SISTEMA PARA ANÁLISE AUTOMÁTICA DA COMPLEXIDADE DE ALGORITMOS NÃO RECURSIVOS NO PIOR CASO

Marco Antonio de Castro Barbosa
UNICRUZ - Universidade de Cruz Alta, Dept. Informática,
Cruz Alta, Brasil, 98050-010
marco@unicruz.edu.br

Abstract

The Analysis of Algorithms, or the calculation of the Complexity of Algorithms is an indispensable task for the construction of good and efficient algorithms. Moreover the possibility of to esteem the time of an algorithm execution, the calculation of its complexity serves to aid in the choice one among some algorithms that solve the same problem. Based in the importance of this area and with the main purpose to support its education in courses of Computation and similars, was developed a tool to analyze automatically the complexity of algorithms. This tools serves to experienced designers, who dominate the area, and to beginners students in this area

Keywords: Complexity of Algorithms, Automatic Analysis, Worst Case.

Resumo

A Análise de Algoritmos, ou o cálculo da Complexidade de Algoritmos é uma tarefa indispensável para a construção de algoritmos bons e eficientes. Além da possibilidade de estimar o tempo de execução de um algoritmo, o cálculo de sua complexidade serve para auxiliar na escolha de um, dentre vários algoritmos que resolvam o mesmo problema. Com base na importância desta área e com a finalidade maior de apoiar seu ensino em cursos de Computação ou afins, foi desenvolvida uma ferramenta para analisar automaticamente a complexidade de algoritmos, servindo tanto para projetistas experientes, que dominem a área, quanto para estudantes iniciantes na área.

Palavras-clave: Complexidade de Algoritmos, Análise Automática de Algoritmos, Pior Caso

1. Introdução

A união da metodologia para o cálculo da complexidade de algoritmos proposta por [TOS90], associada com a idéia da análise automática de programas [WEG75][FLA88][MET88][SIL98], serviu de motivação para o desenvolvimento do protótipo de sistema ANAC – Analisador de Complexidade, que se constitui em uma ferramenta para o apoio ao ensino de Complexidade de Algoritmos. Os objetivos deste sistema são:

? Servir como ferramenta de apoio ao ensino de complexidade de algoritmos, constituindo-se num ambiente de aplicação prática ao embasamento teórico adquirido no ensino de complexidade algorítmica;
? Fornecer um ambiente que guie o usuário nos passos necessários ao desenvolvimento do cálculo da complexidade de algoritmos;
? Calcular a complexidade de algoritmos no Pior Caso;

? Estimular o cálculo da complexidade de algoritmos, como um meio de tornar programas mais eficientes.

Este artigo está dividido em 5 seções. A seção 2 apresenta alguns conceitos de Complexidade de Algoritmos. Na seção 3 é apresentada a ferramenta ANAC. Na seção 4 é apresentado um exemplo de aplicação da ferramenta. Na seção 5 são apresentadas as conclusões.

2. Complexidade de Algoritmos

Por que analisar a complexidade de um algoritmo? Segundo [SED96], existem muitas respostas para esta questão, dependendo do contexto, algumas das possíveis respostas podem ser: a finalidade da utilização do algoritmo; a importância do algoritmo com relação a outros algoritmos que solucionem o mesmo problema, para que se possa decidir por qual deles optar na implementação da solução do problema; a dificuldade da precisão e análise da resposta requerida.

A mais simples razão para se analisar um algoritmo é para poder apurar suas características e avaliar a viabilidade de sua utilização prática, ou para poder comparar o algoritmo com outros algoritmos desenvolvidos com o mesmo objetivo do algoritmo em questão. É desejável saber quanto tempo uma implementação de um algoritmo particular irá consumir durante sua execução ou quanto espaço irá requerer para poder certificar-se da eficiência do algoritmo.

Complexidade Computacional é a área da Ciência da Computação que elucida as razões do porquê alguns problemas são tão difíceis de serem resolvidos por computadores. A complexidade de um algoritmo está relacionada com o esforço computacional necessário para a sua execução.

Algoritmos podem ser avaliados por uma variedade de critérios de medidas. Os critérios mais freqüentemente utilizados são a taxa de crescimento do tempo ou espaço requerido para resolver grandes instâncias de um problema. Tais medidas são denominadas, respectivamente, de complexidade de tempo e complexidade de espaço.

