



MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21d/2023/08.25.12.04-NTC

AVALIAÇÃO COMPUTACIONAL DE NÚCLEOS DINÂMICOS CANDIDATOS PARA O MONAN

Luiz Flávio Rodrigues
Carlos Renato de Souza
Denis Magalhães de Almeida Eiras
Eduardo Georges Khamis
Eduardo Lúcio Mendes Garcia
João Messias Alves da Silva
Kleucio Claudio
Roberto Pinto Souto

Documento técnico de avaliação
computacional de núcleos dinâmi-
cos candidatos para o MONAN.

URL do documento original:

<<http://urlib.net/8JMKD3MGP3W34T/49MKMSS>>

INPE
São José dos Campos
2023

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Coordenação de Ensino, Pesquisa e Extensão (COEPE)
Divisão de Biblioteca (DIBIB)
CEP 12.227-010
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/7348
E-mail: pubtc@inpe.br

CONSELHO DE EDITORAÇÃO E PRESERVAÇÃO DA PRODUÇÃO INTELLECTUAL DO INPE - CEPPII (PORTARIA Nº 176/2018/SEI-INPE):

Presidente:

Dra. Marley Cavalcante de Lima Moscati - Coordenação-Geral de Ciências da Terra (CGCT)

Membros:

Dra. Ieda Del Arco Sanches - Conselho de Pós-Graduação (CPG)
Dr. Evandro Marconi Rocco - Coordenação-Geral de Engenharia, Tecnologia e Ciência Espaciais (CGCE)
Dr. Rafael Duarte Coelho dos Santos - Coordenação-Geral de Infraestrutura e Pesquisas Aplicadas (CGIP)
Simone Angélica Del Ducca Barbedo - Divisão de Biblioteca (DIBIB)

BIBLIOTECA DIGITAL:

Dr. Gerald Jean Francis Banon
Clayton Martins Pereira - Divisão de Biblioteca (DIBIB)

REVISÃO E NORMALIZAÇÃO DOCUMENTÁRIA:

Simone Angélica Del Ducca Barbedo - Divisão de Biblioteca (DIBIB)
André Luis Dias Fernandes - Divisão de Biblioteca (DIBIB)

EDITORAÇÃO ELETRÔNICA:

Ivone Martins - Divisão de Biblioteca (DIBIB)
André Luis Dias Fernandes - Divisão de Biblioteca (DIBIB)



MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21d/2023/08.25.12.04-NTC

AVALIAÇÃO COMPUTACIONAL DE NÚCLEOS DINÂMICOS CANDIDATOS PARA O MONAN

Luiz Flávio Rodrigues
Carlos Renato de Souza
Denis Magalhães de Almeida Eiras
Eduardo Georges Khamis
Eduardo Lúcio Mendes Garcia
João Messias Alves da Silva
Kleucio Claudio
Roberto Pinto Souto

Documento técnico de avaliação
computacional de núcleos dinâmi-
cos candidatos para o MONAN.

URL do documento original:

<<http://urlib.net/8JMKD3MGP3W34T/49MKMSS>>

INPE
São José dos Campos
2023



Esta obra foi licenciada sob uma Licença Creative Commons Atribuição-NãoComercial 3.0 Não Adaptada.

This work is licensed under a Creative Commons Attribution-NonCommercial 3.0 Unported License.

AGRADECIMENTOS

Ao apoio imprescindível do Dr. Roberto Pinto Souto do LNCC nos testes com os núcleos dinâmicos.

RESUMO

Documento técnico de avaliação computacional dos núcleos dinâmicos candidatos para o MONAN (Model for Ocean-land-Atmosphere prediction). Foram analisados três núcleos dinâmicos, GEF (Global Eta Framework), FV3/SHIELD (Finite-Volume Cubed-Sphere Dynamical Core / System for High-resolution prediction on Earth-to-Local Domains) e o MPAS (Model for Prediction Across Scales). As métricas de avaliação foram a Manutenibilidade, Portabilidade, Performance, Escalabilidade e Eficiência.

Palavras-chave: Avaliação computacional. Qualidade de Software. Processamento de Alto Desempenho. Previsão Numérica do Tempo.

