IrNoC: Uma Ferramenta de Simulação Para Redes-em-Chip Irregulares

1st Samuel da Silva Oliveira
Departamento de Informática e Matemática Aplicada
Universidade Federal do Rio Grande do Norte
Natal, Brazil
samueloliveira@ppgsc.ufrn.br

2nd Márcio Eduardo Kreutz
Departamento de Informática e Matemática Aplicada
Universidade Federal do Rio Grande do Norte
Natal, Brazil
kreutz@dimap.ufrn.br

ABSTRACT

Networks-on-Chip is an interesting approach for large Systems-on-Chip interconnection when multiple communication paths allow for performance improvement and scalability. Switches inside routers define routes for any type of traffic pattern. During design phase network models need to be evaluated concerning constraints through simulation tools. Most simulator available focus on networks having regular topologies. This paper presents the IrNoC Simulator, a tool dedicated to simulating irregular topologies concerning latency and the number of packages that fulfill their deadlines. Several experiments compare evaluations for regular and irregular approaches to network topology construction.

Keywords

Simulator, IrNoC, Irregular Topology, Network-on-Chip, System-on-Chip

RESUMO

Redes-em-Chip é uma abordagem interessante para grandes Sistemas-em-Chip com vários caminhos de comunicação, permitindo uma melhoria no desempenho e escalabilidade. Os comutadores dentro dos roteadores definem rotas para qualquer tipo de padrão de tráfego. Durante a fase de projeto, os modelos de rede precisam ser avaliados em relação às restrições por meio de ferramentas de simulação. A maioria dos simuladores disponíveis concentra-se em redes com topologias regulares. Este artigo apresenta o simulador IrNoC, uma ferramenta dedicada à simulação de topologias irregulares relacionadas à latência e ao número de pacotes que cumprem seus prazos. Vários experimentos comparam avaliações de abordagens regulares e irregulares para a construção de topologia de rede.

Palavras-chave

Simulador, IrNoC, Topologia Irregular, Rede-em-Chip, Sistema-em-Chip

1. INTRODUÇÃO

A fabricação de Sistemas-em-Chip (*Systems-on-Chip* - SoC) [1] teve um grande aumento na última década e surgiram sistemas mais complexos, necessitando de arquiteturas confiáveis. Para atender a essas necessidades, as Redes-em-Chip (*Networks-on-Chip* - NoC) [8] foram desenvolvidas.

A NoC consiste em um conjunto de roteadores (comumente ligados a um *IP/CORE* formando uma topologia. As comunicações são realizadas através de troca de pacotes entre os roteadores. A etapa de troca de pacotes se dá através do uso de algoritmos de roteamento. A NoC é escalável, possui suporte a paralelismo. A otimização é essencial para uma melhor comunicação e menos latência. Vários parâmetros podem ser trocados para obter restrições de *design*, como topologia, números de roteadores, tipo de roteamento e arbitragem.

A escolha da topologia é uma decisão importante, porque a topologia tem um grande impacto na questão de área, latência e consumo de energia. Topologias irregulares trazem melhorias para as três métricas mencionadas acima [12]. Em uma topologia irregular, cada roteador pode ter um link para qualquer roteador na rede, facilitando o processo de comunicação.

Na parte da exploração do espaço de projeto é testado o desempenho da NoC para algumas restrições predefinidas. A avaliação é feita através de ferramentas de simulação, essas ferramentas devem ter alta precisão e confiabilidade nos seus resultados. Na literatura existem várias ferramentas para simulações de desempenho de NoC, mas a maioria delas é destinada para topologias regulares.

Este artigo tem como objetivo apresentar um simulador para topologias irregulares. O IrNoC é um simulador descrito em SystemC [9]. O IrNoC trabalha com métricas como latência e *deadline* de pacotes, ajudando o projetista a avaliar o desempenho da rede-em-chip. Nosso objetivo é mostrar mais uma ferramenta alternativa, em comparação com outras ferramentas existentes.

Esse artigo está organizado da seguinte forma: A Sessão II apresenta alguns simuladores para topologias irregulares. A Sessão III mostra a arquitetura da ferramenta IrNoC. A Sessão IV detalha os resultados avaliados, enquanto a Sessão V discute os resultados e sugere trabalhos futuros.

2. TRABALHOS RELACIONADOS

Na literatura, diversos simuladores de NoC podem ser encontrados e usados pela comunidade de pesquisadores e projetistas. Nesta sessão são apresentados algumas ferramentas de simulação de redes-em-chip irregulares.

