Estudo de Redes Multiestágios em presença de Multicast

Caio Von Rondow Morais[1], Jose Nacif[1], Ricardo Ferreira[1]

[1]email: caio.rondow@ufv.br, Universidade Federal de Viçosa, Brazil

Resumo. *As redes multiestágio são uma alternativa econômica, com um custo de $O(n\log(n))$ em termos de elementos de chaveamento. No entanto, encontrar um roteamento válido nessas redes é um grande desafio. Uma solução possível é utilizar redes rearranjáveis, que podem ser reconfiguradas para conectar qualquer entrada a qualquer saída, reorganizando suas conexões internas para evitar bloqueios. No entanto, em aplicações reais, as conexões geralmente seguem um padrão multicast, o que torna o roteamento mais complexo, mesmo em redes rearranjáveis. Demonstramos que, embora a rede Benes seja amplamente reconhecida na literatura como a principal referência em redes rearranjáveis, ela perde essa propriedade na presença de multicast. Para contornar esse problema, exploramos redes Omega ou shuffle-exchange com estágios extras e mostramos que elas podem ser reorganizadas de maneira a evitar bloqueios. Como o espaço de busca de soluções é exponencial, utilizamos uma implementação em GPU, o que nos permitiu derivar algumas propriedades importantes. Também avaliamos algoritmos de aprendizado de máquina utilizando os dados gerados pela GPU. Finalmente, este estudo aponta novas direções para o desenvolvimento de redes rearranjáveis na presença de multicast.*

1. Introdução

Interconexões são essenciais na computação paralela [Ferreira et al. 2011]. Redes crossbar têm o custo é $O(n^2)$. Redes em malha são econômicas, mas enfrentam desafios de roteamento. Redes multiestágio (MINs) equilibram custo e desempenho, embora possam precisar de até $2\log(n)$ estágios para evitar bloqueios [Vendramini and Ferreira 2010].

Redes rearranjáveis podem estabelecer todas as conexões possíveis, mas o roteamento com múltiplos destinos ou *multicast* é NP-completo em redes Clos [Jastrzkebski and Kubale 2010], que foram pioneiras na década de 1950. Nos anos 1960, surgiu a rede Benes [Beneš 1965] que reduziu o custo, sendo rearranjáveis para conexões 1 para 1. Em 1968, um algoritmo polinomial para roteamento na rede Benes foi apresentado [Waksman 1968]. Apesar das redes multiestágio e Benes continuarem sendo importantes para conexão de computadores de alto desempenho em datacenters [Zhao et al. 2024], para FPGAs com memórias HBM [Choi et al. 2021] ou CGRA com compilação dinâmica [Silva et al. 2019], não existe nenhuma prova que a Benes e seu algoritmo sejam válidos em presença de multicast.

Este estudo propõe uma exploração do espaço de soluções de redes Benes e redes Omega. Como o espaço de configurações é exponencial, introduzimos o uso de GPUs na exploração. Como resultado, derivamos um caso que prova que a Benes que não é rearranjável com multicast. Uma abordagem exaustiva para redes com muitas conexões é inviável, porém mostramos que é possível observar várias propriedades mesmo em redes com 4 ou 8 conexões. Aplicamos também algoritmos de aprendizado de máquina para avaliar sua viabilidade na predição.

2. Fundamentos

Uma MIN é um conjunto de chaves ou *switches* conectados seguindo um padrão específico [Ferreira et al. 2018]. Ela pode ser classificada como bloqueante, rearranjável ou não bloqueante. Uma rede é bloqueante se não conseguir realizar alguma permutação de entrada/saída. É rearranjável se conexões podem ser reorganizadas para evitar conflitos. Uma rede é não bloqueante se qualquer conexão de uma entrada para uma saída puder ser roteada, sem a necessidade de modificar as outras conexões.