COMPUTATIONAL EVALUATION OF CANDIDATE DYNAMIC CORES FOR MONAN

ABSTRACT

Technical document of computational evaluation of candidate dynamic cores for MONAN (Model for Ocean-land-Atmosphere prediction). Three dynamic cores were analyzed, GEF (Global Eta Framework), FV3/SHiELD (Finite-Volume Cubed-Sphere Dynamical Core / System for High-resolution prediction on Earth-to-Local Domains) and MPAS (Model for Prediction Across Scales). The evaluation metrics were Maintainability, Portability, Performance, Scalability and Efficiency.

Keywords: Computational evaluation. Software quality. High Performance Computing. Numerical Weather Prediction.

LISTA DE FIGURAS

	<u>Pág.</u>
1.1 - Ilustração da grade não estruturada de Voronoi do MPAS.	5
1.2 - Exemplo de regionalização com grade de Voronoi - Refinamento nos Andes.	5
2.1 - Esferas cubadas - Grade GEF.	10
2.2 - Aplicações do modelo SHIELD.	12
2.3 - Ilustração da grade não estruturada do FV3.	12
3.1 - Métricas das sub características da Manutenibilidade com os modelos FV3, GEF e MPAS.	24
3.2 - Métricas das sub características da Manutenibilidade dos modelos considerados satisfatórios.	26
3.3 - Tempo de processamento paralelo (MPI) e speedup (Ganho de desempenho) do núcleo da dinâmica do MPAS no cluster Minerva.	31
3.4 - Eficiência do processamento paralelo (MPI) do núcleo da dinâmica do MPAS no cluster Minerva.	32
3.5 - Tempo de processamento paralelo (MPI) e Speedup do núcleo da dinâmica do MPAS no cluster Rattler.	33
3.6 - Eficiência do processamento paralelo (MPI) do núcleo da dinâmica do MPAS no cluster Rattler.	34
3.7 - Performance e speedup medidos até 7 nós e teóricos (extrapolados) acima de 7 nós.	35
3.8 - Desempenho e speedup do core dinâmico do modelo FV3 na Minerva.	37
3.9 - Eficiência do core dinâmico do modelo FV3 na Minerva.	38
4.1 - Comparação da performance entre os modelos.	42

LISTA DE TABELAS

	<u>Pág.</u>
2.1 - Configurações utilizadas nos experimentos - MPAS.	7
2.2 - Dados do modelo GEF.	11
2.3 - Configurações utilizadas nos experimentos - SHIELD.	13
2.4 - Critérios de pontuação de acordo com as métricas de avaliação de portabilidade.	19
3.1 - Análise das métricas e pontuação dos modelos FV3, GEF e MPAS.	21
3.2 - Análise das métricas e pontuação dos modelos FV3, GEF e MPAS sem os critérios relativos a documentação, tamanho médio dos módulos, “Implicit”, “Fan-in” e “Fan-out”.	23
3.3 - Análise do conjunto total de métricas e pontuação dos modelos considerados satisfatórios.	25
3.4 - Pontuação do modelo MPAS de acordo com as métricas de avaliação de portabilidade.	27
3.5 - Pontuação do FV3 de acordo com as métricas de avaliação de portabilidade.	28
3.6 - Tempos de execução retornados pela saída do modelo MPAS.	29
3.7 - Especificação dos tempos da rodada FV3.	36
C.1 - Valores de referência da Complexidade Ciclomática de McCabe.	51
C.2 - Métricas utilizadas na avaliação da Manutenibilidade.	53

LISTA DE SIGLAS E ABREVIATURAS

COSIM	Los Alamos National Laboratory
CPTEC	Centro de Previsão de Tempo e Estudos Climáticos
DIMNT	Divisão de Modelagem Numérica
ESMF	Earth System Modeling Framework
FV3	Finite-Volume Cubed-Sphere Dynamical Core
GCC	Grupo de Computação Científica
GEF	Global Eta Framework
GFDL	Geophysical Fluid Dynamics Laboratory
GFS	Global Forecast System
GSFC	Goddard Space Flight Center
HDF5	Hierarchical Data Format
INPE	Instituto Nacional de Pesquisas Espaciais
ISO/IEC	International Organization for Standardization / International Electrotechnical Commission
LNCC	Laboratório Nacional de Computação Científica
MONANN	Model for Ocean-IaNd-Atmosphere predictionN
MPAS	Model for Prediction Across Scales
MPI	Message Passing Interface
NCAR	National Center for Atmospheric Research
ND	Núcleo Dinâmico
NEMSIO	NOAA Environmental Modeling System
NetCDF	Network Common Data Form
NOAA	National Oceanic and Atmospheric Administration
OpenACC	Open Accelerators
OpenMP	Open Multi-Processing
PEE	Performance, Escalabilidade e Eficiência
QS	Qualidade de Software
SCVT	Spherical Centroidal Voronoi Tessellation
SHIELD	System for High-resolution prediction on Earth-to-Local Domains
SID	Serviço de Informação e Documentação
UFS	Unified Forecast System