A ferramenta IrNirgam [2] é baseada no Nirgam e escrita SystemC e C ++ com foco em topologias irregulares. Essa ferramenta trabalha com topologia irregular, *Mesh* e *Torus*. Todas as configurações, como topologia, tamanho, simulação, geração de tráfego, algoritmos de roteamento e tamanho do *buffer*, são fornecidas através de arquivos de configuração. Em tempo de execução, os *cores* das aplicações e os algoritmos de roteamento podem ser conectados ao mecanismo principal sem a necessidade de compilar o código novamente.

Latência e taxa de transferência são métricas de desempenho que o IrNirgam retorna como saída. A ferramenta usa comutação de pacotes do tipo *wormhole* para controle de fluxo. Os pacotes são divididos em *flits*, o primeiro *flit* é cabeçalho e contém informações sobre o destino, o último *flit* é conhecido como calda e os *flits* intermediários são *flits* de dados.

O mecanismo principal do simulador implementa modelos de redes-em-chip como uma rede de blocos que forma um módulo NoC. A topologia define a maneira como vários nós são interconectados na rede. Todas as topologias são caracterizadas por algumas propriedades:

- Diâmetro: A distância máxima entre dois nós da rede;
- Largura da bisseção: O número mínimo de fios que devem ser cortados quando a rede é dividida em dois conjuntos iguais de nós;
- Grau do nó: Número de canais que conectam um nó a seus vizinhos.

O IrNirgam se preocupa em avaliar métricas de desempenho para vazão e latência.

Vazão: No IrNirgam, a vazão (em Gbps) é calculada para cada canal de saída do roteador separadamente, considerando a quantidade de trechos que passam por cada roteador.

Latência: A latência é definida como o atraso entre a injeção em um nó de origem e a recepção do último pacote no nó de destino. Na ferramenta IrNirgam, a latência média é medida em ciclos de clock e é estimada para cada canal de saída e para toda a rede.

NC-G-SIM [13] suporta as topologias *Mesh*, *Torus*, *Mesh* 3D e topologias irregulares, os algoritmos de roteamento XY e o *odd-even*. A topologia 3D usa o algoritmo XYZ como um esquema de roteamento baseado em tabela bem distribuído. O roteamento pelas tabelas é feito usando arquivos no formato [*sourceID*, *destinationID*, *nexttileID*, *pathId*], a cada caminho é atribuído um ID exclusivo '*pathID*' para tornar os caminhos facilmente distinguíveis. Cada nó decide a próxima direção de roteamento com base no arquivo de tabela correspondente do nó e nas entradas. Essas tabelas são preenchidas com base no conhecimento localizado da rede.

O simulador NC-G-SIM suporta seis vizinhos de qualquer bloco, contém um número máximo de 500 blocos para topologias irregulares e 3D. A característica de tráfego é essencialmente necessária para executar uma simulação. O tráfego é gerado através das ferramentas CBR e Bursty. O NC-G-SIM implementa adicionalmente o tráfego BWCBR, gerando tráfego simultâneo de um nó de origem para mais de um destino, tornando as condições de simulação mais reais.

A ferramenta recebe como entrada um arquivo com parâmetros de simulação, onde faz a escolha da topologia, outro arquivo com configuração de tráfego como entrada e um arquivo com tabelas de roteamento. A saída é um arquivo com resultados de simulação, que inclui métricas de desempenho, latência, taxa de transferência e dissipação de energia, além de mostrar logs e gráficos.

Os parâmetros configuráveis da ferramenta são o tamanho das topologias, tamanho do *buffer*, tamanho do pacote e *flit*, intervalo do pacote e número de canais virtuais. A frequência usada é de 1 Ghz, que pode ser alterada. A comutação de pacotes *wormhole* é usada, onde os pacotes são serializados em várias faixas, portanto, *buffers* menores podem ser usados em contraste com a técnica de comutação de pacotes. No final da simulação, a estrutura do NC-G-SIM gera um arquivo de logs que é detalhado e conciso ao alternar diferentes níveis de log. Os resultados detalhados da simulação podem ser lidos, verificados e comparados em termos de carga de *flit*, dissipação de energia, taxa de transferência, latência e desvio padrão do arquivo de resultados gerado na conclusão da simulação. Os gráficos podem ser plotados usando o GnuPlot [7], também podem ser plotadas essas métricas de desempenho de saída.

GPNOCSIM [5] é um simulador de uso geral e suporta todas as topologias. A ferramenta é desenvolvida em Java, além disso, seu *design* modular ajuda a incorporar novas arquiteturas.