Em relação à complexidade de tempo, pode-se ainda ter as complexidade no Pior Caso ou no Caso Médio. A complexidade no pior caso é geralmente a medida mais empregada na prática. Fixado um tamanho de entrada, a análise no pior caso é feita em relação ao número máximo de operações fundamentais necessárias para a resolução de qualquer problema do tamanho fixado. Seu valor pode ser considerado como um limite de complexidade que não será ultrapassado, sendo portanto, uma garantia de qualidade mínima do algoritmo. Se a complexidade é tomada como uma complexidade "média" sobre todas as entradas do tamanho dado, então a complexidade é denominada de complexidade esperada ou complexidade média. A complexidade média de um algoritmo é usualmente mais difícil de se obter do que a complexidade no pior caso. A análise no caso médio é baseada nas distribuições probabilísticas dos dados de entrada do algoritmo. A dificuldade no cálculo do caso médio está justamente nas distribuições probabilísticas dos dados de entrada, que nem sempre são conhecidas. Apesar disto, existem bons exemplos de aplicação deste cálculo, porém utilizando sempre distribuição uniforme. Um exemplo característico é o algoritmo de ordenação Quicksort, que apresenta desempenho médio O(n log n) [HOR78], enquanto o seu desempenho no pior caso é O(n2), mas sabe-se que na prática, o algoritmo Quicksort raramente terá este desempenho pessimista.

A análise de algoritmos é uma atividade que contribui para o entendimento fundamental da Ciência da Computação. Segundo [AHO82], a complexidade é o coração da computação.

Nos últimos anos a pesquisa na área de algoritmos progrediu bastante. Esse processo foi alcançado com o desenvolvimento de algoritmos mais rápidos, tais como o da Transformada Rápida de Fourier – FFT [SED96]. Mas, também houve, a importante descoberta de certos problemas para os quais todos algoritmos são ineficientes. Estes resultados tem despertado considerável interesse no estudo de algoritmos, e a área de projeto e análise de algoritmos vem tornando-se um campo de intenso interesse.

O processo de análise da complexidade de algoritmos é uma tarefa que exige conhecimento e perícia do analista. Isto tem motivado, ao longo dos anos, pesquisas com a finalidade de criar métodos e construir ferramentas que tornem o processo de análise automático.

A análise da complexidade de um algoritmo é usualmente tratada de maneira muito particular, ou seja, é uma atividade muito dependente da classe dos algoritmos a serem analisados. O cálculo depende essencialmente da função tamanho da entrada e das operações fundamentais. No entanto, alguns aspectos do cálculo da complexidade não dependem apenas do tipo de problema que o algoritmo resolve, dependem também das estruturas que compõem o algoritmo, podendo ser assim generalizados. Esta idéia motivou o desenvolvimento de uma metodologia de cálculo de complexidade para estruturas algorítmicas para o pior caso [TOS90].

3. A Ferramenta ANAC

No ensino de Projeto e Análise de Algoritmos, a utilização de uma ferramenta de análise automática de complexidade é motivadora do desenvolvimento de algoritmos eficientes e uso de técnicas que visam melhorar o desempenho do algoritmo.

Algumas pesquisas relacionadas ao tema da mecanização do processo de análise algorítmica e de metodologias para a automatização da análise de algoritmos foram motivadoras para o desenvolvimento da ferramenta ANAC – Analisador de Complexidade.

A base para o desenvolvimento do sistema ANAC é a metodologia de cálculo de complexidade para estruturas algorítmicas proposta em [TOS90]. Esta metodologia engloba as principais estruturas que podem estar presentes em um algoritmo, por exemplo: atribuição, seqüência, condicional, iteração e iteração condicional. Esta metodologia é direcionada à análise no pior caso.

O sistema ACME – Analisador de Complexidade Média [SIL98], foi desenvolvido com o propósito de adaptar a metodologia de [ROS97] para uma ferramenta que automatizasse o processo de cálculo de complexidade de algoritmos no caso médio e também serviu de inspiração para o desenvolvimento do sistema ANAC.

O sistema ANAC é uma ferramenta de apoio ao cálculo de Complexidade de Algoritmos. Os principais objetivos deste sistema são:

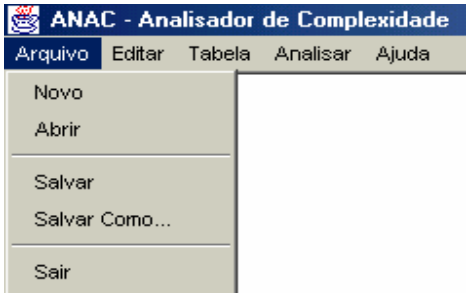
- Dar suporte ao desenvolvimento de algoritmos eficientes;
- Servir como ferramenta de apoio ao ensino de complexidade de algoritmos, constituindo-se num ambiente de aplicação prática ao embasamento teórico adquirido no ensino de complexidade algorítmica;
- Calcular a complexidade de algoritmos no Pior Caso; e,
- Estimular o cálculo da complexidade de algoritmos.

