Relatório de Sistemas Operacionais I: N Rainhas

Abilio Gambim Parada, Pargles Dall’Oglio

# 1. Introdução

Este trabalho tem como objetivo apresentar uma breve descrição do desenvolvimento de um programa capaz de distribuir N rainhas em um tabuleiro de forma que não haja conflito entre elas. Este relatório também relata as variáveis contidas na seção crítica assim como os mecanismos de sincronização utilizados como de forma a resolver o problema das N Rainhas de forma paralela. Também são apresentados os resultados médios de tempo em execução para N rainhas e seus desvios padrão. Ao final também é realizada uma comparação entre as avaliações com distintos computadores utilizando quantidades de processadores diferentes e analisado Speed Up e Eficiência.

# 2. Classes e Métodos Implementados

O programa foi desenvolvido seguindo o paradigma de programação orientado a objeto, sendo desenvolvido na linguagem Java. Abaixo serão apresentadas as classes e os métodos de maior importância implementados. O programa contém um total de seis classes, sendo cinco desenvolvidas e uma obtida da internet como forma de comparação.

## 2.1. Main

Essa classe apenas cria um JFrame que será responsável pela inicialização das demais classes.

## 2.2. NRainhasModel

Classe responsável pelo calculo e geração de cada quadradinho que o tabuleiro deve conter.

## 2.3. NRainhasUI

Contém todos os componentes necessários para a geração da interface gráfica. Classe que estende a classe JPanel, possui todas as declarações de botões, laibels, caixas de texto, caixas de seleção entre outros. Contém também todos os eventos que serão gerados a partir da interação com o usuário, assim como a apresentação e salvamento dos resultados.

## 2.4. Queens Recursivo

Classe obtida em [1], obtém de forma recursiva o resultado para o problema das N Rainhas. Essa classe foi utilizada como forma de comparação para os algoritmos sequencial e paralelo desenvolvidos, conforme será apresentado na seção X. A saída de dados foi alterada para se comportar como as demais.

## 2.5. NQueens

Trata-se de umas das classes com maior importância. Obtém a quantidade de processadores disponíveis assim como a quantidade de rainhas a serem processadas. Essa classe também contém os métodos responsáveis por selecionar o algoritmo a ser executado assim como fazer a divisão de tarefas entre as threads e criar instâncias para cada uma delas. Os métodos mais importantes que compõem essa classe são descritos nas subseções abaixo.

## 2.5.1. public void executa() throws Exception

Método que contém basicamente uma seleção entre os algoritmos disponíveis e também é responsável pela inicializaão das variáveis necessárias para a execução de cada um dos algoritmos.

## 2.5.2. public void escalonarTarefas()

Método chamado apenas quando a execução paralela é solicitada. Esse método divide a quantidade de rainhas a serem calculadas por cada um dos processadores disponíveis e salva essa informação em um vetor que pode ser utilizado no método *private void iniciarTarefas().*

## 2.5.3. private void iniciarTarefas() throws IOException

O método acima é utilizado para criar as threads para execução. O código abaixo ilustra seu funcionamento:

paraleloQueens rainha;

for(int i = 0 ; i< processadores;i++)

{

superior += escalonador[i];

queen = new paraleloQueens(inferior,superior,rainhas,nomeArquivo);

queen.start();

inferior= superior;

}

Como pode-se observar no código acima, o laço vai criando e iniciando threads para calcularem N/nucleos rainhas, de um limite inferior até um limite superior. Esse mesmo laço é utilizado para a execução do algoritmo sequencias, porém, o cálculo da posição de todas as rainhas é atribuído a somente uma thread, ou seja, o limite inferior é a posição 0 e o limite superior é a posição N, não há intervalos para serem calculados por outras threads.

## 2.6. paraleloQueens

É a classe responsável pelo paralelismo e por estender a classe Threads. Os principais métodos que a compõem são descritos nas subseções abaixo.

## 2.6.1. public void run()

Sobrescreve o método run da classe Thread para a utilização do paralelismo. Contém o laço principal do algoritmo. Sua execução é feita através de um vetor de N posições (cada posição indica a localização de uma rainha) indicando através de números, de 0 até N, indicando a posição de uma rainha na linha. Esse laço vai inserindo rainhas em posições subsequentes e a cada inserção o método *boolean unsafe(int colunaAtual)* é chamado para verificar se existe algum conflito entre as rainhas já inseridas. Casso haja conflito a posição atual da rinha é deslocada e assim sucessivamente até encontrar, ou não, alguma solução. Trata-se de uma algoritmo de força bruta que não utiliza nenhuma técnica de espelhamento ou heurística justamente para melhor avaliar a diferença entre a utilização e não utilização de paralelismo. O algoritmo divide entre os núcleos disponíveis as colunas a serem avaliadas.

## 2.6.2. boolean unsafe(int colunaAtual)

Método utilizado durante a execução do método principal *public void run()*. O método unsafe é responsável por avaliar se existe algum conflito entre as rainhas inseridas até o momento, através de um laço que compara se houve conflito na linha ou na diagonal entre todas as rainhas inseridas até o momento.