No GPNOCSIM, cada *switch* possui muitas portas para conectar-se a *switches* adjacentes e blocos de propriedade intelectual (*Intellectual Property* - IP). Cada porta contém dois conjuntos de buffers (entrada e saída), um roteador e um controlador. O roteador decodifica o fluxo do cabeçalho e determina a porta de saída. O controlador é particionado em dois controladores independentes, um para entrada de canal e outro para canal de saída. O árbitro usado é *Round-Robin*.

A ferramenta usa técnicas de comutação de pacotes do tipo *wormhole* e usa canais virtuais em cada porta. O GPNOCSIM contém várias distribuições de tráfego, onde os usuários podem definir qual será usada. Os parâmetros de entrada são definidos através de arquivos de configuração. A simulação termina quando todos os ciclos foram concluídos.

Abaixo, é apresentada a primeira tabela que realiza a comparação e as diferenças entre o IrNoC e os outros simuladores apresentados nessa sessão, ajudando a mostrar a contribuição da ferramenta desenvolvida aqui.

O simulador IrNoC foi o único encontrado que avalia topologias irregulares para pacotes com restrições de tempo.

3. DESCRIÇÃO E ARQUITETURA DO SIMULADOR

O IrNoC é uma ferramenta de simulação para topologias irregulares de Redes-em-Chips, desenvolvida no nível TLM da biblioteca SystemC . A entrada para a ferramenta é um grafo de aplicação através, esse grafo é definido através de um arquivo contendo distribuições de tráfego.

O esquema de roteamento é baseado em tabelas, nas quais o algoritmo de menor caminho encontra as melhores distâncias entre cada par de roteadores em uma comunicação. O

Table 1: Comparação dos Simuladores

Tabela da Comparação dos Simuladores				
Simulador	Latência	Deadline	Pacotes Tempo Real	Canais Virtuais
IrNirgam	X			X
NC-G-SIM	X			X
GPNOCSIM	X			X
IrNoC	X	X	X	X

IrNoC emprega comutação *wormhole* e esquema de arbitragem *Round-Robin*.

O simulador suporta topologias irregulares direcionadas a aplicações em tempo real e não tempo real. Como resultado, o IrNoC avalia a latência média e o número de pacotes que podem cumprir seus *deadlines*.

3.1 Fluxo de Projeto

A Figura 1 mostra o fluxo de avaliação do IrNoC. O simulador começa lendo o arquivo que contém as informações de tráfego da rede, passando esses dados como entrada para a ferramenta de simulação. Quando os dados de tráfego chegam ao simulador, a topologia irregular é formada pelo algoritmo de menor caminho através dessas informações e, em seguida, a rede é simulada roteando os pacotes até atingir os prazos previstos para cada pacote. Ao final, os resultados relacionados às métricas de simulação são gerados, completando assim todo o fluxo de avaliação.

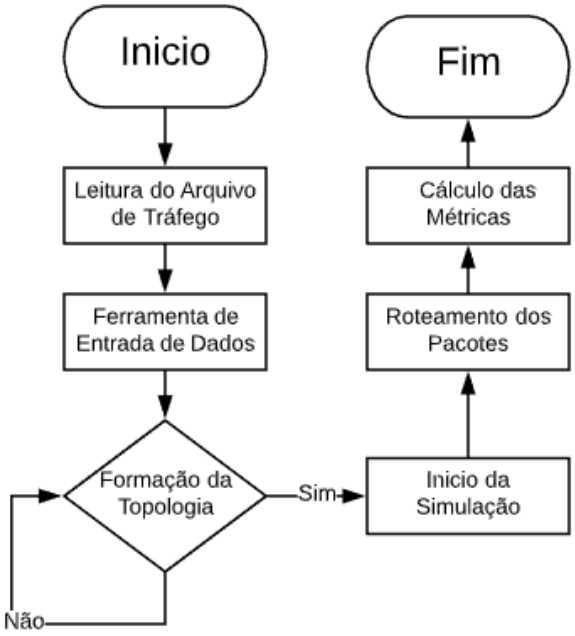

Figure 1: Fluxo da Arquitetura do IrNoC

Os padrões de tráfego são definidos usando um arquivo de distribuição, que contém os roteadores de origem e destino, o número total de pacotes e o prazo de entrega dos pacotes.

Os dados são inseridos na ferramenta e a topologia é calculada através das fontes e destinos no arquivo de tráfego. O algoritmo de Floyd-Warshall (FW) [6] é usado para calcular a menor distância. O simulador permanece nessa etapa até a topologia ser formada.