A rede Clos é não bloqueante ou rearranjável, dependendo da complexidade de seus switches internos. Ou seja, switches maiores evitam melhor os bloqueios mas aumentam os custos. Por outro lado, a rede Benes [Beneš 1965] simplifica a estrutura com switches simples 2x2. A Benes é rearranjável com um algoritmo polinomial sem conexões multicast [Waksman 1968]. Já a rede Omega (ou shuffle-exchange) [Lawrie 1975] tem um padrão regular de permutação para interconectar seus nós, mas é bloqueante com $\log(n)$ estágios. Porém, pode se tornar rearranjável ao adicionar mais $\log(n)$ estágios extras, apesar de ainda ser um ponto em aberto. Mais ainda, o número de configurações cresce exponencialmente, como por exemplo, 10^{35} para apenas $n = 32$ entradas/saídas.

2.1. Benes

Benes [Beneš 1965] é uma rede recursiva, onde a rede 8x8 é composta por duas sub-redes 4x4, e tem um algoritmo polinomial [Waksman 1968]. No entanto, sua reconfigurabilidade não foi provada na presença de multicast, no melhor conhecimento dos autores.

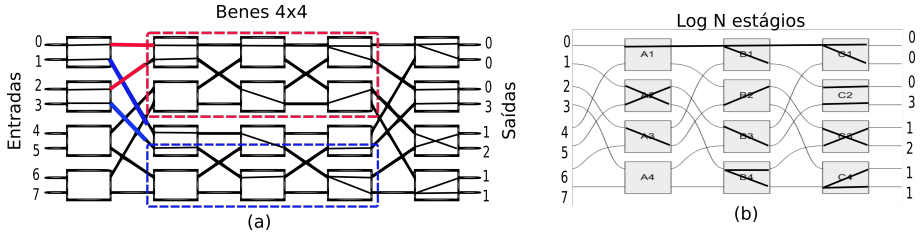

Figura 1. (a) Benes 8x8 com Multicast; (b) Omega com $\log n$ Estágios.

A Figura 1(a) mostra uma rede Benes 8x8. Podemos observar que ela é composta por duas sub-redes recursivas 4x4 (em vermelho e azul). O exemplo ilustra o roteamento do padrão multicast [0,0,0,3,1,2,1,1], onde a entrada 0 conecta nas saídas 0, 1 e 2; a entrada 3 na saída 3; a entrada 1 nas saídas 4, 6 e 7; a entrada 2 na saída 5. Para 8x8, temos $8^8 = 16$ milhões de combinações possíveis considerando multicast. A rede Benes da Figura 1(a) tem 5 estágios com 4 switches em cada, totalizando 20 switches. Cada switch tem 4 possíveis estados: direto, cruzado, multicast da entrada de cima e multicast da entrada de baixo. Portanto, existem $4^{switches} = 4^{20} = 1,1$ Tera ou um trilhão de configurações. Mesmo com este número de configurações muito maior que o número de combinações, existem combinações para as quais não existe nenhuma configuração.

É importante diferenciarmos os termos combinação e configuração. Combinação é o mapeamento da entrada para a saída. Por exemplo, a combinação [0,0,0,3,1,2,1,1] fará as conexões $0 \rightarrow 0, 0 \rightarrow 1, 0 \rightarrow 2, 3 \rightarrow 3, 1 \rightarrow 4, 2 \rightarrow 5, 1 \rightarrow 6$ e $1 \rightarrow 7$. Para executar esta combinação temos que configurar os switches. Supondo os códigos 0, 1, 2 e 3 para programar direto, cruzado, entrada de cima e entrada de baixo, respectivamente.

Na Figura 1(a) os dois primeiros switches do primeiro estágio estão configurados com 0. A configuração terá 40 bits, 2 bits para cada switch, como por exemplo, na Figura 1(a) temos $[s_1 = 0, s_2 = 0, \ldots, s_{20} = 3]$, onde s_1 é o primeiro switch e s_{20} seria o último, da esquerda para direita, de cima para baixo. Podemos ter mais de uma configuração para a mesma combinação. Neste exemplo, s_3 pode ser configurado com 0,1,2 ou 3 sem alterar o resultado, ou seja, teremos pelo menos 4 configurações para esta mesma combinação. Vale ressaltar que, 2^{40} configurações resulta em 1 trilhão de possibilidades.