SUMÁRIO

	<u>Pág.</u>
1 INTRODUÇÃO	1
2 METODOLOGIA	3
2.1 Modelos e Núcleos Dinâmicos	4
2.1.1 MPAS	4
2.1.1.1 Código e Artefatos	6
2.1.1.2 Configurações utilizadas nos experimentos	6
2.1.2 GEF	9
2.1.2.1 Código e Artefatos	10
2.1.2.2 Configurações utilizadas nos experimentos	10
2.1.3 FV3	11
2.1.3.1 Código e Artefatos	13
2.1.3.2 Configurações utilizadas nos experimentos	13
2.2 Recursos Computacionais	14
2.3 Avaliação da Qualidade de Software	15
2.3.1 Manutenibilidade	15
2.3.2 Portabilidade	16
2.3.3 Performance, Escalabilidade e Eficiência (PEE)	19
3 RESULTADOS	21
3.1 Manutenibilidade	21
3.2 Portabilidade	26
3.2.1 MPAS	26
3.2.2 FV3	27
3.3 Performance, Escalabilidade e Eficiência (PEE)	28
3.3.1 MPAS	29
3.3.2 FV3	35
4 CONCLUSÃO E DISCUSSÕES	38
4.1 Manutenibilidade	38
4.2 Portabilidade	39
4.3 Performance, Escalabilidade e Eficiência (PEE)	40
4.4 Discussão e Análise	42
4.5 Trabalhos futuros	44
5 REFERÊNCIAS BIBLIOGRÁFICAS	45
APÊNDICE A – CARACTERÍSTICAS DA QUALIDADE DE SOFTWARE	47
APÊNDICE B – CRITÉRIOS DE PONTUAÇÃO DE SUBCARACTERÍSTICAS E CARACTERÍSTICAS	50

APÊNDICE C – AVALIAÇÃO DE CARACTERÍSTICAS DA MANUTENIBILIDADE	51
C.1 Introdução	51
C.1.1 Métrica de Complexidade Ciclomática de McCabe	51
C.1.2 Métricas de Software da RADC	52
C.2 Metodologia	52
C.2.1 Métricas	52
C.3 Ferramentas para avaliação da Manutenibilidade	55
C.4 Geração das métricas	56

1 INTRODUÇÃO

Prever a condição climática em escalas de horas, dias e meses à frente é um dos maiores desafios técnico-científicos enfrentados pela humanidade. No entanto, este desafio tem sido vencido por diversos países e tem se demonstrado a importância de se ter informações confiáveis em tempo real das condições atmosféricas e oceânicas presentes e futuras. A importância se qualifica principalmente pela preservação da vida de pessoas e mitigação de prejuízos socioeconômicos advindos do uso destas informações. O Brasil tem um relevante histórico de desenvolvimento e aplicação de modelagem para a previsão de tempo e clima.

Porém, os esforços estão fragmentados e dispersos em vários grupos e modelos que não interagem entre si. Como resultado, as previsões hoje produzidas no país não utilizam em sua plenitude o atual conhecimento científico considerado como estado-da-arte em modelagem numérica, métodos e dados e, assim, há um grande espaço para melhorar a competitividade em relação aos produtos de previsão gerados por outras instituições internacionais. Nesse contexto, há um potencial enorme do país se colocar em um patamar superior com a focalização e unificação dos esforços e expertises através da adoção de um sistema unificado de modelagem do Sistema Terrestre que seja comunitário e que atenda todas as escalas (temporais e espaciais) de fenômenos de relevância para a sociedade brasileira.

Para tanto foi desenvolvido um projeto que tem por objetivo principal desenvolver tal sistema de modelagem ancorado em um robusto sistema de assimilação de dados e aperfeiçoado com técnicas de inteligência artificial. Com tal sistema, o Brasil alcançará o estado-da-arte em previsão da atmosfera e oceanos para o benefício da sociedade brasileira e da América do Sul em geral.