A simulação da IrNoC começa após a formação da topologia e a simulação termina quando o tempo de simulação é igual ao *deadline* do pacote, semelhante à uma aplicação em tempo real.

Quando a simulação termina, são calculadas as métricas de latência e quantidade de pacotes que cumpriram seus *deadline*.

A latência é calculada através da soma da latência de todos os pacotes sobre o número total de pacotes enviados. Abaixo, pode-se ver a fórmula usada para este cálculo.

$$Latência\ Média = \sum_{i=1}^{Nº\ de\ Pctes} \frac{Latência\ de\ Pcte}{Nº\ de\ Pctes} \quad (1)$$

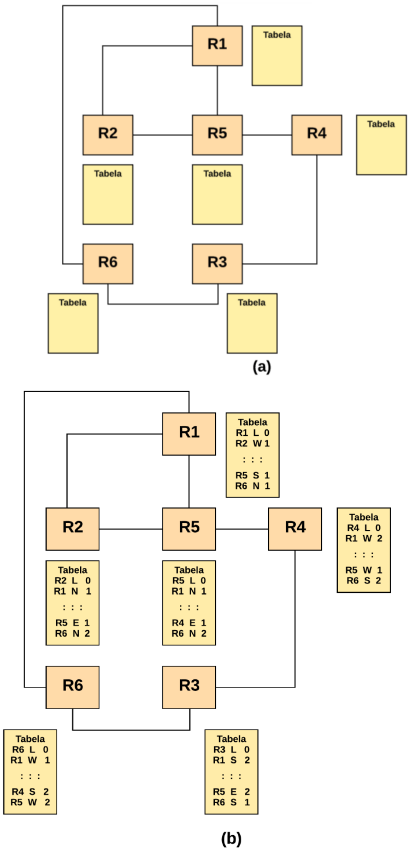

Figure 2: Processo de Formação da Topologia

3.2 Topologia

A topologia é formada através do arquivo de tráfego. Os roteadores de origem e destino são usados como entrada para a ferramenta de geração de tráfego, e o algoritmo Floyd-Warshall devolve o menor caminho entre os roteadores de origem e os roteadores de destino. Quando a topologia é concluída, as tabelas de roteamento são criadas. O FW constrói as tabelas usando o menor caminho para cada roteador na rede. A Figura 2 mostra esse processo.

As tabelas de roteamento são formadas por cada endereço do roteador na rede. Cada tabela contém três campos: destino, porta de saída e o número de saltos (*hops*). A linha da tabela contém os valores para cada roteador. O campo de destino contém o roteador de destino para onde os pacotes serão enviados; o campo da porta de saída é usado para mostrar qual porta será usada para enviar os pacotes; o campo número de *hops* mostra o número de saltos para o roteador de destino. Um exemplo da tabela de roteamento pode ser visto na figura 3.

3.3 Roteadores

O IrNoC usa dois tipos diferentes de roteadores, um roteador genérico e outro com canais virtuais. Esses dois tipos de roteador são descritos na figura 4.

Cada roteador pode ter um número de portas variável para comunicação com outros roteadores, restrito a um mínimo de quatro. Cada roteador contém portas de entrada e saída, *buffers* em cada porta de entrada, árbitros, tabela de roteamento e implementação de algoritmo de roteamento.

Opcionalmente, os roteadores também podem ser configurados com canais virtuais preemptivos (CVP), úteis para pacotes em tempo real.

Destino	Porta de Saída	Número de Hops
R1	E	1
R2	W	0
R3	N	2
R4	S	1
R5	N	1
R6	N	2
R7	E	3
R8	E	2

