V_i \rangle_1^n = X_{\sigma_0} \in \mathcal{V}_{[n]}$ como entrada. Vamos ver como uma m-árvore de decisão representa tal algoritmo. Começamos visitando a raiz da m-árvore. Ao visitarmos um nó da árvore, fazemos as m comparações entre elementos de V cujos índices são indicados pelos m pares do rótulo do nó. Essas m comparações têm 2^m resultados possíveis, e cada resultado desses corresponde a uma seqüência de '$<$' e '$>$'. Devemos então ir para o filho do nó que é ligado na m-árvore a esse nó através da aresta rotulada com a seqüência de '$<$' e '$>$' correspondente. Note ainda que não exigimos nesta definição que o rótulo de um nó interno tenha os pares ordenados distintos dois-a-dois. Isso se deve ao fato do algoritmo que a m-árvore representa poder ter processadores ociosos numa certa iteração, e numa m-árvore de decisão vamos indicar tal situação como se vários processadores estivessem fazendo a mesma comparação. Sendo assim, algumas subárvores da m-árvore não terão sentido algum, já que estarão supondo resultados distintos para uma mesma comparação. Observe que com essa liberdade, teremos que dado um algoritmo determinístico que utiliza m processadores para ordenar um vetor de entrada com n elementos, pode existir mais de uma m-árvore de decisão que representa a execução desse algoritmo.

Assim como no caso dos grafos analisado nos capítulos anteriores, cada entrada (i.e. cada vetor) está associada a uma única folha de uma m-árvore. De fato, simulemos uma m-árvore de decisão A com um vetor $V \in \mathcal{V}_{[n]}$ como entrada, como sugerimos acima. A cada nó interno que visitamos, fazemos m comparações entre pares de elementos de V. Essas m comparações têm 2^m resultados possíveis. Mas evidentemente os elementos de V satisfazem exatamente um desses resultados e portanto não há dúvida quanto à escolha do próximo vértice a ser visitado. Observe que isso é verdade mesmo para os nós internos cujos rótulos têm pares de índices repetidos. Concluímos então que dada uma m-árvore de decisão A, qualquer vetor de entrada $V \in \mathcal{V}_{[n]}$ determina um único caminho em A da raiz até uma folha de A. Note ainda que dado um algoritmo de ordenação que usa m processadores e é representado por duas m-árvores A_1 e A_2 distintas, vale que para qualquer $V \in \mathcal{V}_{[n]}$ o comprimento do caminho que V determina em A_1 é igual ao comprimento do

caminho que V determina em A_2.

Assim, dada uma m-árvore de decisão A e um vetor $V \in \mathcal{V}_{[n]}$, definimos custo(A, V) como sendo o comprimento do caminho em A da raiz até a folha de A associada a V. Graças à observação final do parágrafo anterior, temos que custo assim definido não depende da escolha da m-árvore que escolhermos para representar o algoritmo. Podemos interpretar custo(A, V) como sendo o número de iterações que o algoritmo representado pela m-árvore de decisão A precisa para ordenar esse vetor V.

Vamos chamar de $\mathrm{Ord}(n, m)$ o problema de ordenar vetores com n inteiros distintos através de algoritmos paralelos que usam m processadores. Com isso podemos definir a complexidade média de pior caso $\mathcal{C}^{\mathcal{D}}(\mathrm{Ord}(n, m))$ e a complexidade aleatória $\mathcal{C}^{\mathcal{AI}}(\mathrm{Ord}(n, m))$ para o problema de ordenação em paralelo de maneira análoga à de propriedades de grafos no Capítulo 1. Primeiramente, seja $\mathcal{A}_{\mathrm{Ord}(n,m)}$ o conjunto de todas as m-árvores de decisão que resolvem o problema de ordenação de vetores de $\mathcal{V}_{[n]}$.

Definição 5.4.3 *Definimos a* **complexidade determinística média de pior caso** $\mathcal{C}^{\mathcal{D}}(\mathrm{Ord}(n, m))$ *do problema* $\mathrm{Ord}(n, m)$ *pondo*

$$\mathcal{C}^{\mathcal{D}}(\mathrm{Ord}(n, m)) = \sup_{\alpha} \min_{A} \mathbb{E}_{\alpha}(\mathrm{custo}(A, V)) = \sup_{\alpha} \min_{A} \sum_{V \in \mathcal{V}_{[n]}} \alpha(V)\, \mathrm{custo}(A, V),$$

onde o supremo é sobre as distribuições de probabilidade α sobre $\mathcal{V}_{[n]}$, e o mínimo é sobre $A \in \mathcal{A}_{\mathrm{Ord}(n,m)}$.

Definição 5.4.4 *Um* **algoritmo aleatório** *R para o problema de $\mathrm{Ord}(n, m)$ é dado por uma distribuição de probabilidade q_R sobre $\mathcal{A}_{\mathrm{Ord}(n,m)}$.*

Definição 5.4.5 *A* **complexidade aleatória** *de $\mathrm{Ord}(n, m)$ é*

$$\begin{aligned} \mathcal{C}^{\mathcal{AI}}(\mathrm{Ord}(n, m)) &= \inf_{R} \max_{V} \mathbb{E}_{q_R}(\mathrm{custo}(A, V)) \\ &= \inf_{R} \max_{V} \sum_{A} q_R(A)\, \mathrm{custo}(A, V), \end{aligned}$$

onde o ínfimo é sobre os algoritmos aleatórios R para $\mathrm{Ord}(n, m)$ dados por q_R, o máximo é sobre $V \in \mathcal{V}_{[n]}$, e a soma é sobre $A \in \mathcal{A}_{\mathrm{Ord}(n,m)}$.

Note que nestas definições também podemos substituir o ínfimo e o supremo por mínimo e máximo, respectivamente. Além disso, afirmamos que $\mathcal{C}^{\mathcal{D}}(\mathrm{Ord}(n,m)) = \mathcal{C}^{\mathcal{AI}}(\mathrm{Ord}(n,m))$. De fato, a demonstração do Teorema 2.2.1 vale aqui também, já que $\mathrm{custo}(A,V) > 0$ para qualquer m-árvore de decisão A e qualquer vetor $V \in \mathcal{V}_{[n]}$. Este resultado também vale para o caso de ordenação seqüencial, pois claramente se $m = 1$ então as m-árvores de decisão equivalem às árvores da Definição 5.4.1. Mais genericamente, observe que os conceitos de complexidade média de pior caso e complexidade aleatória, bem como a sua igualdade, são válidos para qualquer problema cujos conjuntos de dados de entrada e de algoritmos de resolução sejam finitos.