Esse projeto ancorado no Termo de Abertura de Projeto, SEI 01340.005344/2021-50, tem como objeto o Desenvolvimento comunitário de um modelo numérico do Sistema Terrestre adaptado para as condições tropicais e subtropicais da América do Sul e de suas aplicações para previsão

de tempo, clima e ambiente em escalas espaço-temporais relevantes para a sociedade brasileira.

No desenvolvimento do projeto alinham-se diversos órgãos da ciência, da educação e da meteorologia brasileira formando um esforço comunitário capitaneado pelo INPE. Esse esforço é gerenciado por um comitê científico que tem como função precípua planejar e gerenciar o desenvolvimento do modelo comunitário.

O modelo numérico oriundo do projeto foi denominado **MONAN**, acrônimo de ***Model for Ocean-land-Atmosphere prediction*** e tem em sua primeira fase a avaliação para escolha de um Núcleo Dinâmico (ND) onde serão acoplados os códigos computacionais contendo as físicas adequadas às características específicas do modelo.

A avaliação computacional dos NDs ficou a cargo do Grupo de Computação Científica (GCC), parte da Divisão de Modelagem Numérica (DIMNT) da Coordenação Geral de Ciências da Terra (CGCT) que é uma das coordenações do Instituto Nacional de Pesquisas Espaciais (INPE). A tarefa tem o auxílio substancial do Laboratório Nacional de Computação Científica (LNCC).

Os NDs avaliados e objetos deste relatório foram escolhidos por estarem no estado da arte em termos computacionais e já terem passado por processo de avaliação em passado recente pela NOAA - *National Oceanic and Atmospheric Administration*, órgão de pesquisa e operacional norte americano e por alguns deles estarem operacionais em diversos centros. Os testes de escolha realizados pela NOAA se deram entre os anos de 2015 e 2017 e culminaram na escolha de um ND que foi posteriormente usado no modelo global da instituição (AVEC, 2023).

Há que se ressaltar que os NDs continuaram a receber atualizações após os testes realizados pela NOAA e portanto estão modernizados e com novas capacidades exibindo assim novas performances computacionais e capazes de utilizar arquiteturas computacionais diversas. Optou-se então para que fossem realizados outros testes com essas novas versões disponibilizadas de forma a

se escolher aquele que melhor se adapta às necessidades do modelo MONAN. Tal processo é mostrado nos próximos capítulos deste relatório.

2 METODOLOGIA

Modelos numéricos são softwares complexos. Os softwares tratam de duas classes de requisitos: requisitos funcionais e requisitos não funcionais.

- a) **Requisitos funcionais** são aqueles diretamente ligados aos interesses dos usuários na funcionalidade do software, isto é, a produção de resultados adequados que atendem minimamente a função para qual o software foi especificado. No caso de modelos de previsão de tempo, caso específico do software em avaliação, trata-se do acerto das previsões quando comparados com dados observados. Esse documento não trata desses requisitos e avalia o segundo grupo de requisitos, os requisitos não funcionais.
- b) **Requisitos não funcionais** são totalmente ligados à qualidade do software. O desenvolvimento de um software tem ciclos contínuos de requisitos-desenvolvimento-testes-operação que permitem que a vida útil do software seja longa e possa sobreviver por décadas. O ciclo de vida do software se perpetua por gerações de desenvolvedores, os quais devem garantir a qualidade para que novas gerações possam dar continuidade ao desenvolvimento. Para isso torna-se fundamental o uso de técnicas e processos que facilitem o atendimento aos requisitos não funcionais.

Para a avaliação do núcleo dinâmico do MONAN levou em consideração alguns aspectos da Qualidade de Software (QS). Segundo a norma ISO/IEC 9126, revisada pela norma ISO/IEC 25010:2011, quando a QS se refere ao produto, estas fornecem uma estrutura para especificar características de QS e realizar comparações entre produtos de software. Outras características e métricas de QS encontradas em livros de engenharia de software também foram introduzidas. Estas normas, características e métricas serão utilizadas

como base para a construção dos critérios de escolha do núcleo dinâmico. As características de QS avaliadas estão detalhadas no Apêndice A.

Dentro do contexto de qualidade interna, onde o software é avaliado para ser reutilizado e modificado (ciclo de vida contínua), a definição das características e subcaracterísticas devem ser levantadas em função da área de aplicação do produto de software. Esta definição deve ser feita antes do início do desenvolvimento do mesmo. Produtos de maior porte devem ser subdivididos em módulos e cada um destes deve ter seus próprios conjuntos de características e sub características. Portanto, um Núcleo Dinâmico de qualidade deve ser utilizado considerando alguns aspectos principais da QS que são inerentes ao Produto.