Figure 3: Exemplo de Tabela de Roteamento

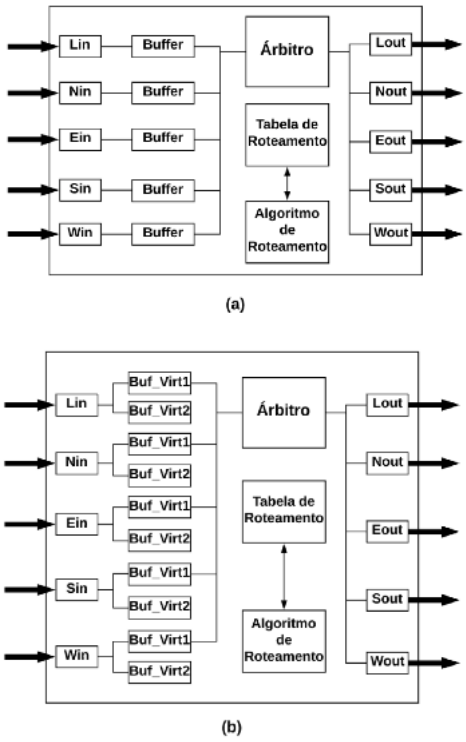

Figure 4: Exemplos dos Modelos de Roteadores

4. AVALIAÇÃO DE DESEMPENHO

Para avaliar o desempenho, foram considerados oito grafos que variam o *deadline* das tarefas. Foram geradas aplicações sintéticos usando a ferramenta TGFF [4]. Os experimentos foram realizados em dois conjuntos de aplicações com 10 e 15 tarefas, respectivamente. As aplicações diferem no padrão de comunicação, quantidade de dados a serem transferidos, tipo de pacotes (tempo real e não tempo real) e *deadline* dos pacotes.

Foram testados deadlines longos, médios e curtos, variando entre 500 ns e 1200 ns. As figuras 5 a 8 mostram os grafos usados para testes.

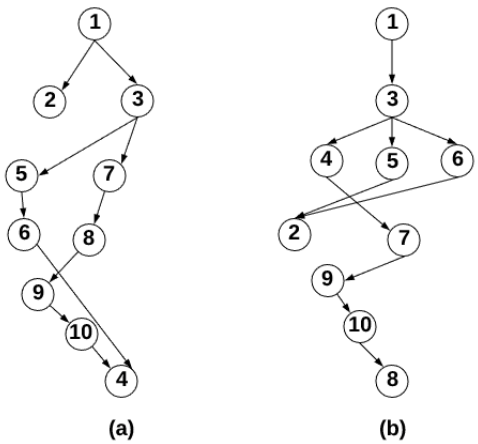

Figure 5: Primeira Aplicação (a) e Segunda Aplicação (b) - 10 Tarefas

O algoritmo genético (AG) [10] é usado para testar as to-

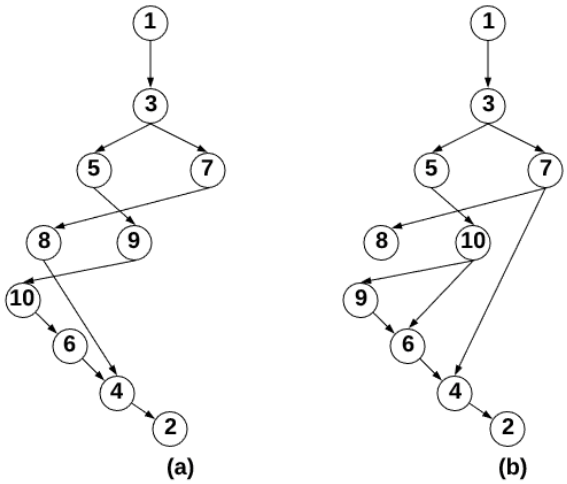

Figure 6: Terceira Aplicação (a) e Quarta Aplicação (b) - 10 Tarefas

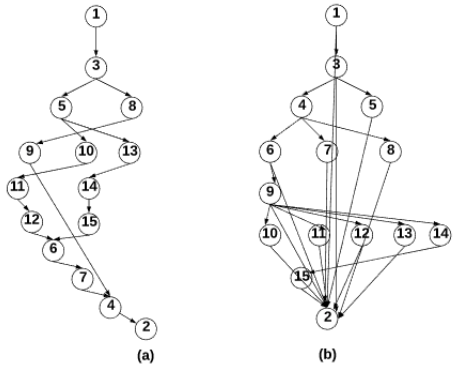

Figure 7: Primeira Aplicação (a) e Segunda Aplicação (b) - 15 Tarefas

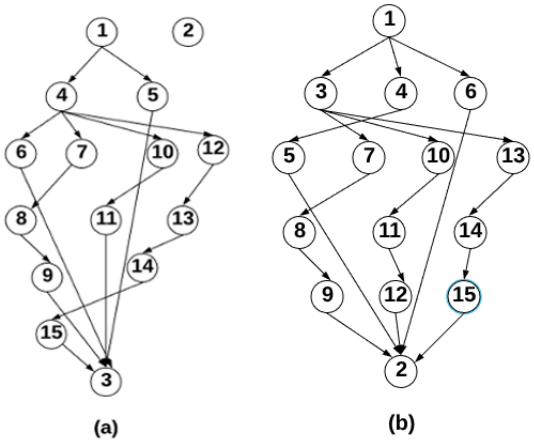

Figure 8: Terceira Aplicação (a) e Quarta Aplicação (b) - 15 Tarefas

pologias, reduzindo os números de roteadores para testar os limites do simulador. O simulador gera diferentes topologias para cada aplicação. Vale ressaltar que este artigo pretende apresentar um simulador para topologias irregulares, em vez de uma ferramenta de otimização de rede.