Como um exemplo de aplicação da igualdade entre as complexidades aleatória e determinística para problemas de seleção, observemos o seguinte corolário do Teorema 5.3.3.

Teorema 5.4.6 *Para qualquer distribuição de entrada α, existe um algoritmo paralelo determinístico $A = A(\alpha)$ que soluciona o problema* SELEÇÃO$_k$ *para entradas com n elementos, usando $m \geq n$ processadores, em tempo médio constante.*

Demonstração

Seja $\mathrm{Sel}_k(n,m)$ o problema de solucionar SELEÇÃO$_k$ usando m processadores e tendo como entrada conjuntos com n elementos . Sabemos que se $m \geq n$, temos que $\mathcal{C}^{\mathcal{AI}}(\mathrm{Sel}_k(n,m)) = O(1)$, onde esta complexidade é definida da forma natural. Assim, o nosso resultado decorre imediatamente da relação

$$\sup_{\alpha} \min_{A} \mathsf{E}_{\alpha}(\mathrm{custo}(A,V)) = \mathcal{C}^{\mathcal{D}}(\mathrm{Sel}_k(n,m)) = \mathcal{C}^{\mathcal{AI}}(\mathrm{Sel}_k(n,m)) = O(1).$$

□

Tendo em vista a extensa bibliografia disponível sobre ordenação (veja por exemplo [1], [29]), apenas com a igualdade entre as complexidades aleatória e determinística acima já podemos derivar alguns resultados interessantes. Por exemplo, existem algoritmos determinísticos seqüenciais de ordenação cujo tempo médio de execução é $O(n \log n)$ qualquer que seja a distribuição de entrada sobre $\mathcal{V}_{[n]}$ (os algoritmos MERGESORT e HEAPSORT por exemplo). Com isso temos que $\mathcal{C}^{\mathcal{AI}}(\mathrm{Ord}(n,1)) = \mathcal{C}^{\mathcal{D}}(\mathrm{Ord}(n,1)) = O(n \log n)$. Por outro lado, a partir de

árvores de decisão para ordenação, não é difícil concluir que o tempo médio de qualquer algoritmo determinístico é $\Omega(n \log n)$, supondo que as entradas são distribuídas uniformemente. Assim temos que $\mathcal{C}^{\mathcal{AI}}(\mathrm{Ord}(n,1)) = \Omega(n \log n)$ e portanto a introdução de passos aleatórios em algoritmos de ordenação seqüencial não traz ganhos assintoticamente significativos. Além disso, podemos facilmente definir um algoritmo aleatório cujo tempo médio é $\Theta(n \log n)$. De fato, lembramos que se todas as entradas possíveis são distribuídas uniformemente, o algoritmo QUICKSORT tem tempo médio de execução igual a $\Theta(n \log n)$. Assim, seja QSALEAT o algoritmo aleatório que, ao receber uma entrada $V = \langle V_i \rangle_1^n \in \mathcal{V}[n]$, gera aleatoriamente uma permutação σ de $[n]$ (de forma que todas as permutações sobre $[n]$ sejam igualmente prováveis), e em seguida chama o QUICKSORT para ordenar $\sigma(V) = \langle V_{\sigma(i)} \rangle_1^n$. Dessa maneira a complexidade de QSALEAT é claramente $\Theta(n \log n)$.

A idéia por trás do QSALEAT envolve uma certa simetria do problema de ordenação que será muito útil a seguir. Essa simetria se resume no seguinte: dada uma m-árvore (ou uma árvore, no caso seqüencial) de decisão para ordenação, qualquer permutação que aplicarmos aos índices que compõem os rótulos dessa árvore nos fornecerá uma nova m-árvore (uma árvore, respectivamente) de decisão para ordenação. Essa simetria nos permite demonstrar o seguinte teorema enunciado por Yao em [41]. (Na realidade, essa simetria é uma propriedade de toda a classe de problemas de seleção. Poderíamos então ter definido m-árvores de decisão para problemas de seleção em geral e o resultado a seguir também seria válido neste caso.)

Teorema 5.4.7 *Seja d_u a distribuição de probabilidade uniforme sobre $\mathcal{V}_{[n]}$. Então vale que $\mathcal{C}^{\mathcal{AI}}(\mathrm{Ord}(n,m)) = \min_A \mathsf{E}_{d_u}(\mathrm{custo}(A,V))$, onde o mínimo é sobre todas as m-árvores de decisão $A \in \mathcal{A}_{\mathrm{Ord}(n,m)}$.*

Demonstração

Vamos fixar uma m-árvore de decisão $A_0 \in \mathcal{A}_{\mathrm{Ord}(n,m)}$ para a qual vale que $\min_{A \in \mathcal{A}_{\mathrm{Ord}(n,m)}} \mathsf{E}_{d_u}(\mathrm{custo}(A,V)) = \mathsf{E}_{d_u}(\mathrm{custo}(A_0,V))$. Observe primeiramente que $\mathcal{C}^{\mathcal{AI}}(\mathrm{Ord}(n,m)) = \mathcal{C}^{\mathcal{D}}(\mathrm{Ord}(n,m)) \geq \mathsf{E}_{d_u}(\mathrm{custo}(A_0,V))$. Assim o nosso objetivo é provar a desigualdade reversa.

No que segue, iremos denotar por $\sigma(A_0)$ a m-árvore de decisão que se obtém a partir de A_0 permutando-se os rótulos dos nós de A_0 de acordo com σ. Por exemplo, se a raiz de A_0 tem rótulo $(i_1,j_1),(i_2,j_2),\ldots,(i_m,j_m)$, então a raiz de $\sigma(A_0)$ terá rótulo $(\sigma(i_1),\sigma(j_1)),(\sigma(i_2),\sigma(j_2)),\ldots,(\sigma(i_m),\sigma(j_m))$.