2.2. Omega

Diferente das redes Clos e Benes, a rede Omega [Lawrie 1975] surgiu para atender a uma sequência de padrões bem comportados para conexão de memória e processador. A Figura 1(b) mostra uma rede Omega 8x8. Ao contrário da Benes que tem padrões diferentes entre os estágios, a Omega tem o mesmo padrão com $\log(n)$ estágios, enquanto a Benes sempre tem $2\log(n) - 1$ estágios para 2^n entradas.

Apesar de ter apenas 3 estágios, a Omega ilustrada na Figura 1(b) é capaz de rotear o mesmo padrão com multicast ilustrado para a Benes, que usa 5 estágios, na Figura 1. Entretanto, a Omega tem apenas 12 switches, portanto possui $4^{12} = 16$ milhões de configurações. Apesar do número de configurações ser o mesmo do número de combinações, existem muitas combinações que a Omega com 3 estágios não consegue fazer. Por exemplo, se quisermos conectar a entrada 6 na saída 6 da Figura 1(b), não é possível, pois o caminho que levaria a saída 6 já está sendo usado no primeiro estágio.

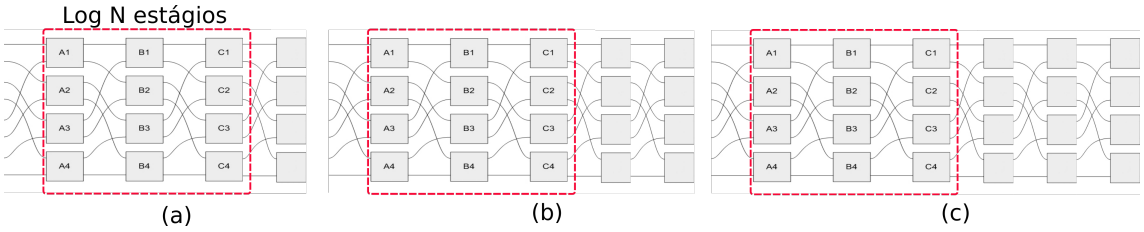

Figura 2. (a) Um estágio extra; (b) 2 Extras; (c) 3 Extras.

Este problema pode ser solucionado com estágios extras [Vendramini and Ferreira 2010] (Figura 2). Em termos de configurações, com 1, 2 e 3 extras temos 4 bilhões (4G), um trilhão (1T) e 256 trilhões de configurações, respectivamente. Ou seja, mesmo para redes pequenas, o número de configurações é grande, gerando um grande espaço de busca para os algoritmos de roteamento.

3. Combinações e Configurações para Redes 4x4

Para melhor entender o comportamento das redes, iremos avaliar primeiro a rede 4x4 que tem apenas 256 combinações, o que nos permite uma exploração exaustiva das soluções. Existem poucos trabalhos que buscam estudar as propriedades das redes [Wu and Feng 1980, Gazit and Malek 1989, Dai and Shen 2008]. Construímos um simulador em Python que gera todas as configurações de uma Omega com 2 e 3 estágios e para Benes que tem 3 estágios para 4x4. No caso da Omega com 2 estágios, temos 256 configurações possíveis para cobrir as 256 combinações, porém 112 combinações não são cobertas e a Omega é bloqueante com 2 estágios.

Tabela 1. Distribuição da Porcentagem de Configurações 4x4, 256 combinações para as Redes Omega com 1 extra e Benes.

Número de Configurações										
cfgs	2	4	8	10	16	20	32	100	176	Total
qte	48	72	40	16	16	16	36	8	4	4096
porc	18,7 %	28,1 %	15,6 %	6,2 %	6,2 %	6,2 %	14,1 %	3,1 %	1,6 %	100 %

A Tabela 1 mostra a distribuição da quantidade de configurações[1] para cada combinação na Benes 4x4 e na Omega $+1$ extra, que são equivalentes. Existem 48 combinações que tem apenas 2 configurações das 4.096 disponíveis. Ou seja, dependendo da combinação pode ser mais fácil ou mais difícil achar uma programação ou configuração. Existem 36 combinações que tem 32 possíveis padrões, sendo mais fácil para encontrar uma configuração válida do que as combinações que só tem apenas duas. Por exemplo, para a combinação [1,2,3,0] só tem 2 configurações e para [0,3,3,3] existem 32 e para [3,3,3,2] tem apenas 10.

