Desempenho de Implementações MPI de Redes Reversíveis usando Comunicação Unilateral e Tipos de Dados Derivados

Matheus Liberato[1,2], Carlos Henrique Costa Ribeiro[1], Jairo Panetta[1]

[1]Instituto Tecnológico de Aeronáutica (ITA)
São José dos Campos/SP – Brasil

[2]Instituto Federal de São Paulo (IFSP)
Campos do Jordão/SP – Brasil

Abstract. *Due to a growing demand of protecting information, new cryptographic algorithms have been developed, such as those based on random reversible networks. The performance of MPI implementations of these algorithms is dominated by the amount of communication between processes, due to the structure of network connections. This paper investigates how to use One-Sided Communication and Derived Datatypes to speed-up such implementations.*

Resumo. *Devido à crescente necessidade de proteger informações, novos algoritmos criptográficos estão sendo desenvolvidos, como os algoritmos baseados em redes reversíveis aleatórias. O desempenho de implementações MPI desses algoritmos é dominado pela quantidade de comunicações entre processos, fruto da estrutura de conexão das redes. Este trabalho investiga como utilizar comunicação unilateral (One-Sided Communication) e tipos de dados derivados (Derived Datatype) para acelerar tais implementações.*

1. Introdução

Atualmente, podemos encontrar na literatura diversos algoritmos criptográficos baseados em Autômatos Celulares (ACs) [Roy et al. 2014, Anghelescu 2011, Tomassini and Perrenoud 2001]. Uma característica presente nos ACs é que pequenas perturbações no dado de entrada resultam em um dado de saída completamente diferente, o que é desejável para aplicações envolvendo criptografia.

O tipo básico de AC, denominado AC elementar, é composto por um vetor de n células binárias (valores 0 e 1) denominado *reticulado* e uma regra de transição determinística usada para atualizar os valores das células ao longo do tempo. Segundo [Wolfram 1984], este tipo de AC é definido formalmente pela Equação 1, em que a_i^t representa a célula binária a_i no tempo t, para $0 \leq i \leq n$. A vizinhança da célula a_i^t é dada pelas r células à esquerda e pelas r células à direita, sendo r denominado *raio*. A função de transição ϕ produz um valor binário que atualiza a célula a_i^{t+1} com base nos valores de vizinhança de a_i^t.

$$a_i^{t+1} = \phi[a_{i-r}^t, ..., a_{i+r}^t] \tag{1}$$

Um novo algoritmo criptográfico baseado em redes reversíveis foi proposto por [Macêdo 2014]. Ao contrário dos ACs tradicionais, em que a vizinhança considera obrigatoriamente as $2r$ células mais próximas de a_i, em redes reversíveis a vizinhança de a_i é

formada por qualquer conjunto de $2r$ células da rede, desde que um conjunto de restrições definidas por Macêdo seja respeitado. Utilizando a mesma nomenclatura da Equação 1, podemos definir formalmente uma rede reversível conforme apresentado pela Equação 2.

$$a_i^{t+1} = \phi[a^t_{f^i_{-r}}, ..., a^t_{f^i_r}] \tag{2}$$

As células que compõem a vizinhança de a_i^t são definidas pelas funções f^i, que podem assumir diversos comportamentos conforme a conexão de vizinhança desejada, podendo até mesmo estabelecer conexões de forma aleatória [Erdos and Renyi 1960]. Este tipo de rede é chamado de Rede Aleatória Reversível (AR).

Para cifrar um texto de m *bits* usando um sistema criptográfico baseado em redes reversíveis, cada *bit* da mensagem original corresponde a uma célula do reticulado. Os valores das células a_i^0, para $0 \leq i \leq n$, correspondem à configuração inicial do reticulado. O processo de cifragem, também conhecido como evolução temporal, ocorre após submeter todas as células a_i^t a uma regra de transição por k passos no tempo, sendo k um dado de entrada. Neste contexto, a regra de transição ϕ implementa a chave criptográfica do algoritmo [Gutowitz 1993].

É possível implementar uma versão paralela do algoritmo criptográfico baseado em redes reversíveis utilizando comunicação bilateral (*ISend/IRecv*), entretanto o código é complexo e de difícil manutenção. Neste trabalho apresentamos duas implementações usando comunicação unilateral (*One-Sided Communication*) e avaliamos seu desempenho. Estas implementações são muito mais simples de codificar e de manter.