Cada aplicação envia diferentes quantidades de pacotes. A aplicação 1 envia 10000 pacotes por tarefa; a aplicação 2 envia 12000; a 3 enviam 5000 e a aplicação 4 envia 8000 por tarefa. Cada pacote tem tamanho de 5 flits e a taxa de injeção é de 3 nanossegundos.

As Figuras 9, 10, 11 e 12 mostram os gráficos para métricas de latência e prazo. As tabelas 2, 3, 4, 5, 6 e 7 mostram esses resultados com mais detalhes, incluindo os números de roteadores de cada tipo usado nos testes.

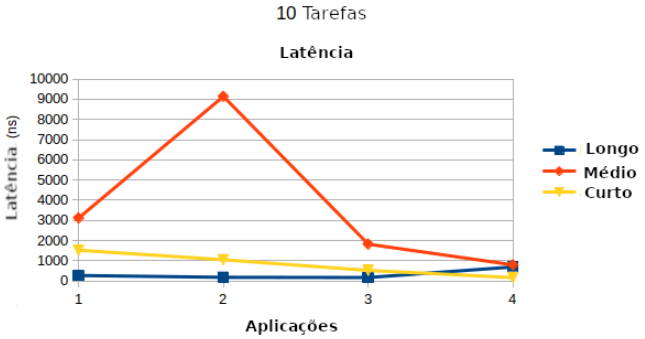

Figure 9: Resultados de Latência Para as Aplicações com 10 Tarefas

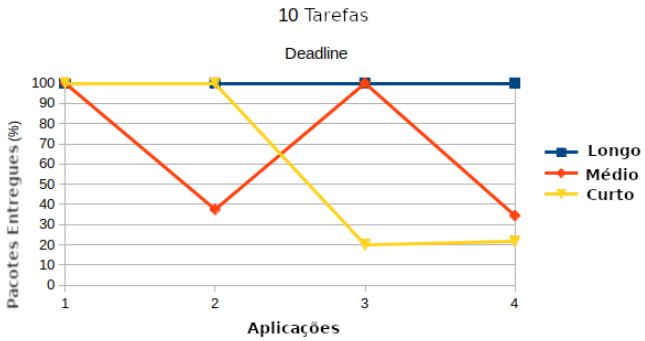

Figure 10: Resultados de Deadline Para as Aplicações com 10 Tarefas

Table 2: Topologias Irregulares Com 10 Tarefas e Deadline Longo

Aplicação	Latência	Pacotes RT Entregues	T1	T2
1	279,350	100%	5	1
2	188,101	100%	3	3
3	177,907	100%	3	3
4	700,154	100%	4	2

Os experimentos referentes as aplicações com 10 tarefas mostraram que o simulador desenvolvido aqui conseguiu em sua maior parte encontrar topologias que mantivessem a latência média em uma faixa menor que 2000 nanossegundos.

Table 3: Topologias Irregulares Com 10 Tarefas e Deadline Médio

Aplicação	Latência	Pacotes RT Entregues	T1	T2
1	3120	100%	6	2
2	9140	37,5%	0	10
3	1830	100%	3	4
4	802,789	34,42%	0	10

Table 4: Topologias Irregulares Com 10 Tarefas e Deadline Curto

Aplicação	Latência	Pacotes RT Entregues	T1	T2
1	1530	100%	6	2
2	1058	100%	3	3
3	530	20%	1	9
4	173,757	21,73%	10	0

Quanto a questão de latência pode ser observado que até com deadlines curtos, em algumas configurações de topologias encontradas foi possível entregar todos os pacotes cumprindo os prazos de deadline.