Seja $\mathcal{A}_0 \subseteq \mathcal{A}_{\mathrm{Ord}(n,m)}$ o conjunto de todas as árvores de decisão $\sigma(A_0)$ para cada $\sigma \in S_n$. Defina a distribuição de probabilidade q_u sobre $\mathcal{A}_{\mathrm{Ord}(n,m)}$ pondo

$$q_u(A) = \begin{cases} 1/|\mathcal{A}_0| & \text{se } A \in \mathcal{A}_0 \\ 0 & \text{caso contrário.} \end{cases}$$

Seja R_u o algoritmo aleatório dado pela distribuição q_u e seja $V_0 \in \mathcal{V}_{[n]}$ tal que $\max_V \mathsf{E}_{q_u}(\mathrm{custo}(A,V)) = \mathsf{E}_{q_u}(\mathrm{custo}(A,V_0))$. Temos então que

$$\begin{aligned} \mathsf{E}_{q_u}(\mathrm{custo}(A,V_0)) &= \frac{1}{|\mathcal{A}_0|}\sum_{A\in\mathcal{A}_0}\mathrm{custo}(A,V_0) \\ &= \frac{1}{|S_n|}\sum_{\sigma\in S_n}\mathrm{custo}(\sigma(A_0),V_0) \\ &= \frac{1}{|S_n|}\sum_{\sigma\in S_n}\mathrm{custo}(A_0,\sigma(V_0)) \\ &= \frac{1}{n!}\sum_{\sigma\in S_n}\mathrm{custo}(A_0,\sigma(V_0)) \\ &= \frac{1}{n!}\sum_{V\in\mathcal{V}_{[n]}}\mathrm{custo}(A_0,V) \\ &= \mathsf{E}_{d_u}(\mathrm{custo}(A_0,V)). \end{aligned}$$

Assim, temos a igualdade desejada, pois

$$\begin{aligned} C^{\mathcal{A}l}(\mathrm{Ord}(n,m)) &= C^{\mathcal{D}}(\mathrm{Ord}(n,m)) \geq \mathsf{E}_{d_u}(\mathrm{custo}(A_0,V)) \\ &= \mathsf{E}_{q_u}(\mathrm{custo}(A,V_0)) \geq C^{\mathcal{A}l}(\mathrm{Ord}(n,m)). \end{aligned}$$

□

Intuitivamente, esse teorema nos diz que nenhuma distribuição de probabilidade é mais "difícil" que a distribuição uniforme. Assim, através do estudo de algoritmos ótimos para a distribuição uniforme, podemos encontrar limitantes para $C^{\mathcal{A}l}(\mathrm{Ord}(n,m)) = C^{\mathcal{D}}(\mathrm{Ord}(n,m))$.

5.5 Um limite inferior para a complexidade aleatória de $\mathrm{Ord}(n, m)$

Nesta última seção, iremos demonstrar um limite inferior de $\Omega\{(\log n)/\log(1+m/n)\}$ para a complexidade de pior caso de algoritmos aleatórios que usam m processadores para ordenar um conjunto com n elementos, onde $m \geq n$. Com isso concluiremos que não existe algoritmo paralelo aleatório para ordenação que utilize mais de n processadores e atinja speed up ótimo.

O resultado que apresentaremos é devido a Alon e Azar [4] e deriva fundamentalmente de dois resultados anteriores além da igualdade $\mathcal{C}^{\mathcal{AI}}(\mathrm{Ord}(n,m)) = \mathcal{C}^{\mathcal{D}}(\mathrm{Ord}(n,m))$. Seja $\mathrm{c}(k,n)$ o número mínimo de comparações necessárias para ordenar n elementos em k iterações considerando todos os algoritmos determinísticos de ordenação que obrigatoriamente param após k iterações.

O primeiro resultado é devido a Alon, Azar e Vishkin, ver [5] e [6].

Lema 5.5.1 *Se $k \leq \log n$ então existe uma constante $b > 0$ tal que $\mathrm{c}(k,n) \leq kn^{1+b/k}$.*

□

O segundo resultado, devido a Alon e Azar [4], é o seguinte.

Lema 5.5.2 *O número médio de comparações que qualquer algoritmo paralelo determinístico precisa fazer para ordenar $n \geq 2$ elementos em $k \leq \log n$ iterações é maior ou igual a $ckn^{1+1/k}$ para alguma constante $0 < c < 1/2$.*

□

Omitimos as demonstrações dos dois lemas acima. Antes de apresentarmos o limite inferior de Alon e Azar, vamos demonstrar dois resultados auxiliares.

Novamente S_n denota o conjunto com todas as permutações sobre $[n]$. Dado $S \subseteq S_n$ e $\sigma_0 \in S_n$, pomos $\sigma_0(S) = \{\sigma_0\,\sigma : \sigma \in S\}$.

Lema 5.5.3 *Se $S \subseteq S_n$ e $|S| \leq n!/2$ então para todo $s \geq 1$, existem $\sigma_1, \sigma_2, \ldots, \sigma_s \in S_n$ permutações não necessariamente distintas tais que*

$$\left|\bigcap_{i=1}^{s} \sigma_i(S)\right| \leq n!/2^s.$$

Demonstração

Escolha aleatória e independentemente s permutações $\sigma_1, \sigma_2, \ldots, \sigma_s \in S_n$ não necessariamente distintas. Dada $\sigma_0 \in S_n$, temos que $\mathbf{P}(\sigma_0 \in \sigma_i(S)) = |S|/n!$, para todo $1 \leq i \leq s$. Portanto $\mathbf{P}(\sigma_0 \in \bigcap_{i=1}^{s} \sigma_i(S)) = (|S|/n!)^s \leq 1/2^s$. Temos assim que $\mathbf{E}(|\bigcap_{i=1}^{s} \sigma_i(S)|) = n!(|S|/n!)^s \leq n!/2^s$, e portanto existem s permutações em S_n tais que $|\bigcap_{i=1}^{s} \sigma_i(S)| \leq n!/2^s$.

□

Observe que precisamos do lema a seguir pois o Lema 5.5.1 só é válido para algoritmos que sempre param após um número fixo de iterações.

