**Capítulo 2**

**2.1 - Materiais e Métodos**

O principal problema a ser superado neste trabalho é a solução dos sistemas de equações lineares para obter as tensões dos nós dos circuitos são processados pelo módulo do sistema eletrônico de potência do DT sendo desenvolvido para o controle de operação da Usina Hidrelétrica de Jirau instalada no Rio Madeira no Estado de Rondônia, com aplicação para treinamentos, pré-operação, dentre outras. Estima-se que os circuitos possam atingir e impactar o desempenho do sistema, visto que este módulo do sistema é crítico, ou seja, outros modelos dependem da resolução do circuito para seguir o seu fluxo. Vale ressaltar, que os sistemas lineares típicos do DT possuem coeficientes complexos e uma estratégia para sua resolução é dividi-los em dois sistemas. Um sistema com a parte real e outro sistema com a parte complexa onde as variáveis são números complexos ou fasores.

Neste trabalho a análise de circuito será implementada usando tecnologia GPU CUDA para circuito da literatura como estudo de caso para que na sequência do estudo o software desenvolvido possa ser modificado e melhorado para a análise de circuitos reais no ambiente DT. A metodologia empregada para implementar o software experimental será a análise nodal porque a análise de malha só é utilizada para circuitos planos. O ambiente DT requer a análise de circuitos não planos.

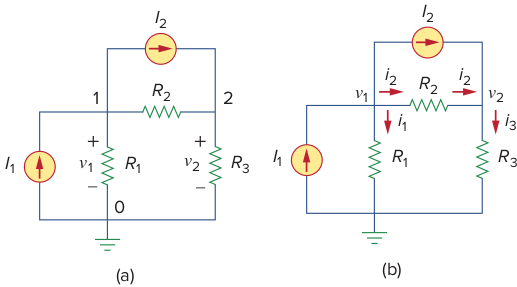


Figura 1. Circuito para análise nodal, adaptado de Alexander [5].

O objetivo da análise nodal é encontrar as tensões dos nós. Assim, dado um circuito com nós sem fontes de tensão, a análise nodal do circuito envolve as seguintes etapas. 1) escolha um nó como o nó referência (terra). Atribua as tensõespara os restantes [20]. As tensões são referenciadas em relação ao nó de referência; 2) Aplique a KCL a cada um dos nós não referência. Use a Lei de Ohm para expressar as correntes da ramificação em termos das tensões dos nós; 3) resolver as equações simultâneas resultantes do circuito para obter as tensões conhecidas dos nós [21].

**2.2 – Resolução de circuitos pela análise nodal**

O primeiro passo da análise nodal é selecionar um nó de referência. O nó de referência é chamado de terra. Assume-se que o nó de referência tem potencial zero. O aterramento do chassi é usado em dispositivos onde o gabinete ou chassi atua como um ponto de referência para todos os circuitos. Depois de selecionar um nó de referência, é atribuída a tensão aos nós não referência.

No circuito da Figura 1(a), o nó 0 é o nó de referência (). Os nós 1 e 2 recebem as tensões e respectivamente. As tensões do nó são definidas em relação ao nó de referência [22]. A Figura 1(a), mostra cada tensão do nó é o aumento da tensão do nó de referência para o nó não referência correspondente ou simplesmente a altitude desse nó em relação ao nó de referência.

A segunda etapa da análise nodal é feita aplicação da KCL a cada nó não referência do circuito. A Figura 1(b) mostra que e não são somados como as correntes através dos resistores e respectivamente. A aplicação da KCL para o nó 1 obtém-se

(1)

Até o nó 2, tem-se

(2)

Nessa etapa é aplicada a lei de Ohm para expressar as correntes desconhecidas e em termos das tensões nos nós [23]. Como a resistência é um elemento passivo, pela convenção dos sinais passivos, a corrente deve fluir de um potencial mais alto para um potencial mais baixo. Esse princípio é expresso por

. (3)

A partir desse princípio, a partir da Figura 1(b), obtém-se

ou

ou (4)

ou

Substituindo a Equação (4) na Equação (1) e Equação (2), tem-se, respectivamente

(5)

(6)

Em relação às condutâncias, Equação (5) e (6), tem-se

(7)

(8)

A terceira etapa da análise nodal é a resolução das tensões dos nós. Aplica-se a KCL aos nós não referência. Obtém-se equações simultâneas, como Equações (5) e (6) ou (7) e (8) [24]. Para o circuito da Figura 1, a solução das Equações (5) e (6) ou (7) e (8) produz as tensões dos nós e usando qualquer método padrão, como o método de substituição, o método de eliminação, a regra de Cramer ou a inversão da matriz [25]. Para usar um dos dois últimos métodos, é necessário converter as equações simultâneas na forma de matricial. As Equações (7) e (8) podem ser escritas na forma matricial