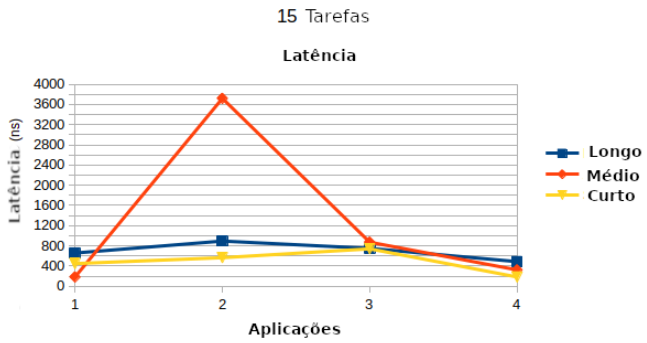

Figure 11: Resultados de Latência Para as Aplicações com 15 Tarefas

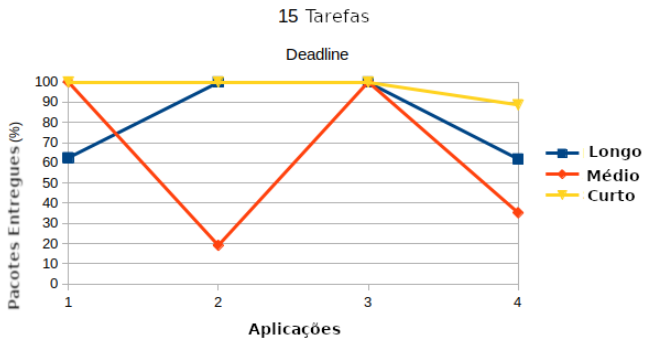

Figure 12: Resultados de Deadline Para as Aplicações com 10 Tarefas

Quando aumentado o tamanho das aplicações para trabalharem com 15 tarefas, pode-se observar que no quesito

Table 5: Topologias Irregulares Com 15 Tarefas e Deadline Longo

Aplicação	Latência	Pacotes RT Entregues	T1	T2
1	658,867	62,51%	0	10
2	895,676	100%	5	3
3	750,993	100%	3	5
4	488,399	61,90%	0	10

Table 6: Topologias Irregulares Com 15 Tarefas e Deadline Médio

Aplicação	Latência	Pacotes RT Entregues	T1	T2
1	184,467	100%	6	2
2	3720	19,14%	0	10
3	873,848	100%	2	6
4	324,339	35,29%	0	10

Table 7: Topologias Irregulares Com 15 Tarefas e Deadline Curto

Aplicação	Latência	Pacotes RT Entregues	T1	T2
1	450	100%	6	2
2	566,67	100%	6	2
3	741,765	100%	5	3
4	184,467	88,69%	0	10

latência, o IrNoC conseguiu manter a maioria dos resultados com uma taxa de latência menor que 800 nanossegundos. No quesito deadline pode ser observado que para as aplicações contendo um deadline mais curto, em sua maioria foi possível encontrar topologias que satisfizessem uma entrega de 100% dos pacotes dentro do prazo.

Também foram testadas aplicações com 10 tarefas reduzindo a taxa de injeção de pacotes. As figuras 13 e 14 mostram os resultados.

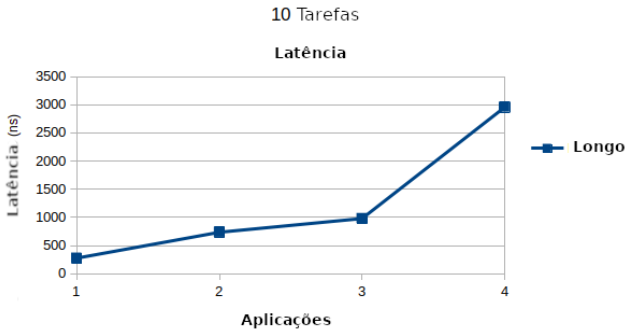

Figure 13: Resultados de Latência para Aplicações com 10 Tarefas e Taxa de Injeção Reduzida

Para os experimentos com redução de taxa de injeção, os testes foram realizados somente para deadlines longos. Pode-se perceber também que o IrNoC no quesito deadline

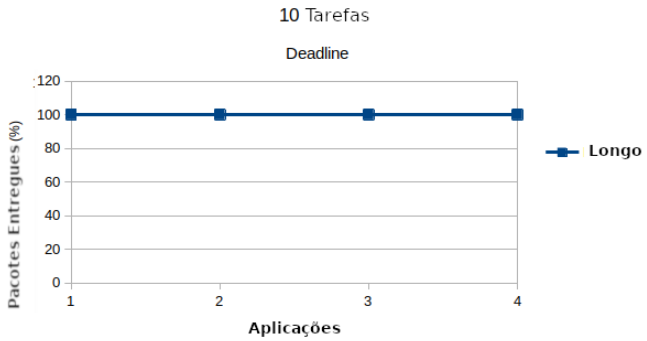

Figure 14: Resultados de Deadline para Aplicações com 10 Tarefas e Taxa de Injeção Reduzida

Table 8: Topologias Irregulares Com 10 Tarefas e Deadline Longo Com Taxa de Injeção Reduzida

Aplicação	Latência	Pacotes RT Entregues	T1	T2
1	279,350	100%	5	1
2	188,101	100%	3	3
3	177,907	100%	3	3
4	700,154	100%	4	2

consegue entregar todos os pacotes e a taxa de latência teve uma queda quando comparado ao mesmo experimento com uma taxa de injeção maior.