Lema 5.5.4 *Seja A um algoritmo determinístico que ordena vetores uniformemente sorteados de $\mathcal{V}_{[n]}$ com m processadores em tempo médio igual a T iterações. Então, para todo $s \geq 1$, existe um algoritmo determinístico que ordena vetores de $\mathcal{V}_{[n]}$ em $2T + \lceil T \rceil \leq 4T$ iterações, e necessariamente pára após $4T$ iterações, com número médio de comparações limitado superiormente por $2Tms + \lceil T \rceil n^{1+b/T}/2^s$, para $b > 0$ como no Lema 5.5.1.*

Demonstração

Suponha que queremos ordenar o conjunto $X = \{x_1, \ldots, x_n\} \subset \mathbb{Z}$. Lembre que para uma permutação $\sigma \in S_n$, denotamos por X_σ o vetor $\langle x_{\sigma(i)} \rangle_1^n$. Seja S o conjunto de todas as permutações σ de $[n]$ para as quais A não consegue ordenar X_σ em $2T$ ou menos iterações. Claramente $|S| \leq n!/2$, pois o tempo médio de A é igual a T. Pelo Lema 5.5.3 anterior, existem $\sigma_1, \sigma_2, \ldots, \sigma_n \in S_n$ tais que $|\bigcap_{i=1}^{s} \sigma_i(S)| \leq n!/2^s$.

Seja A'' o algoritmo no qual s cópias de A são executadas simultaneamente por $\lfloor 2T \rfloor$ iterações tal que a i-ésima cópia ($1 \leq i \leq s$) executa com os elementos do vetor de entrada $V = X_{\sigma_0}$ permutados de acordo com σ_i^{-1}. O algoritmo A'' ordena V corretamente nesse número de iterações a não ser que σ_0 pertença a $\bigcap_{i=1}^{s} \sigma_i(S)$. Mas isso acontece com probabilidade menor ou igual a $1/2^s$. Seja o algoritmo A' que também executa s cópias de A simultaneamente por $\lfloor 2T \rfloor$ iterações como A'', mas que se ao final desse número de iterações A' não conseguir ordenar o conjunto de entrada, então A' chama o melhor algoritmo determinístico que ordena vetores de $\mathcal{V}_{[n]}$ em $\lceil T \rceil$ iterações, o que no pior caso de acordo com o Lema 5.5.1, requer no máximo mais $\lceil T \rceil n^{1+b/T}$ comparações adicionais. Assim, temos que A' sempre

pára após $2T + \lceil T \rceil \leq 4T$ iterações com um número médio de comparações menor ou igual a $2Tms + \lceil T \rceil n^{1+b/T} 2^{-s}$, lembrando que 2^{-s} é a probabilidade de A' não conseguir ordenar o vetor de entrada em $\lfloor 2T \rfloor$ iterações.

□

Vamos agora demonstrar o resultado de Alon e Azar.

Teorema 5.5.5 *O tempo necessário para ordenar qualquer conjunto com n elementos no melhor algoritmo aleatório que usa m processadores é* $\Omega\{(\log n)/\log(1+m/n)\}$ *para* $m \geq n$.

Demonstração

Iremos provar que qualquer algoritmo determinístico A que utiliza m processadores precisa em média de $\Omega\{(\log n)/\log(1 + m/n)\}$ iterações para ordenar vetores uniformemente sorteados de $\mathcal{V}[n]$. Mas como vimos pelo Teorema 5.4.7, vale que $C^{\mathcal{A}l}(\mathrm{Ord}(n,m)) = \min_A \mathsf{E}_{d_u}(custo(A,V))$, onde d_u é a distribuição uniforme sobre vetores V de $\mathcal{V}_{[n]}$ e o mínimo é sobre os algoritmos $A \in \mathcal{A}_{\mathrm{Ord}(n,m)}$. Com isso teremos o resultado desejado.

Seja $0 < c < 1/2$, como no Lema 5.5.2. Seja $b \geq 1$ satisfazendo o Lema 5.5.1.

Tome uma constante d para a qual vale que

$$\frac{1}{16d} = 1 + \log(16b) + \log\frac{1}{2c}. \tag{1}$$

Note que o lado direito de (1) é estritamente maior que 1, pois $c < 1/2$ e $b \geq 1$.

Seja A um algoritmo determinístico que ordena vetores de $\mathcal{V}_{[n]}$, distribuídos uniformemente, com $m \geq n$ processadores em tempo médio T. Vamos provar que

$$T \geq \frac{d \log n}{\log(1 + m/n)}. \tag{2}$$

Se $T \geq d \log n$, então (2) vale para todo $m \geq n$. Vamos então supor que $T < d \log n < b \log n$. Tome $s = \lceil (b/T) \log n \rceil$. Claramente temos que $(b/T)\log n \leq s \leq 2(b/T)\log n$. Pela Proposição 5.5.4, existe uma algoritmo determinístico que ordena vetores de $\mathcal{V}[n]$ em no máximo $4T < 4d \log n < 4^{-1} \log n$ iterações com número médio de comparações menor ou igual a

$$\begin{aligned} 2Tms + \frac{\lceil T \rceil n^{1+b/T}}{2^s} &\leq \frac{2Tm(2b\log n)}{T} + \frac{\lceil T \rceil n^{1+b/T}}{2^{T^{-1} b \log n}} \\ &= 4mb\log n + \lceil T \rceil n^{1+b/T} n^{-b/T} \leq 4mb\log n + 2Tn. \end{aligned}$$

Assim pelo Lema 5.5.2, temos $4mb\log n + 2Tn \geq 4cTn^{1+(4T)^{-1}}$. Logo

$$\frac{4mb\log n}{nT} \geq 4cn^{(4T)^{-1}} - 2. \tag{3}$$

Lembre que $T < d\log n$ e portanto

$$\frac{\log n}{4T} > \frac{1}{4d}. \tag{4}$$

Observe que por (1), vale que $(4d)^{-1} > -\log c > 0$, e assim por (4) temos $\log c + (4T)^{-1}\log n > 0$, mas isto por sua vez implica $cn^{(4T)^{-1}} > 1$. Usando esse resultado em (3), temos

$$\frac{4mb\log n}{nT} \geq 4cn^{(4T)^{-1}} - 2 \geq 2cn^{1/4T}.$$

Aplicando logaritmo nos dois extremos, temos

$$\log\frac{m}{n} + \log(16b) + \log\frac{\log n}{4T} \geq \frac{\log n}{4T} + \log(2c). \tag{5}$$

De (4), vem $(4T)^{-1}\log n > (4d)^{-1} > 4$. Com isso

$$\frac{\log n}{4T} - \log\frac{\log n}{4T} \geq \frac{\log n}{8T} + \left(\frac{\log n}{8T} - \log\frac{\log n}{4T}\right) \geq \frac{\log n}{8T}. \tag{6}$$

Além disso, de (1) e (4) vem que

$$\frac{\log n}{16T} > \frac{1}{16d} > \log(16b) - \log(2c). \tag{7}$$

Juntando (5), (6) e (7) vem

$$\log\frac{m}{n} \geq \frac{\log n}{4T} - \log\frac{\log n}{4T} + \log(2c) - \log(16b) \geq \frac{\log n}{8T} + \log(2c) - \log(16b) \geq \frac{\log n}{16T}.$$

Para $m \geq n$, temos então que

$$T > \frac{\log n}{16\log(m/n)} > \frac{d\log n}{\log(1 + m/n)},$$

como queríamos demonstrar.