Considerando o circuito da Figura 1(a) para o uso da análise nodal e ganho computacional, tem se a multiplicação de matrizes em paralelo, ou seja, resolvendo cada nó paralelamente. Onde o nó1 e nó2 do nosso exemplo tem sua resolução vinculada a um construtor único, o qual fica responsável pela resolução dos cálculos. Para a divisão do problema existem kernels responsáveis por executar cada função repassada pela sintaxe da *Application programming interface* (API) CUDA. A definição por usar núcleos da API CUDA é feita pela declaração dos recursos invocados explicitamente*.* Isso significa que os dados são transferidos para a memória da GPU paralelizando as tarefas de resolução dos nós através do acesso à memória global da GPU.

**2.3 – A solução do sistema linear da equação paralelizada em GPU CUDA**

A solução do sistema linear da Equação (9) pode ser feita usando recursos GPU CUDA com processamento em paralelo através da distribuição dosfluxos de processamento independente *(threads)* que são uma das características da arquitetura CUDA [26].

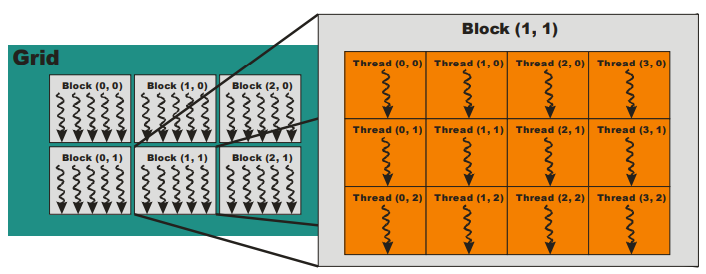


Figura 2. Elementos da arquitetura CUDA, Fonte NVIDIA CUDA [27].

A Figura 2 mostra os principais elementos da arquitetura CUDA. O processo de distribuição dos *threads* em CUDA ocorre pela definição de *blocks* no *grid* da placa gráfica. O *grid* é a configuração da placa para a execução de um *kernel*. *Block* forma o conjunto de *threads* que serão processadas simultaneamente em memória compartilhada. O processo de comunicação entre *blocks* ocorre através da memória global, a qual possui desempenho inferior a memória compartilhada. A identificação de *blocks* e *threads* é composto de identificadores tridimensionais, através destes o desenvolvedor tem acesso a *thread* a que deve processar [28].

**2.4 – O algoritmo do processo adotado**

No processo adotado na programação em C++ foi feita a implementação do Algoritmo 1 (abaixo) que é responsável pela multiplicação de matrizes, necessário para a resolução de sistema linear implicando em se computar a inversão da matriz, sendo esta uma matriz quadrada por resolução pela análise nodal.

|  |  |
| --- | --- |
| Algoritmo 1 - Multiplicar matriz | |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | Entrada:  **A** = matriz que está sendo multiplicada  **B** = matriz com os valores de potência  Saída:  **C** = matriz com os resultados dos cálculos  **R** = matriz de resultado  # função para multiplicar uma matriz por um vetor  # função decorada para cuda  Função = multiplicar vetor( A, b ):  **C** = **A** \* **b**  Retornar **C**  # função para multiplicar duas matrizes  Função = multiplicar matriz( **A**, **B** ):  # Para nas colunas da matriz ‘**B**’  Para inicio até fim colunas(b) faça  **R**[:, i] = multicar vetor( **A**, **B**[:,i] )  Retornar **R** |

No início do algoritmo temos os parâmetros de entrada compostos pela matriz **A** e **B**, o resultado delas é uma saída **R,** que corresponde a multiplicação da matriz **A** pelo inversa da matriz **B**. Na linha 2 pode-se informar os valores que compõem a matriz **A**, na linha 3 temos os valores da matriz **B**. Na linha 11 temos a função que processa a multiplicação direto na arquitetura CUDA, através da decoração da função executando o processamento na GPU e devolvendo para a CPU e retornando o resultado no vetor **C**. A linha 16 é responsável por invocar o processamento feito na linha 11 para cada elemento da matriz a serem calculados. Foram criadas funções que são passadas para a API CUDA para serem processadas as matrizes na GPU e enviadas ao CPU, para resolver os circuitos elétricos pela análise nodal na programação paralela.