2. Metodologia

Em uma abordagem paralela, cada processo é responsável por evoluir uma parte do reticulado inicial, ou seja, dado um reticulado com n células e p processos, cada processo será responsável por evoluir $\frac{n}{p}$ células, o que corresponde a uma divisão de domínio por blocos. A vizinhança de uma célula pode estar distribuída entre os p processos, sendo necessário obter esses valores através de troca de mensagens entre os processos.

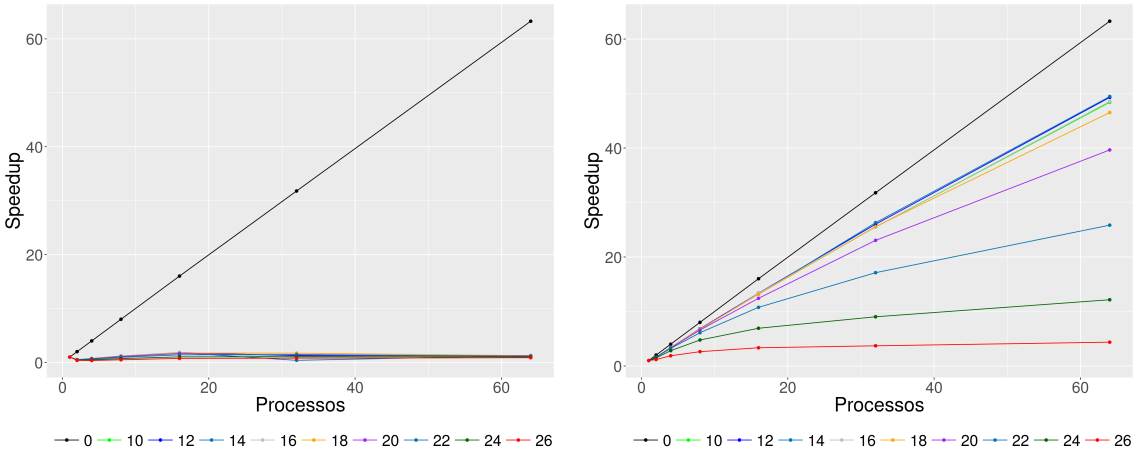
Neste trabalho foram desenvolvidas duas implementações usando o modelo de comunicação *One-Sided Communication*, também conhecido como *Remote Memory Access* (RMA) [Forum 2015, Pacheco 2011]. Em ambas as implementações, cada processo libera o acesso à região de memória referente ao reticulado em um trecho do código. Na primeira implementação cada comunicação acessa um único vizinho e traz um único dado. Na segunda implementação, antes de iniciar a comunicação, cada processo identifica os dados necessários de cada vizinho e acessa a memória exportada pelos vizinhos uma única vez, trazendo todos os dados necessários.

3. Resultados

Os experimentos descritos nessa seção foram executados no supercomputador SDumont. Todas as implementações foram feitas usando a linguagem de programação C em conjunto com a biblioteca MPI, disponível no Intel Parallel Studio XE 2017. As execuções utilizam um reticulado com 2^{26} células e 1, 2, 4, 8, 16, 32 e 64 processos. A Figura 1 apresenta o valor do *speedup* em função do número de processos para diversos volumes

de comunicação entre dois processos. Alterando os valores das funções f^i ajustamos o volume de comunicação entre dois processos de 2^0 a 2^{26} comunicações. A execução com 2^0 comunicações, representada pela linha na cor preta, é usada como um valor de referência para o *speedup* ideal.

A Figura 1(a) refere-se à primeira implementação, em que apenas um valor é obtido por comunicação. Como pode ser observado, os valores de *speedup* obtidos foram baixos se comparados ao valor de *speedup* ideal, pois o tempo gasto na comunicação, sincronismo e acesso ao dado na memória de outro processo é muito maior do que o tempo para atualizar uma célula do reticulado.



(a) Primeira implementação. (b) Segunda implementação.

Figura 1. *Speedup* em função do número de processos, variando de 2^0 até 2^{26} a quantidade de comunicações entre processos.