□

Gostaríamos de ressaltar neste último teorema não apenas a importância do limite inferior em si, mas sobretudo o uso que sua demonstração faz do Teorema 2.2.1 para ordenação. Queremos com isso mostrar a importância desse resultado, que foi provado inicialmente num contexto bastante específico para propriedades de grafos, mas que na realidade extrapola esse universo e oferece uma abordagem valiosa para o estudo de algoritmos aleatórios.

Apêndice A

Definições para grafos

O objetivo deste apêndice não é fornecer uma introdução detalhada à teoria dos grafos (para este fim sugerimos [10] e [13]; de fato várias definições aqui apresentadas são baseadas nessas duas referências), mas apenas dar algumas definições e enunciar um teorema básico (o teorema de Hall) que usamos neste trabalho. Supomos que o leitor tenha, na realidade, conhecimento do que iremos discorrer superficialmente a seguir, e que este apêndice sirva apenas como mera referência para as notações utilizadas.

Um **grafo orientado** D é um par ordenado de conjuntos finitos (V, A) onde A é um conjunto de pares ordenados de elementos distintos de V. Um **grafo simples não-orientado** G que passaremos a chamar simplesmente de **grafo** é um par ordenado de conjuntos finitos (V, A) tal que A é um conjunto de pares *não*-ordenados de elementos distintos de V. Os elementos de V são os **vértices** de G e os de A são as suas **arestas**.

Seja dado um grafo $G = (V, A)$. Vamos a seguir listar alguns conceitos relacionados com G.

O **tamanho** de G é igual a $|A(G)|$ e a **ordem** ou a **cardinalidade** de G é igual a $|V(G)|$.

Chamaremos a $V(G)$ de **conjunto dos vértices** de G, e a $A(G)$ de **conjunto das arestas** de G. Dois vértices x, $y \in V(G)$ são **vizinhos** ou **adjacentes** em G se $xy = yx = \{x, y\}$ pertence a $A(G)$, i.e., se xy é uma aresta de G. Se $a = xy$ é uma aresta de G, dizemos que a tem **pontas** x e y, ou que a **incide** em x e em y. Defimos a **vizinhança** de um vértice x em G como sendo o conjunto de vértices adjacentes a x em G, e denotamos tal conjunto por $\Gamma_G(x)$. Quando não houver dúvida omitiremos

o subscrito G, e escreveremos $\Gamma(x)$ para $\Gamma_G(x)$. O **grau** de um vértice x em G é $d_G(x) = |\Gamma_G(x)|$. O **grau máximo** de G é igual a $\Delta(G) = \max\{d_G(x) : x \in V(G)\}$. O **grau mínimo** de G é igual a $\delta(G) = \min\{d_G(x) : x \in V(G)\}$. O **grau médio** de G é $\overline{d}(G) = |V(G)|^{-1} \sum_{x \in V(G)} d_G(x) = 2|A(G)|/|V(G)|$. Se os graus médio e máximo se referem apenas a um conjunto S, com $\varnothing \neq S \subseteq V(G)$, então usaremos a notação $\overline{d}_S(G)$ e $\Delta_S(G)$, respectivamente. Mais precisamente, temos que $\Delta_S(G) = \max\{d_G(x) : x \in S\}$ e que $\overline{d}_S(G) = |S|^{-1} \sum_{x \in S} d_G(x)$.

O grafo G é **bipartido com classes** (U, W) se $V(G) = U \cup W$, $U \cap W = \varnothing$, e toda aresta de G tem uma ponta em U e a outra em W. Neste caso, (U, W) é uma **bipartição** de G, e dizemos que G é um grafo **bipartido**. Embora um grafo bipartido possa ter várias bipartições, os grafos bipartidos que consideramos têm sempre uma bipartição distinguida.

Dizemos que um grafo H é um **subgrafo** de G se $V(H) \subseteq V(G)$ e $A(H) \subseteq A(G)$. Além disso, se $V(H) = V(G)$ então dizemos que H é um **subgrafo gerador** de G e escrevemos $H \subseteq_g G$. Dado um subconjunto de vértices $X \subseteq V(G)$, este conjunto X **induz** o subgrafo $G[X]$ em G, onde o conjunto de vértices de $G[X]$ é igual a X e uma aresta a pertence a $A(G[X])$ se e somente se a pertence a $A(G)$ e a tem ambas as pontas em X. Analogamente, dado um subconjunto de arestas $S \subseteq A(G)$, escrevemos $G[S]$ para o subgrafo de G induzido, ou **gerado**, por S, i.e., o grafo cujo conjunto de arestas é igual a S e o conjunto de vértices é o conjunto das pontas das arestas em S.

Dizemos que dois grafos G e H são **isomorfos** se existe uma bijeção $\phi : V(G) \to V(H)$ tal que $xy \in A(G)$ se e somente se $\phi(x)\phi(y) \in V(H)$. Se G e H são dois grafos bipartidos com bipartições distinguidas (U_G, W_G) e (U_H, W_H) respectivamente, dizemos que G e H são **isomorfos como grafos bipartidos** se existem bijeções $\phi : U_G \to U_H$ e $\psi : W_G \to W_H$ tais que $xy \in A(G)$ se e somente se $\phi(x)\psi(y) \in A(H)$ para todo $x \in U_G$ e $y \in W_G$.

Suponha que $V(G) = \{x_1, x_2, \ldots, x_n\}$. A **matriz de adjacência** $M = M(G) = (m_{ij})$ de G é uma matriz $|V(G)| \times |V(G)|$ tal que

$$m_{ij} = \begin{cases} 1 & \text{se } x_i x_j \in A(G), \\ 0 & \text{caso contrário.} \end{cases}$$

Um subconjunto $S \subset V(G)$ de vértices é um **conjunto independente** de G se nenhum par de elementos de S for adjacente em G.