4. GPU para Explorar as Redes 8x8

Como mencionado, a Benes tem 1 Trilhão de configurações para cobrir as 16 milhões de combinações e a Omega com 1, 2 e 3 estágios extras tem 4 Giga, 1 Tera e 256 Tera configurações. Devido ao tamanho do espaço de busca fizemos uma implementação em GPU para descobrir quantas configurações existem para cada combinação.

Criamos um vetor com 16 milhões de entradas, uma para cada combinação. Cada thread irá gerar um conjunto de configurações e cada configuração será acrescentada na entrada correspondente no vetor de combinações. Assim saberemos quantas configurações existem para cada combinação. O primeiro resultado foi que, para Omega com 0, 1 e 2 estágios extras existem 97%, 60% (10 milhões) e 1,3% (226 mil) combinações que são bloqueantes, ou seja, não existe nenhuma configuração para aquela combinação. Apenas a rede com 3 extras conseguiu rotear todas as combinações e, portanto, é rearranjável. A Benes foi criada como a primeira rede rearranjável 1 para 1, porém não é rearranjável em presença de multicast e 10% das combinações não tem solução. No melhor conhecimento dos autores, não existe uma prova formal que a Benes não é rearranjável com multicast.

5. Benes não é rearranjável com Multicast

A implementação com GPU explorou um trilhão de configurações e mostrou que 10% das combinações com multicast não podem ser resolvidas com a Benes. Por exemplo, o padrão [0,0,0,0,1,2,1,3] é bloqueante, e foi derivado da exploração com a GPU, onde verificamos 13 bilhões de configurações por segundo. Este padrão é parecido com o padrão ilustrado na Figura 1(a) que é roteável, porém ao modificarmos a conexão da entrada 3 para a saída 7, a Benes deixa de ser. O motivo é que ao usar a sub-rede de cima (em vermelho), a entrada 0 aloca todas as saídas da sub-rede para ter acesso aos dois switches superiores de saída, restando para entrada 1 usar a sub-rede de baixo (em azul) para conectar os dois switches inferiores de saída. Apesar da entrada 2 conseguir rotear usando a sub-rede de cima (vermelho), a entrada 3 só poderá usar a sub-rede de baixo que

[1] A coluna Total é o somatório do produto da quantidade de combinações (qte) e as configurações (cfgs).

não tem nenhuma saída disponível, porque já foram alocadas pela entrada 1. Portanto, é um padrão bloqueante. Para fazer uma prova por contraexemplo, iremos usar padrões derivados deste exemplo e generalizar.

A Figura 3 ilustra um contraexemplo para prova que uma rede Benes $2^n \times 2^n$ não é rearranjável. Outra alternativa de prova é por indução, uma vez que se a Benes 8x8 é não rearranjável, então a Benes 16x16 também não será rearranjável pois é composta internamente por duas sub-redes 8x8 e assim sucessivamente. A seguir, vemos a prova com contraexemplo.

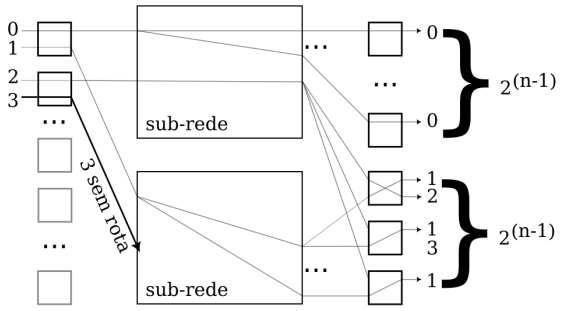

Figura 3. Contraexemplo onde a Benes não é rearranjável com multicast.

Proposição: Uma rede Benes (2^n entradas e saídas) é rearranjável em presença de multicast.

Contraexemplo: A combinação $[\underbrace{0, c_1, 0, c_2 \ldots, 0, c_{2^{n-1}}}_{2^{n-1}}, \underbrace{1, 2, 1, 3, \ldots, 1, b_{2^{n-1}}}_{2^{n-1}}]$ é bloqueante, portanto a rede Benes não é rearranjável em presença de multicast, onde c_i e b_i podem ser qualquer entrada conectada na parte de cima ou de baixo das saídas, respectivamente.