Uma característica das redes reversíveis é que as conexões entre as células da rede permanecem inalteradas durante todo o processo de evolução, permitindo, assim, o uso de tipo de dados derivados (*MPI Derived Datatype*) para mapear os dados necessários por cada processo. Deste modo, cada processo cria novos tipos de dados com as posições de memória desejadas uma única vez antes de iniciar a comunicação. Na segunda implementação, todos os dados necessários para evoluir o trecho do reticulado são obtidos em apenas uma comunicação.

Como pode ser visualizado na Figura 1(b), o tempo gasto na comunicação é amortizado, o que resulta em um valor de *speedup* mais próximo ao ideal. Isso ocorre pois o uso de tipo de dados derivados aumenta o volume de dados transmitidos entre processos por comunicação. Vale destacar que, apesar do *overhead* adicional no cálculo das posições de memória, a implementação usando tipo de dados derivados (segunda implementação) foi aproximadamente 37,72 vezes mais rápida[1] se comparada à implementação em que apenas um dado é retornado por comunicação (primeira implementação).

4. Conclusão e Trabalhos Futuros

O desempenho obtido por uma implementação paralela pode ser mensurado pela relação entre: o número de comunicações, o número de computações por comunicação e o

[1]Razão entre os tempos de execução das duas implementações com 2^{18} comunicações entre processos.

sincronismo entre as tarefas. Ao minimizar as comunicações entre entidades paralelas e maximizar a quantidade de computações que serão realizadas sobre os dados recebidos, é possível obter uma paralelização mais próxima do ideal. Neste artigo, a utilização de tipo de dados derivados e comunicação usando o modelo RMA possibilitou uma melhora de desempenho, aumentando o volume de dados transferidos por comunicação e reduzindo o número de comunicações necessárias para evoluir a rede. Redes reversíveis podem ser vistas como um tipo específico de grafo, em que cada vértice contém um valor e cada aresta corresponde à comunicação entre dois vértices distintos. Neste contexto, a utilização de tipo de dados derivados em conjunto com *One-Sided Communication* se apresenta como alternativa viável para obter melhor desempenho no processamento de grafos em geral. Futuramente, pretende-se investigar como aumentar o desempenho de uma implementação paralela de redes reversíveis criando redes com estrutura topológica de componentes conexas ao invés de redes aleatórias.

5. Agradecimentos

Os autores agradecem ao Laboratório Nacional de Computação Científica (LNCC/MCTI) por prover recursos de HPC do supercomputador SDumont, que contribuíram para os resultados apresentados nesse artigo. Também agradecemos ao Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP) por auxiliar esta pesquisa através da portaria de afastamento para capacitação N901.2016. Finalmente, agradecemos ao Instituto Tecnológico de Aeronáutica (ITA) por viabilizar esta pesquisa.

Referências

Anghelescu, P. (2011). Encryption algorithm using programmable cellular automata. In *2011 World Congress on Internet Security (WorldCIS-2011)*, pages 233–239.

Erdos, P. and Renyi, A. (1960). On the evolution of random graphs. In *PUBLICATION OF THE MATHEMATICAL INSTITUTE OF THE HUNGARIAN ACADEMY OF SCIENCES*, pages 17–61.

Forum, M. P. (2015). Mpi: A message-passing interface standard. Technical report, MPI Forum, Knoxville, TN, USA.

Gutowitz, H. (1993). *Cryptography with Dynamical Systems*, pages 237–274. Springer Netherlands, Dordrecht.

Macêdo, H. B. d. (2014). *Redes complexas e autômatos celulares aplicados à criptografia*. PhD thesis, Instituto Tecnológico de Aeronáutica.

Pacheco, P. (2011). *An Introduction to Parallel Programming*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition.

Roy, S., Nandi, S., Dansana, J., and Pattnaik, P. K. (2014). Application of cellular automata in symmetric key cryptography. In *2014 International Conference on Communication and Signal Processing*, pages 572–576.

Tomassini, M. and Perrenoud, M. (2001). Cryptography with cellular automata. *Applied Soft Computing*, 1(2):151 – 160.

Wolfram, S. (1984). Cellular automata as models of complexity. *Nature*, 311(5985):419–424.