O **grafo complementar** G^c do grafo G é tal que $V(G^c) = V(G)$ e $xy \in A(G^c)$ se e somente se $xy \notin A(G)$ para todo x, $y \in V(G)$ tais que $x \neq y$. Se G é um grafo bipartido com classes (U, W), então o grafo complementar G^c de G como grafo bipartido com classes (U, W) é tal que $V(G^c) = V(G)$ e para todos os vértices $x \in U$ e $y \in W$ temos que $xy \in A(G^c)$ se e somente se $xy \notin A(G)$.

Seja $C = C^m = (V, A)$ o grafo de ordem $m + 1$ com $V' = \{x_0, x_1, \ldots, x_m\}$ e $A = \{a_i : a_i = x_i x_{i+1},\ 0 \leq i < m\}$. Dizemos que um grafo C' é um **caminho** se C' for isomorfo a C^m para algum $m \geq 0$. Denotamos o caminho $C = C^m$ acima por $(x_0, a_0, x_1, a_1, \ldots, a_{m-1}, x_m)$. O **comprimento** de C é igual a m e diremos que C vai de x_0 a x_m. Em geral um caminho será um subgrafo de um grafo considerado.

O grafo G é **conexo** se para todo par de vértices x, $y \in V(G)$, existe um caminho que vai de x a y. Uma **árvore com raiz**, ou simplesmente uma **árvore**, é um grafo conexo T com $|A(T)| = |V(T)| - 1$ e que tem um vértice especial destacado, que chamaremos de **raiz**. Muitas vezes, estaremos considerando uma árvore T e um grafo G ao mesmo tempo; para distinguir os vértices de um dos vértices do outro, chamaremos os vértices de T de **nós**. Note que para quaisquer dois nós de uma árvore T, existe um único caminho entre eles em T. Suponha que x, $y \in V(T)$ são nós de uma árvore T tais que $xy \in A(T)$, C_1 é o caminho em T que vai de x até a raiz de T, e C_2 é o caminho que vai de y até a raiz de T. Se o comprimento de C_1 for maior que o comprimento de C_2 dizemos que x é **filho** de y (ou equivalentemente que y é **pai** de x), caso contrário dizemos que y é filho de x. Dado um nó x de uma árvore T tal que x não é a raiz de T e x não tem filhos em T, dizemos que x é uma **folha**. Se um nó de uma árvore não é uma folha, então ele é um **nó interno** dessa árvore.

Um **grafo completo** ou **clique** K_r $(r \geq 1)$ é um grafo com r vértices no qual todo par de vértices distintos forma uma aresta. Freqüentemente, denotaremos um clique por K_V indicando que V é o conjunto de vértices do clique. Um **grafo completo bipartido** $K_{U,W}$ com classes (U, W) é o grafo bipartido com classes (U, W) em que todo vértice de U é adjacente a todo vértice de W.

Dado $k \geq 1$, uma **k-coloração própria dos vértices** de G é uma função $c : V(G) \to [k]$ tal que para todo par de vértices x, $y \in V(G)$ adjacentes em G, temos

que $c(x) \neq c(y)$. O **número cromático** de G é o menor k para o qual existe uma k-coloração própria dos vértices de G. Note que se G contém um grafo completo K_r, então o número cromático de G é maior ou igual a r.

Denotaremos por E_V o **grafo vazio** (i.e. o grafo com conjunto de arestas vazio) cujo conjunto de vértices é V. Denotaremos por $E_{U,W}$ o **grafo bipartido vazio com classes** (U, W).

Dado um inteiro $k \geq 0$, um **k-fator** de G é um subgrafo gerador H de G tal que todos os vértices de H têm grau k em H. Um **emparelhamento** $M \subseteq A(G)$ em um grafo G é tal que quaisquer duas arestas distintas de M não têm pontas em comum. Um emparelhamento M de um grafo G é **perfeito** se $|M| = |V(G)|/2$. Se G é um grafo bipartido com classes (U, W), então um emparelhamento M em G é perfeito se $|M| = |U| = |W|$. Observe que dado um emparelhamento M de um grafo G, esse emparelhamento induz um subgrafo H em G. Neste caso, um grafo induzido por um emparelhamento perfeito equivale a um 1-fator. A seguir vamos enunciar um resultado fundamental na teoria de emparelhamentos: o **teorema de Hall**.

Teorema *Seja G um grafo bipartido com classes (U, W). Então G contém um emparelhamento M tal que $|M| = |U|$ se e somente se*

$$|\Gamma_G(S)| \geq |S|$$

para todo $S \subseteq U$.

□

Apêndice B

Algumas notações

Vamos listar a seguir algumas das notações que utilizamos nesta dissertação. Suponha dados dois grafos G e H, um vértice x de G, e um conjunto não-vazio S de vértices de G. Ainda, sejam U, V, e W três conjuntos com $U \cap W = \varnothing$.

$A(G)$ é o conjunto de arestas do grafo G.

$d_G(x)$ é o grau do vértice x em G. Quando não houver dúvidas omitiremos o subscrito G.

$\overline{d}(G)$ é o grau médio dos vértices de G, ou seja $\overline{d}(G) = 2|A(G)|/|V(G)|$.

$\overline{d}_S(G)$ é o grau médio dos vértices de S em G, ou mais precisamente $\overline{d}_S(G) = |S|^{-1} \sum_{y \in S} d_G(y)$.

$\delta(G)$ denota o grau mínimo dos vértices de G.

$\delta_S(G)$ denota o grau mínimo dos vértices de S em G, ou seja $\delta_S(G) = \min\{d_G(y) : y \in S\}$.

$\Delta(G)$ denota o grau máximo dos vértices de G.

$\Delta_S(G)$ denota o maior grau em G dos vértices de S, ou mais precisamente, $\Delta_S(G) = \max\{d_G(y) : y \in S\}$.

E_V é o grafo vazio sobre V, ou seja, é o grafo com conjunto de vértices V tal que $A(E_V) = \varnothing$.

$E_{U,W}$ é o grafo bipartido vazio com classes U e W, ou seja, é o grafo bipartido com classes (U, W) e conjunto de arestas vazio.