Prova: A Benes terá duas sub-redes de tamanho 2^{n-1}. Se conectarmos a entrada 0 na sub-rede de cima, todas as saídas da sub-rede serão usadas para conectar a entrada 0 aos switches de saída da parte de cima. Para a entrada 1 só temos a opção de usar a sub-rede de baixo, como 1 está conectado a todos os switches da parte de baixo, todas as saídas das sub-rede para parte de baixo estarão alocadas. Para conectar a entrada 2 a segunda saída do primeiro switch da parte de baixo, podemos usar a sub-rede de cima. Restando então apenas a sub-rede de baixo para entrada 3. Como a entrada 3 se conecta aos switches da parte de baixo e todas as saídas da sub-rede já estão ocupadas pela entrada 1, não é possível conectar. Portanto a Benes não consegue resolver esta combinação com multicast.

6. Resultados Experimentais

6.1. Redes 4x4

Devido a complexidade para modelagem analítica das redes multi-estágio [Gazit and Malek 1989], propomos a avaliação de modelos de aprendizado de máquina. Estes modelos podem simplificar as tarefas de posicionamento de aplicações em redes multiestágios [Silva et al. 2019, Morais et al. 2023]. Primeiro, iremos avaliar as redes 4x4. A proposta é ter um primeiro estudo de casos. Construímos um conjunto

de dados para Omega mínima que é bloqueante, queremos saber se o modelo pode dizer se uma combinação é roteável ou não. Como o conjunto de dados tem apenas 256 combinações, ao criar os modelos KNN, Random Forest e Rede Neural, usando as opções padrões do scikit-learn, obtivemos a acurácia de 64%, 73% e 66%, respectivamente.

Repetimos o experimento para a Omega com 1 extra, queremos predizer o número de configurações (problema de regressão). O erro médio quadrático na regressão foi de 13,04% para a Rede Neural, 14,79% para o Random Forest, 12,92% para o SVM, e 11,07% para o Gradient Boosting, respectivamente. Ou seja, dada uma combinação podemos predizer quantas configurações existem com um erro próximo de 12%. Por exemplo, a combinação [1,1,0,2] teve uma predição de 9,4 configurações, o valor real são 8 configurações em 256, então teve uma predição que existem 3% de chances de encontrar a configuração correta em 256 possibilidades.

6.2. Redes 8x8

Já para redes 8x8, temos $8^8 = 16$ milhões de combinações, sendo que, para 3 estágios extras tem 256 trilhões de configurações. A Figura 3 mostra um histograma com a distribuição das configurações em cada combinação. Como o número de configurações pode ser grande, utilizamos o $log_2(configs)$. Tem 2048 combinações muito difíceis pois só tem 432 configurações em um espaço de 256 trilhões, ou seja, praticamente 1 em 1 trilhão. A partir do histograma, dividimos os dados em 4 classes, fácil(>20), médio(<20), difícil(<17) e muito difícil (<14). Por exemplo a combinação c=[0,0,0,3,1,2,1,1] possui 4 apenas entradas (0, 1, 2 e 3) pois possui multicast de três nas entradas 0 e 1. Usamos esta informação para ajudar o modelo fazer uma correlação com multicast e a dificuldade de se fazer uma determinada combinação. Mais multicast, em geral, fica mais fácil, mas apenas um pouco de multicast não quer dizer que seja mais fácil.