Nesta comparação entre os valores de deadline, pode-se analisar que a ferramenta IrNoC, mesmo com valores de deadline curtos, pode fornecer 100% em alguns casos de teste. Aumentando o número de roteadores, o IrNoC também conseguiu atender às demandas de 100% dos pacotes entregues em vários casos de teste.

Usando a ferramenta IrNoC, pode-se executar testes em diferentes tamanhos e configurações e obter os valores das métricas para latência e deadline. A preocupação desse artigo é mostrar os resultados e não a avaliação. A parte Avaliação está fora do escopo deste trabalho.

5. CONCLUSÃO E TRABALHOS FUTUROS

Topologias irregulares estão sendo cada vez mais usadas para projetar Sistemas-em-Chip. Obter uma ferramenta para esta tarefa é essencial para o projetista avaliar as métricas necessárias para o seu projeto.

Neste artigo, a ferramenta IrNoC é apresentada. O IrNoC pode atender a diferentes características de topologias irregulares, tamanho de rede variável, tamanho de deadline, tipos de roteadores e avaliar métricas de latência e pacotes entregues. A ferramenta é desenvolvida em SystemC, garantindo maior precisão em comparação com outras ferramentas.

No futuro, pretende-se implementar novos recursos na ferramenta, como o módulo WiNoC (rede-em-chip sem fio) [3], para descobrir o impacto que as WiNoCs teriam ao trabalhar com topologias irregulares.

Agradecimentos

Este estudo foi financiado em parte pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

6. REFERENCES

[1] L. Benini and G. De Micheli. Networks on chips: A new soc paradigm. *Computer-IEEE Computer Society-*, 35(EPFL-ARTICLE-165542):70–78, 2002.

[2] N. Choudhary, M. Gaur, and V. Laxmi. Irregular noc simulation framework: Irnirgam. In *2011 international conference on emerging trends in networks and computer communications (ETNCC)*, pages 1–5. IEEE, 2011.

[3] S. Deb, K. Chang, A. Ganguly, and P. Pande. Comparative performance evaluation of wireless and optical noc architectures. In *23rd IEEE International SOC Conference*, pages 487–492. IEEE, 2010.

[4] R. P. Dick, D. L. Rhodes, and W. Wolf. Tgff: task graphs for free. In *Proceedings of the Sixth International Workshop on Hardware/Software Codesign.(CODES/CASHE'98)*, pages 97–101. IEEE, 1998.

[5] H. Hossain, M. Ahmed, A. Al-Nayeem, T. Z. Islam, and M. M. Akbar. Gpnocsim-a general purpose simulator for network-on-chip. In *2007 International Conference on Information and Communication Technology*, pages 254–257. IEEE, 2007.

[6] S. Hougardy. The floyd–warshall algorithm on graphs with negative cycles. *Information Processing Letters*, 110(8-9):279–281, 2010.

[7] P. K. Janert. *Gnuplot in action: understanding data with graphs*. Manning, 2010.

[8] A. Jantsch, H. Tenhunen, et al. *Networks on chip*, volume 396. Springer, 2003.

[9] W. Müller, W. Rosenstiel, and J. Ruf. *SystemC: methodologies and applications*. Springer Science & Business, 2007.

[10] M. A. C. Pacheco et al. Algoritmos genéticos: princípios e aplicações. *ICA: Laboratório de Inteligência Computacional Aplicada. Departamento de Engenharia Elétrica. Pontifícia Universidade Católica do Rio de Janeiro. Fonte desconhecida*, page 28, 1999.

[11] P. R. Panda. Systemc-a modeling platform supporting multiple design abstractions. In *International Symposium on System Synthesis (IEEE Cat. No. 01EX526)*, pages 75–80. IEEE, 2001.

[12] S. Tosun, Y. Ar, and S. Ozdemir. Application-specific topology generation algorithms for network-on-chip design. *IET computers & digital techniques*, 6(5):318–333, 2012.

[13] K. Vyas, N. Choudhary, and D. Singh. Nc-g-sim: A parameterized generic simulator for 2d-mesh, 3d-mesh & irregular on-chip networks with table-based routing. *Global Journal of Computer Science and Technology*, 2013.