G^c denota o grafo complementar de G.

$\mathcal{G}_V$ denota o conjunto de todos os grafos simples não-dirigidos com conjunto de vértices V.

$\mathcal{G}_{U,W}$ denota o conjunto de todos os grafos simples não-dirigidos bipartidos com classes (U, W)

$\Gamma_G(x)$ é conjunto com todos os vértices adjacentes a x em G. Quando não houver dúvidas, não indicamos o grafo, pondo simplesmente $\Gamma(x) = \Gamma_G(x)$.

K_V é o grafo completo com conjunto de vértices igual a V.

$K_{U,W}$ é o grafo bipartido completo com classes (U, W).

$[n]$, para $n \in \mathbb{N}$, representa o conjunto $\{1, 2, \ldots, n\}$.

$O(.)$, para funções f, g dizemos que $g = O(f)$ se existem constante $c > 0$ e n_0 tais que $g(n) \leq c\, f(n)$ para todo $n \geq n_0$.

$o(.)$, para funções f, g dizemos que $g = o(f)$ se $\lim_{n \to +\infty} g(n)/f(n) = 0$.

$\Omega(.)$, para funções f, g dizemos que $g = \Omega(f)$ se existem constantes $c > 0$ e n_0 tais que $g(n) \geq c\, f(n)$ para todo $n \geq n_0$.

$\mathrm{Ord}(n, m)$ é o problema de ordenar vetores com n inteiros distintos através de algoritmos paralelos que usam m processadores.

P^c denota a propriedade complementar de uma propriedade P de grafos, mais precisamente, um grafo pertence a P^c se e somente se o seu complementar pertence a P.

P^* denota a propriedade dual de uma propriedade P de grafos, ou seja, um grafo pertence a P^* se e somente se seu complementar *não* pertence a P.

$\mathcal{P}_{U,W}$ denota o conjunto de todas as propriedades invariantes por isomorfismos, monotônicas e não-triviais de grafos bipartidos com classes (U, W).

$\mathcal{P}_V$ representa o conjunto de todas as propriedades invariantes por isomorfismos, monotônicas e não-triviais de grafos com conjunto de vértices V.

$\Theta(.)$, para funções f, g temos que $g = \Theta(f)$ se $g = O(f)$ e $g = \Omega(f)$.

u, v e w nos primeiros quatro capítulos deste trabalho denotam as cardinalidades dos conjuntos U, V e W respectivamente.

$V(G)$ é o conjunto de vértices do grafo G.

$G[S]$ denota o subgrafo de G induzido por S, dado por $V(G[S]) = S$ e $A(G[S]) = A(G) \cap A(K_S)$.

$G \cup H$ é a união dos grafos G e H, ou seja é o grafo com conjunto de vértices igual a $V(G) \cup V(H)$ e conjunto de arestas $A(G) \cup A(H)$. Observe que *não* diferenciamos os vértices e arestas de acordo com o grafo de origem.

$G \cup X$, onde $X \subseteq A(K_{V(G)})$, é o grafo com conjunto de vértices $V(G)$ e conjunto de arestas $A(G) \cup X$.

$G + H$ é a união disjunta de G e H, i.e., é o grafo com conjunto de vértices igual à união disjunta de $V(G)$ e $V(H)$, e conjunto de arestas igual à união disjunta de $A(G)$ e $A(H)$.

$G \times H$ é a união disjunta de G e H com todas as arestas entre os dois grafos, i.e., temos que $G \times H = (G + H) \cup K_{V(G),V(H)}$.

$G \setminus H$ é o grafo com conjunto de vértices $V(G)$ e conjunto de arestas $A(G) \setminus A(H)$.

$G \setminus S$ denota o grafo $G[V(G) \setminus S]$, e se $S = \{x\}$, escrevemos $G \setminus x$ para $G \setminus S$.

$G \setminus X$, onde $X \subseteq A(K_{V(G)})$, é o grafo com conjunto de vértices $V(G)$ e conjunto de arestas $A(G) \setminus X$.

$G \subseteq H$ denota que G é um subgrafo de H, ou seja $V(G) \subseteq V(H)$ e $A(G) \subseteq A(H)$.

$G \subseteq_g H$ denota que G é um subgrafo gerador de H, ou seja $V(G) = V(H)$ e $A(G) \subseteq A(H)$.

G^c é o grafo complementar de G, ou seja, é o grafo tal que $V(G^c) = V(G)$ e $A(G^c) = A(K_{V(G)}) \setminus A(G)$. Se G é um grafo bipartido com classes (U, W), entao G^c tem conjunto de vértices igual a $V(G)$ e conjunto de arestas igual a $A(K_{U,W}) \setminus A(G)$.

Bibliografia

[1] Aho, V.A., Hopcroft, J.E., e Ullman, J.D., *The Design and Analysis of Computer Algorithms*, Addison–Wesley, Londres 1976.

[2] Ajtai, M., Komlós, J., e Szemerédi, E., An $O(n \log n)$ sorting network, em *Proc. 15th Symposium on Theory of Computing* (1983), pp. 1–9.

[3] Ajtai, M., Komlós, J., e Szemerédi, E., Sorting in $c \log n$ parallel steps, *Combinatorica* **3** (1983), 1–19.

[4] Alon, N., e Azar, Y., The average complexity of deterministic and randomized parallel comparison sorting algorithms, em *Proc. 28th Annual Symposium on the Foundations of Computer Science* (1987), pp. 489–498.

[5] Alon, N., Azar, Y., e Vishkin, U., Tight complexity bounds for parallel comparison sorting, em *Proc. 27th Annual Symposium on the Foundations of Computer Science* (1986), pp. 502–510.

[6] Azar, Y., e Vishkin, U., Tight comparison bounds on the complexity of parallel sorting, *SIAM Journal of Computing* **16** (1987), 458–464.

[7] Bárány, I., e Füredi, Z., Computing the volume is difficult, em *Proc. 18th Annual ACM Symposium on Theory of Computing* (1986), pp. 442–447.

[8] Best, M.R., van Emde Boas, P., e Lenstra Jr, H.W., *A sharpened version of the Aanderaa–Rosenberg conjecture*, relatório técnico ZW 30/74 do Mathematische Centrum, Amsterdã 1974.

[9] Bollobás, B., *Extremal Graph Theory*, Academic Press, Londres 1978.