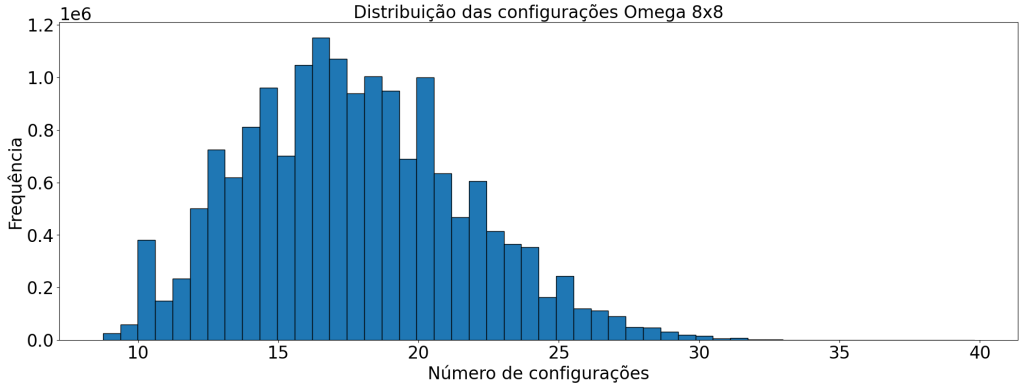

Figura 4. Distribuição das configurações (em log_2) Rede Omega 8x8, 3 estágios.

Treinamos um modelo xgboost com a MLogLoss para um cenário multi-classe, usando apenas 10% dos dados (1,6 milhões de combinações), divididos em 80% treino, 10% teste e 10% validação. O modelo atingiu 73% de acurácia na validação e teste.

A Tabela 2 apresenta a matriz de confusão de teste. A análise da matriz revela que o modelo pode diferenciar a classes Muito Difícil das classes Médio/Fácil.

Os valores na diagonal principal (em **negrito**) representam as predições corretas. Os valores em vermelho indicam os falsos positivos, ou seja, instâncias que foram clas-

Tabela 2. Matriz de Confusão de Teste.

Valor Correto	Muito Difícil	Difícil	Médio	Fácil
Muito Difícil	**397.235**	21.211	616	0
Difícil	104.962	**232.799**	81.700	469
Médio	7.792	95.557	**262.995**	51.938
Fácil	0	1.604	86.029	**332.815**

sificadas incorretamente como pertencentes a uma classe diferente. Os valores em azul representam os falsos negativos, ou seja, instâncias que pertencem a uma determinada classe, mas foram classificadas incorretamente como pertencentes a outra classe.

Um fato interessante é que, apesar das combinações sem multicast serem mais difíceis, haja vista que todos os elementos de chaveamento devem ser configurados, a distribuição não é uniforme. Existem 2.048 combinações que são as mais difíceis pois tem apenas 432 configurações em um espaço de 1 trilhão. Existem 2.048 configurações que também são unicast mas são três vezes mais fáceis pois possuem 1.164 configurações como ilustra a Figura 5 com a distribuição das $!8 = 40320$ combinações unicast.

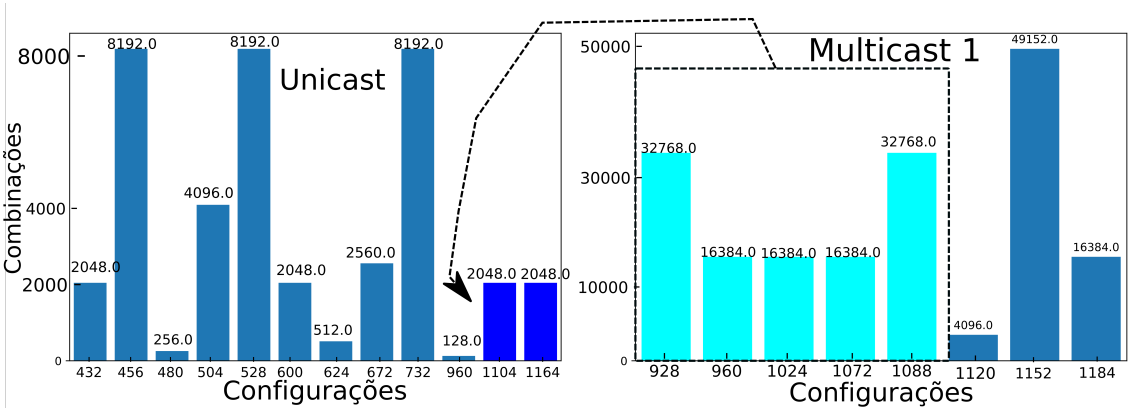

Figura 5. Distribuição das configurações para as 40.320 combinações unicast na esquerda e as combinações mais difíceis com 1 multicast na direita para Omega 8x8, 3 estágios.