O sistema analisa algoritmos escritos na linguagem Pascal-like englobando as principais estruturas presentes em linguagens imperativas: atribuição, seqüência, condicional, iteração e iteração condicional, calculando de forma semi-automática a equação de complexidade e resolvendo-a sempre que forem fornecidos os dados necessários para tal operação. O algoritmo pode conter chamadas a procedimentos não especificados, cuja complexidade deve ser fornecida pelo usuário.

O sistema foi implementado na linguagem de programação Java, pela portabilidade desta linguagem e pelo propósito de disponibilizar o uso do sistema ANAC através da Internet.

4. Exemplo de Aplicação da Ferramenta ANAC

O sistema apresenta-se ao usuário como uma ferramenta para edição de textos. O ANAC possui uma janela com menu de opções **Arquivo**, **Editar**, **Tabela**, **Analisar** e **Ajuda**.

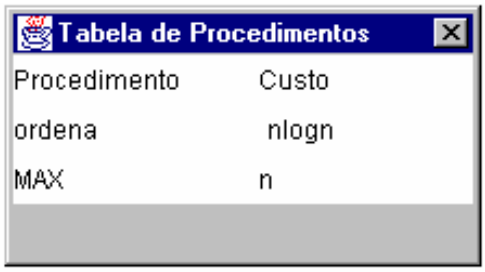


O processo

Fig. 1 – O Menu do ANAC

de análise tem inicio quando o usuário aciona o sub-item **Iniciar** no menu **Analisar**. Estas opções estão representadas na Fig. 1.

Uma das características principais do sistema ANAC é a possibilidade de o usuário definir funções ou utilizar funções pré-estabelecidas. A opção **Utilizar Tabela**, irá carregar e disponibilizar a tabela para uso do sistema. Nesta tabela estarão definidos os procedimentos usados pelo usuário e o custo relativo a cada procedimento. Durante o processo de análise do algoritmo, ao encontrar um procedimento externo, o sistema irá buscar na tabela o custo daquele procedimento (complexidade) desta forma terá subsídio para realizar o cálculo da



complexidade

Fig. 2 – Tabela de Procedimentos

do algoritmo. Um exemplo pode ser observado na Fig. 2, onde pode-se observar os procedimentos ordena, de custo nlogn e MAX de custo n.

Para definir um novo procedimento deve-se acionar a opção **Inserir Item**. O usuário irá definir o nome do procedimento e o custo deste procedimento.

O usuário tem a possibilidade de alterar ou excluir um item da tabela, para tanto, deve acionar as respectivas opções **Alterar Item** ou **Excluir Item**. A janela da Fig.3 representa a

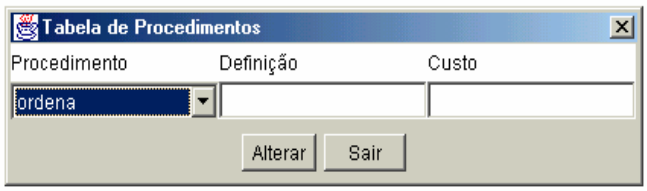
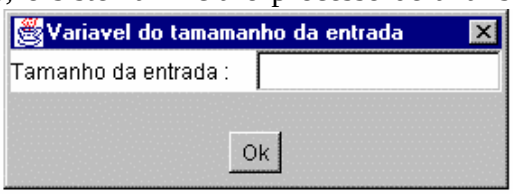


Fig. 3 – Opção Alterar Tabela

opção para alteração, a janela para exclusão é semelhante a esta. Na opção **Alterar**, o usuário poderá modificar o nome do procedimento, o seu custo, ou ambos.

Após o usuário ter definido a utilização, ou não, da tabela e havendo um algoritmo carregado o processo de análise pode ser iniciado. Para isto, deve ser selecionada a opção **Analisar/Iniciar**.