[10] Bollobás, B., *Graph Theory – An Introductory Course*, Springer–Verlag, Nova Iorque 1979.

[11] Bollobás, B., *Random Graphs*, Academic Press, Londres 1985.

[12] Bollobás, B., Complete subgraphs are elusive, *Journal of Combinatorial Theory B* **21** (1976), 1-7.

[13] Bondy, J.A., e Murty, U.S.R., *Graph Theory with Applications*, The MacMillan Press Ltd, Londres 1977.

[14] Catlin, P.A., Subgraphs of graphs I, *Discrete Mathematics* **10** (1974), 225–233.

[15] Dyer, M.E., e Frieze, A.M., Computing the volume of convex bodies: a case where randomness provably helps, em *Probabilistic Combinatorics and its Applications* (Bollobás, B., ed.), Proceedings of Symposia in Applied Mathematics 44, American Mathematical Society, Providence 1991, pp. 123–169

[16] Dyer, M.E., Frieze, A.M., e Kannan, R., A random polynomial time algorithm for approximating the volume of convex bodies (extended abstract), em *Proc. 21st Annual ACM Symposium on Theory of Computing* (1989), pp. 375–381.

[17] Elekes, G., A geometric inequality and the complexity of computing the volume, *Discrete and Computational Geometry* 1 (1986), 289–292.

[18] Floyd, R.W., e Rivest, R.L., Expected time bounds for selection, *Communications of the ACM* **18** (1975), 165–172.

[19] Graham, R.L., Knuth, D.E., e Patashnik, O., *Concrete Mathematics*, Addison–Wesley, Reading 1988.

[20] Hajnal, P., *An $\Omega(n^{4/3})$ lower bound on the randomized complexity of graph properties*, relatório técnico 88–004 da University of Chicago Chicago 1988.

[21] Hajnal, P., *Complexity of Graph Problems*, tese de PhD, University of Chicago, Chicago 1988, relatório técnico 88–19 da University of Chicago Chicago 1988.

[22] Hajnal, P., An $\Omega(n^{4/3})$ lower bound on the randomized complexity of graph properties, *Combinatorica* **11** (1991), 131–143.

[23] Kahn, J., Saks, M., e Sturtevant, D., A topological approach to evasiveness, *Combinatorica* 4 (1984), 297–306.

[24] Karp, R., e Ramachandran, V., *A Survey of Parallel Algorithms for Shared-Memory Machines*, relatório técnico UCB/CSD 88/408 da University of California at Berkeley, Berkeley 1988.

[25] King, V., Lower bounds on the complexity of graph properties, em *Proc. 20th Annual ACM Symposium on Theory of Computing* (1988), pp. 468–476.

[26] King, V., A lower bound for the recognition of digraph properties, *Combinatorica* **10** (1990), 53–59.

[27] King, V., An $\Omega(n^{5/4})$ lower bound on the randomized complexity of graph properties, *Combinatorica* **11** (1991), 23–32.

[28] Kleitman, D.J., e Kwiatkowski, K.J., Further results on the Aanderaa–Rosenberg conjecture, *Journal of Combinatorial Theory* **28** (1980), 85–95.

[29] Knuth, D.E., *The Art of Computer Programming—Vol. 3, Sorting and Searching*, Addison–Wesley, Reading 1973.

[30] Lipton, R.J., e Snyder, L., On the Aanderaa–Rosenberg conjecture, *SIGACT News* **6** (1974), 30–31.

[31] McDiarmid, C., On the method of bounded differences, em *Surveys in Combinatorics, 1989* (Siemons, J. ed.) London Mathematical Society Lecture Notes Series 141, Cambridge University Press, Cambridge 1989, pp. 148–188

[32] von Neumann, J., Zur Theorie der Gesellschaftsspiele, *Mathematische Annalen* **100** (1928), 295–320.

[33] Reischuk, R., A fast probabilistic parallel sorting algorithm, em *Proc. 22nd Annual Symposium on the Foundations of Computer Science* (1981), pp. 212–219.

[34] Rivest, R.L., e Vuillemin, S., A generalization and proof of the Aanderaa–Rosenberg conjecture, em *Proc. 7th SIGACT Conf.*, Albuquerque 1975.

[35] Rivest, R.L., e Vuillemin, S., On recognizing graph properties from adjacency matrices, *Theor. Comp. Sci.* **3** (1978), 371–384.

[36] Rosenberg, A.L., On the time required to recognize properties of graphs: a problem, *SIGACT News* **5** (1973), 15–16.

[37] Saks, M., e Wigderson, A., Probabilistic Boolean decision trees and the complexity of evaluating game trees, em *Proc. 27th Annual Symposium on Foundations of Computer Science* (1986), pp. 29–38.

[38] Sauer, N., e Spencer, J.H., Edge-disjoint replacement of graphs, *Journal of Combinatorial Theory B* **25** (1978), 295–302.

[39] Valiant, L., Parallelism in comparison problems, *SIAM Journal of Computing* **4** (1975), 348–355.

[40] Vazirani, U., Rapidly mixing Markov chains, em *Probabilistic Combinatorics and its Applications* (Bollobás, B., ed.), Proceedings of Symposia in Applied Mathematics 44, American Mathematical Society, Providence 1991, pp. 99–121.

[41] Yao, A.C.C., Probabilistic computations: toward a unified measure of complexity (extended abstract), em *Proc. 18th Annual Symposium on the Foundations of Computer Science* (1977), pp. 222–227.

[42] Yao, A.C.C., Lower bounds to randomized algorithms for graph properties (extended abstract), em *Proc. 28th Annual Symposium on the Foundations of Computer Science* (1987), pp. 393–400.

[43] Yao, A.C.C., Monotone bipartite graph properties are evasive, *SIAM Journal of Computing* **17** (1988), 517–520.

[44] Yao, A.C.C., Lower bounds to randomized algorithms for graph properties, *Journal of Computer and System Sciences* **42**, (1991), 267–287.

[45] Yap, H.P., Computational complexity of graph properties, em *Graph Theory, Singapore 1983* (Koh, K.M., Yap, H.P., eds), Springer–Verlag Lecture Note Series in Mathematics 1073, Berlin 1984, pp. 35–54.

[46] Yap, H.P., *Some Topics in Graph Theory*, London Mathematical Society Lecture Note Series 108, Cambridge University Press, Cambridge 1986.