Nas seções 3 e 4 são apresentados, respectivamente, os resultados das características de Manutenibilidade, Portabilidade e Eficiência, que estão relacionadas com a avaliação computacional dos NDs.

2.1 Modelos e Núcleos Dinâmicos

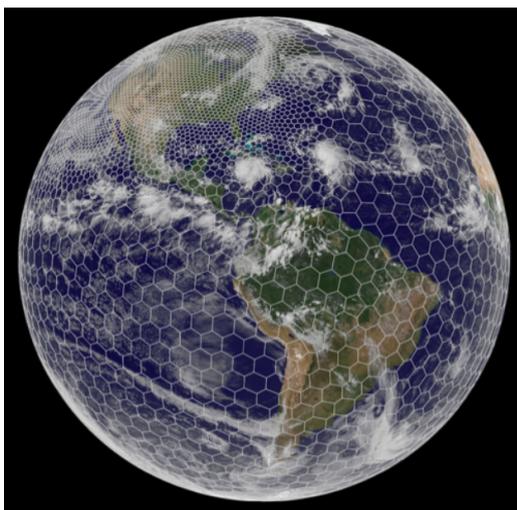
2.1.1 MPAS

O *Model for Prediction Across Scales* (MPAS, 2023) é um projeto de desenvolvimento de um modelo acoplado que abrange vários modelos componentes do sistema terrestre, como por exemplo, modelos atmosféricos e oceânicos. Desenvolvido pelo grupo de modelagem climática do *Los Alamos National Laboratory* (COSIM) e pelo *National Center for Atmospheric Research* (NCAR) com o objetivo de criar um modelo acoplado capaz de realizar desde previsões de tempo regionais até simulações climáticas do sistema terrestre de alto nível.

O modelo usa a grade de *Voronoi*, formalmente chamada de *Spherical Centroidal Voronoi Tessellation* (SCVT), que é uma malha hexagonal semelhante a um favo de mel que pode ser estendido em algumas regiões e comprimido para maior resolução em outras, como mostra a Figura 1.1 e a Figura 1.2. Isso permite que os pesquisadores executem um modelo global em escalas de resolução de tempestades (alta-resolução) onde ele mais importa e resolução mais baixa em todos os demais lugares para conservar recursos

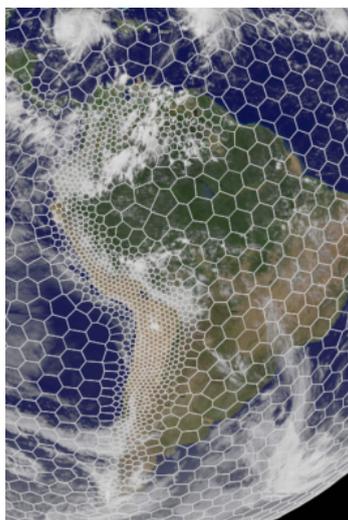
computacionais. O NCAR lidera o desenvolvimento do componente atmosférico do MPAS enquanto que outros parceiros da comunidade de desenvolvimento do MPAS trabalham na parte do oceano, gelo terrestre e componentes de gelo marinho.

Figura 1.1 - Ilustração da grade não estruturada de Voronoi do MPAS.



Fonte: MPAS (2023)

Figura 1.2 - Exemplo de regionalização com grade de Voronoi - Refinamento nos Andes.



Fonte: MPAS (2023)

2.1.1.1 Código e Artefatos

O pacote completo utilizado para a Previsão Global de Tempo, em sua resolução de 15 km foi obtido a partir de:

- I. GitHub MPAS-Model: <https://github.com/MPAS-Dev/MPAS-Model.git>
- II. NCAR/UCAR MPAS-Benchmark:
https://www2.mmm.ucar.edu/projects/mpas/benchmark/v6.x/MPAS-A_benchmark_15km_L56.tar.gz
- III. MPAS Home: <https://mpas-dev.github.io/>

No primeiro link acima encontra-se o repositório oficial do modelo MPAS, de onde podem ser baixadas todas as versões dos códigos do modelo. Para esta avaliação foi utilizada a versão 6.3.