Caso o processo de análise tenha início e não exista nenhum algoritmo carregado o sistema emite uma mensagem ao usuário informando que não existe algoritmo para ser analisado. Do contrário, o sistema inicia o processo de análise e solicita ao usuário que

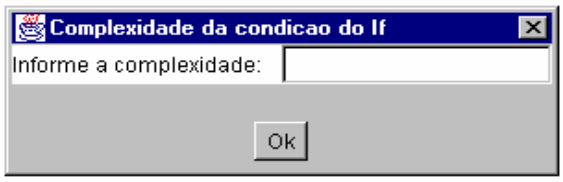


este informe

Fig. 4 – Variável que Identifica o Tamanho da Entrada

a variável que identifica o tamanho da entrada, Fig. 4.

Para poder calcular a complexidade de estruturas condicionais e iterações o sistema irá interagir com o usuário a fim de que este forneça as informações necessárias para que estas estruturas possam ser analisadas. Quando o analisador léxico-sintático encontrar o comando *if* o sistema irá solicitar que o usuário forneça a complexidade da avaliação da sua condição. Por exemplo: *if* $a < b$ *then* , o sistema necessita saber qual a complexidade da avaliação de $a < b$. Para obter esta informação o sistema apresenta ao usuário a janela da Fig. 5. Caso o usuário

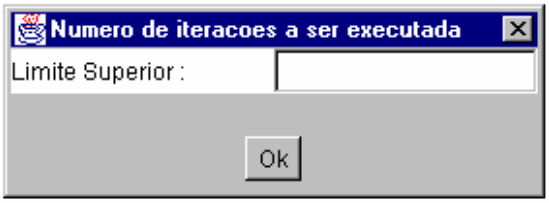


não

Fig. 5 – Complexidade da Estrutura Condicional.

informar a complexidade da condição e simplesmente acionar o botão **ok,** o sistema assumirá complexidade 1 (constante) para a avaliação da condição.

Da mesma forma que atua com o comando *if* o sistema irá agir quando encontrar um comando *while*. Neste caso o usuário terá que informar ao sistema, além da complexidade da avaliação da condição, um limite superior, ou seja, o valor que referente ao número de



iterações

Fig. 6 – Iteração Condicional

que serão executadas, Fig. 6.

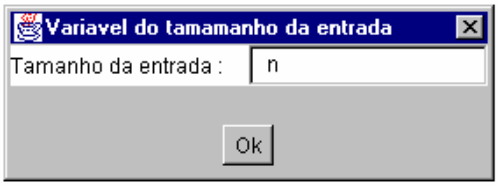
4.1 Exemplo de Aplicação do Protótipo ANAC

Na análise deste algoritmo serão ilustrados todos os passos relativos à análise do algoritmo, incluindo os passos de interação com o usuário.

O presente algoritmo tem por objetivo determinar o máximo e o mínimo de uma tabela, armazenada como vetor Tab, comparando cada elemento com candidatos.

```
program MaxMin;
begin
        Max := Tab[1];
        min := Tab[1];
        for i := 1 to n-1 do
                if Tab[i] > Max then
                        max := Tab[1];
                endif;
                if Tab[i] < min then
                        max := Tab[1];
                endif;
        endfor;
end.
```

Para iniciar a execução do processo de análise do algoritmo acima, o usuário deve acionar no menu Analisar, o item Iniciar. A janela da Fig. 7 irá solicitar que o usuário defina a



variável

Fig. 7 – Identificação do Tamanho da Entrada

que identifica o tamanho da entrada.

Neste exemplo o usuário definiu como *n* a variável do tamanho da entrada. De posse dessa informação o sistema prossegue sua análise:

x1 = C (Max := Tab[1]) + x2 = 1 + x2
x2 = C (min := Tab[1]) + x3 = 1 + x3
x3 = C (for i := 1 to n-1 do x4) + x5 =
[SUM(i = 1 , n-1) x4] + x5
x4 = C (Tab[i]>Max) + x6 + x7

Neste momento da análise o sistema identificou uma estrutura condicional *if*. Para calcular a complexidade desta estrutura o sistema solicitará ao usuário que indique a complexidade da avaliação da condição (*Tab[i] > Max*). Neste exemplo o usuário definiu como complexidade constante (=1). Caso o usuário não digite nenhuma informação o sistema assume a complexidade da avaliação como constante. Fig. 8.

Fig. 8 – Complexidade da Avaliação da Condição

x6 = C (max := Tab[1]) + x8 = 1 + x8
x8 = C (endif) = 0
x6 = 1 + 0 = 1
x7 = C (Tab[i]<min) + x9 + x10

Neste ponto novamente a janela da Fig 8 será apresentada ao usuário para que ele defina a complexidade da avaliação da condição *Tab[i] < min*, encontrada nesta nova estrutura condicional *if*.