## 2.6.3. public static synchronized void putboard(String solution) throws IOException

O método *putboard(String solution)*  é responsável por salvar no arquivo os resultados obtidos. Esse método contém a seção crítica que é composta pelos seguinte trecho de código:

public static synchronized void putboard(String solution) throws IOException

{

arquivo = new FileWriter(nomeArq, true);

BufferedWriter bw = new BufferedWriter(arquivo); <<< precisa do buffer agora?

bw.write(solution);

bw.newLine();

bw.close();

}

Como o método acima está inserido na classe paraleloQueens e são geradas N instancias dessa classe foi necessário criar um método estático e sincronizado de forma a garantir que apenas uma thread por vez obtenha o direito de salvar seu resultado no arquivo.

# 4. Avaliação de Desempenho

Para a avaliação de desempenho entre os algoritmos foram utilizados três computadores com configurações difentes. Os resutados completos que foram utilizados para o cálculo da média e desvio padrão estão disponíveis no Apêndice A. A Tabela 1 apresenta os resultados médios

**Tabela 1. Tempo de execução médio de todos os algoritmos para o computador X**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Sequencial | | Paralelo | | Recursivo | |
| Rainhas | Média | DP | Média | DP | Média | DP |
| 1 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |
| 13 |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |
| 16 |  |  |  |  |  |  |

Os resultados são utilizados nas fo´rmulas (1) e (2).



# 5. Conclusão

Este trabalho apresentou, um relatório referente a um Proxy desenvolvido pelos autores bem como uma descrição dos resultados obtidos.

# 6. Referências

[1] Robert Sedgewick and Kevin Wayne: **“Introduction to Programming in Java: An Interdisciplinary Approach”**, Disponível em: http://introcs.cs.princeton.edu/java/23recursion/Queens.java.html. Acesso em maio de 2012.

[2] Timothy J. Rolfe., **“Queens on a Chessboard: Making the Best of a Bad Situation”**, Disponível em: http://penguin.ewu.edu/~trolfe/SCCS-95/SCCS-95. Acesso em maio de 2012.

[3] Abilio G. Parada e Pargles Dall’Oglio, **“NRainhas”**, Disponível em: https://github.com/pargles/NRainhas. Acesso em maio de 2012.

[4] Bill Butler. **“The N-Queens Problem”**, Disponível em: http://www.durangobill.com/N\_Queens.html. Acesso em maio de 2012.

[5] Stephen J. Hartley. **“Parallel Processing”**, Disponível em: http://www.matematicas.unam.mx/jloa/parallel.html. Acesso em maio de 2012.

[6] Oracle. **“Synchronized Methods”**, Disponível em: http://docs.oracle.com/javase/tutorial/essential/concurrency/syncmeth.html. Acesso em maio de 2012.

**APÊNDICE A – AVALIAÇÃO COMPLETA DOS ALGORITMOS SEQUENCIAL, PARALELO E RECURSIVO**

Avaliação completa para N rainhas para 10 execuções de forma a obeter-se um média e desvio padrão. Para a avaliação de desempenho entre os algoritmos foram utilizados três computadores com configurações difentes. Os resutados completos que foram utilizados para o cálculo da média e desvio padrão estão disponíveis no Apêndice A.

Tabela A.1: Resultados do algoritmo sequencial executado no computador X

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Tempo de execução | | | | | | | | | |
| Rainhas | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |  |  |  |  |
| 13 |  |  |  |  |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |  |  |  |  |
| 16 |  |  |  |  |  |  |  |  |  |  |
| 17 |  |  |  |  |  |  |  |  |  |  |

Tabela A.1: Resultados do algoritmo sequencial executado no computador X

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Tempo de execução | | | | | | | | | |
| Rainhas | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |  |  |  |  |
| 13 |  |  |  |  |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |  |  |  |  |
| 16 |  |  |  |  |  |  |  |  |  |  |
| 17 |  |  |  |  |  |  |  |  |  |  |

Tabela A.1: Resultados do algoritmo sequencial executado no computador X

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Tempo de execução | | | | | | | | | |
| Rainhas | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |  |  |  |  |
| 13 |  |  |  |  |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |  |  |  |  |
| 16 |  |  |  |  |  |  |  |  |  |  |
| 17 |  |  |  |  |  |  |  |  |  |  |

Tabela A.1: Resultados do algoritmo sequencial executado no computador X

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Tempo de execução | | | | | | | | | |
| Rainhas | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |  |  |  |  |
| 13 |  |  |  |  |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |  |  |  |  |
| 16 |  |  |  |  |  |  |  |  |  |  |
| 17 |  |  |  |  |  |  |  |  |  |  |

Tabela A.1: Resultados do algoritmo sequencial executado no computador X

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Tempo de execução | | | | | | | | | |
| Rainhas | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |  |  |  |  |
| 13 |  |  |  |  |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |  |  |  |  |
| 16 |  |  |  |  |  |  |  |  |  |  |
| 17 |  |  |  |  |  |  |  |  |  |  |

Tabela A.1: Resultados do algoritmo sequencial executado no computador X

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Tempo de execução | | | | | | | | | |
| Rainhas | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |  |  |  |  |
| 12 |  |  |  |  |  |  |  |  |  |  |
| 13 |  |  |  |  |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |  |  |  |  |
| 16 |  |  |  |  |  |  |  |  |  |  |
| 17 |  |  |  |  |  |  |  |  |  |  |