No segundo link contém um pacote de benchmark, onde se encontram todos os arquivos e dados necessários para que uma rodada exemplo possa ser executada. Para esta avaliação se executou o modelo MPAS na resolução de 15km com 56 níveis verticais.

No terceiro link se encontra todo o material referencial do modelo MPAS, documentação detalhada, manual do usuário, publicações importantes, dentre outros.

2.1.1.2 Configurações utilizadas nos experimentos

Os parâmetros de configuração do modelo MPAS são definidos no arquivo “*namelist.atmosphere*”. Na Tabela 2.1 são mostrados os valores utilizados.

Nos parâmetros listados na Tabela 2.1, destacam-se as linhas em azul que definem a data e hora de início da simulação (*config_start_time*) e o tempo de duração das simulações (*config_run_duration*), que para esta avaliação foi de 12 horas de duração.

Tabela 2.1 - Configurações utilizadas nos experimentos - MPAS.

Parâmetro	Valor
nhyd_model	
config_time_integration_order	2
config_dt	90
config_start_time	'2010-10-23_00:00:00'
config_run_duration	'0_12:00:00'
config_split_dynamics_transport	TRUE
config_number_of_sub_steps	2
config_dynamics_split_steps	3
config_h_mom_eddy_visc2	0
config_h_mom_eddy_visc4	0
config_v_mom_eddy_visc2	0
config_h_theta_eddy_visc2	0
config_h_theta_eddy_visc4	0
config_v_theta_eddy_visc2	0
config_horiz_mixing	'2d_smagorinsky'
config_len_disp	15000
config_visc4_2dsmag	0.05
config_w_adv_order	3
config_theta_adv_order	3
config_scalar_adv_order	3
config_u_vadv_order	3
config_w_vadv_order	3
config_theta_vadv_order	3
config_scalar_vadv_order	3

(Continua)

Parâmetro	Valor
config_scalar_advection	TRUE
config_positive_definite	FALSE
config_monotonic	TRUE
config_coef_3rd_order	0.25
config_epssm	0.1
config_smdiv	0.1
damping	
config_zd	22000
config_xnutr	0.2
io	
config_pio_num_iotasks	0
config_pio_stride	1
decomposition	
config_block_decomp_file_prefix	'x1.2621442.graph.info.part.'
restart	
config_do_restart	FALSE
printout	
config_print_global_minmax_vel	TRUE
config_print_detailed_minmax_vel	FALSE
IAU	
config_IAU_option	'off'
config_IAU_window_length_s	21600
physics	
config_sst_update	FALSE

(Continua)

Parâmetro	Valor
config_sstdiurn_update	FALSE
config_deepsoiltemp_update	FALSE
config_radtlw_interval	'00:15:00'
config_radtsw_interval	'00:15:00'
config_bucket_update	'none'
config_physics_suite	'mesoscale_reference'
soundings	
config_sounding_interval	'none'

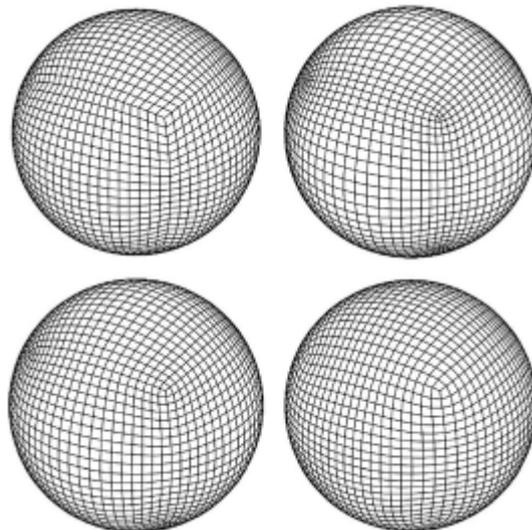
Fonte: Produção dos autores.

2.1.2 GEF

Global Eta Framework (GEF) é um modelo atmosférico global desenvolvido em coordenadas lineares (esfera cubada) e capaz de funcionar em grades esféricas retangulares arbitrárias, usando *stepwise* ("Eta") para a representação do terreno (Latinović et al. 2017), a Figura 2.1 ilustra esta grade. Isso permite que o modelo possa rodar em várias grades esféricas retangulares.

O GEF usa a globalização do modelo regional Eta (Mesinger et al. 2012) que tem como uma das grandes vantagens a formulação *stepwise* do terreno que reduz o erro do gradiente de pressão do sistema sigma na presença de montanhas íngremes. Essa capacidade é particularmente útil para a topografia típica da América do Sul com a cordilheira de montanhas dos Andes.