Após esta definição o processo de análise transcorre até o final sem novas interações com o usuário, uma vez que não encontrou novas estruturas que necessitassem de informações externas para a efetivação do processo de cálculo da complexidade do algoritmo.

x9 = C (max := Tab[1]) + x11 = 1 + x11
x11 = C (endif) = 0
x9 = 1 + 0 = 1
x10 = C (endfor) = 0
x7 = 1 + 1 + 0 = 1
x4 = 1 + 1 + 1 = 1
x5 = C (end) = 0
x3 = [SUM (i = 1 , n-1) 1] + 0 = n + 0 = n
x2 = 1 + n = n
x1 = 1 + n = n
O (n)

5. Conclusão

A apresentação em [TOS90] da análise da complexidade de algoritmos como uma tarefa sistemática a partir de estruturas algorítmicas, e não como é classicamente apresentada, como uma tarefa particular para cada algoritmo, tornou possível automatizar o cálculo de complexidade de algoritmos.

O sistema ANAC, analisa algoritmos não recursivos escritos numa linguagem Pascal-like, o que o torna mais utilizável, por ser o Pascal uma linguagem amplamente difundida no meio acadêmico, ao contrário de outros sistemas que trabalham com a linguagem Lisp ou outras linguagens cuja utilização é mais restrita.

O sistema ANAC é uma ferramenta de apoio ao ensino de complexidade de algoritmos não recursivos, constituindo-se num ambiente interativo entre o usuário e o sistema durante o processo de análise. É um sistema simples de ser usado, que não exige conhecimento de um sistema operacional específico, software matemático ou linguagens pouco.

É um sistema interativo que vai guiando o usuário durante o processo de análise, isto faz com que o mesmo vá adquirindo perícia neste processo e venha a torná-lo um hábito, durante o processo de desenvolvimento de um algoritmo.

O resultado é simplificado, dado em ordem de complexidade diferente de outros sistemas que podem gerar extensas e complicadas equações, que podem ser desestimulantes para quem não esteja habituado com o processo de cálculo de complexidade ou para quem está se iniciando no estudo da Teoria da Complexidade.

O sistema ANAC pretender tornar a análise de algoritmos uma prática freqüente no projeto e desenvolvimento de algoritmos eficientes. A utilização do sistema ou método deverá criar no projetista (usuário) uma cultura de desenvolvimento em que esta prática acabe se tornando uma tarefa menos árdua do que se pode ver atualmente em equipes ou profissionais que desenvolvem algoritmos e buscam eficiência e, no entanto, fogem à prática da análise da complexidade de algoritmos.

Referências

[AHO82] AHO, A.; HOPCROFT, J.; ULLMAN, J. Data Structures and Algorithms. Addison-Wesley. 1982.

[FLA88] FLAJOLET, Philippe; SALVY, Bruno & ZIMMERMANN, Paul. ? ? ? : An Assistante Algorithms Analyser. In Proceedings AAECC'6, Lecture Notes in Computer Science 357, pg. 201-212, 1988. Also available as INRIA Reserch Report 876, 1988.

[HOR78] HOROWITZ, Ellis & SAHNI, Sartaj. Fundamentals of Computer Algorithms. Computer Science Press Inc. 1978

[MET88] METAYER, D. Le. Ace: An automatic complexity evaluator. ACM Transactions on Programming Languages and Systems, 10(2):248-266,1988.

[ROS97] ROSA, Débora Schuch da. Complexidade Média Algorítmica: uma Metodologia para o seu cálculo. - Dissertação de Mestrado - CPGCC / UFRGS, 1997.

[SED96] SEDGEWICK, Robert; FLAJOLET, Philippe. An Introduction to the Analysis of Algorithms. Addison Wesley. 1996.

[SIL98] SILVEIRA; Carlos Morelli D. Analisador de Complexidade Média Baseado nas Estruturas Algorítmicas. UFPEL – Pelotas, 1998.

[TOS90] TOSCANI, L. V. & VELOSO, Paulo A. S. Uma metodologia para cálculo da complexidade de algoritmos. Simpósio Brasileiro de Engenharia de Software, 4. Águas de São Pedro/SP, out. 24-26. Anais, SBC 1990.

[WEG75] WEGBREIT, B. Mechanical program analysis. Communications of the ACM, 18(9):528-539, 1975.