Outro fato interessante é que existem 114.688 combinações com 1 multicast que são mais difíceis que a combinação mais fácil sem multicast que tem 1.164 configurações, enquanto as de multicast tem entre 928 e 1.088 configurações como ilustrado na Figura 5.

7. Conclusões

Este estudo investigou as propriedades das redes de interconexão Benes e Omega, com foco na rearranjabilidade em cenários que envolvem multicast. Devido a complexidade, não é possível estudar explorar o espaço de redes com muitas conexões, mas foi possível mostrar novas propriedades derivadas em redes com poucas entradas. A análise revelou que, embora a rede Benes seja amplamente reconhecida na literatura por sua capacidade de rearranjar conexões em configurações 1 para 1 (unicast), essa propriedade não se sustenta na presença de multicast. Por outro lado, as redes Omega, especialmente quando expandidas com estágios adicionais, mostraram-se promissoras em contornar essa limitação, embora a comprovação definitiva de sua rearranjabilidade em contextos de multicast ainda

demande investigações futuras. A exploração computacional do espaço de soluções, facilitada pelo uso de GPUs e a integração de algoritmos de aprendizado de máquina na análise dos dados gerados demonstrou potencial para aprimorar a predição e otimização das configurações de rede, sugerindo novas direções de pesquisa para uso das redes multiestágios em computação de alto desempenho considerando multicast, que pode estender os trabalhos atuais que são limitados a unicast [Zhao et al. 2024, Choi et al. 2021].

Agradecimentos

Apoio financeiro da Bolsa de Iniciação Científica do CNPq, programa PIBIC Institucional, do CNPq, da FAPEMIG APQ-01577-22, CAPES e da UFV.

Referências

Beneš, V. E. (1965). *Mathematical theory of connecting networks and telephone traffic*. Academic press.

Choi, Y.-k., Chi, Y., Qiao, W., Samardzic, N., and Cong, J. (2021). Hbm connect: High-performance hls interconnect for fpga hbm. In *ACM FPGA*.

Dai, H. and Shen, X. (2008). Rearrangeability of 7-stage 16× 16 shuffle exchange networks. *Frontiers of Electrical and Electronic Engineering in China*, 3:440–458.

Ferreira, R., Canesche, M., Coelho, K., and Nacif, J. (2018). Minimum switching networks. In *Brazilian Symposium on Computing Systems Engineering (SBESC)*.

Ferreira, R. S., Cardoso, J. M., Damiany, A., Vendramini, J., and Teixeira, T. (2011). Fast placement and routing by extending coarse-grained reconfigurable arrays with omega networks. *Journal of Systems Architecture*, 57(8):761–777.

Gazit, I. and Malek, M. (1989). On the number of permutations performable by extra-stage multistage interconnection networks. *IEEE trans on computers*, 38(2).

Jastrzkebski, A. and Kubale, M. (2010). Rearrangeability in multicast clos networks is np-complete. In *International Conference on Information Technology,(ICIT)*. IEEE.

Lawrie, D. H. (1975). Access and alignment of data in an array processor. *IEEE Transactions on computers*, 100(12):1145–1155.

Morais, C. V. R., Penha, J., Nacif, J. A., and Ferreira, R. (2023). New kids on the unblocking: Strategies to overcome blocking networks. In *Simpósio em Sistemas Computacionais de Alto Desempenho*.

Silva, L., Ferreira, R., Canesche, M., Penha, J., , and Nacif, J. (2019). Ready: A fine-grained multithreading overlay framework for modern cpu-fpga dataflow applications. *ACM Transactions on Embedded Computing Systems (TECS)*, 18.

Vendramini, J. C. G. and Ferreira, R. (2010). Parallel routing algorithm for extra level omega networks on reconfigurable systems. In *Symposium on Computing Systems*.

Waksman, A. (1968). A permutation network. *Journal of the ACM (JACM)*, 15(1).

Wu, C.-L. and Feng, T.-Y. (1980). On a class of multistage interconnection networks. *IEEE trans. on Computers*, 100(8).

Zhao, L., Li, Z., and Ma, T. (2024). Making path selection bright: A routing algorithm for on-chip benes networks. *Electronics*, 13(5):981.