Figura 2.1 - Esferas cubadas - Grade GEF.



Fonte: (a) Painel superior esquerdo: gnomonic (Sadourny, 1972); (b) Painel superior direito: conformal (Ranciř c et al., ´ 1996); (c) Painel inferior esquerdo: smooth (SM) (Purser and Ranciř c, 1998); (d) Painel inferior direito: UJ (Ranciř c et al., 2017).

2.1.2.1 C3digo e Artefatos

O modelo GEF foi adquirido diretamente do site de FTP do CPTEC no endereęo abaixo:

I. FTP GRPETA:

http://ftp1.cptec.inpe.br/pesquisa/grpeta/tempo/GEF_Shapiro_filter/GEF_v1.0.0_Shapiro_Filter.tar.gz

2.1.2.2 Configuraę3es utilizadas nos experimentos

O modelo GEF n3o foi executado mas foi adquirido com as configuraę3es apresentadas na Tabela 2.2.

Tabela 2.2 - Dados do modelo GEF.

Variável	Descrição	Valor
IMO	Dimensão de cada face	401
LM	Número de níveis verticais	50
NMIN	Número de faces da esfera	6
NSUB	Número de subdivisões por face	10
LSM	Número de níveis na saída	31
DT	Delta T	40
OUTPUTNUM	Número de outputs na saída	120
NDAYS	Dias de integração	30
SSTC	Fonte do SST (1=NCEP)	1
IDATIN	Data de inicialização	02/09/2019
Resolução	Resolução do modelo	25 km

Fonte: Produção dos autores.

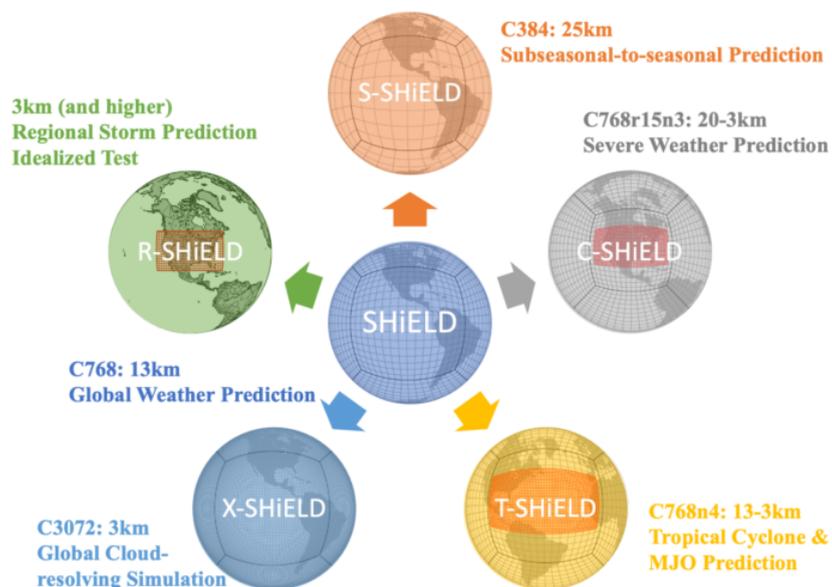
2.1.3 FV3

O SHIELD (Cheng et al. 2022; Harris et al. 2020) é um protótipo de modelo de atmosfera do *Unified Forecast System* (UFS) com capacidade de um sistema de previsão unificado em uma variedade de escalas de tempo e espaço, projetado para uma ampla gama de aplicações. Seu ND de esfera cúbica de volume finito (*Finite-Volume Cubed-Sphere Dynamical Core* - FV3) é importante especialmente para sua dinâmica não hidrostática flexível, recursos de resolução variável, física integrada e sua estrutura do sistema de modelagem flexível (FMS). Na Figura 2.2 observa-se suas diversas formas de aplicação.

O FV3 origina-se do FV-Core (*Finite-Volume dynamical core*) desenvolvido na década de 90 pelo GSFC (*Goddard Space Flight Center*) da NASA. O FV3 é, atualmente, o ND utilizado no modelo acoplado GFS (*Global Forecast System*)

do GFDL (*Geophysical Fluid Dynamics Laboratory*) capaz de suportar simulações atmosféricas hidrostática e não-hidrostática (GFDL, 2022a).

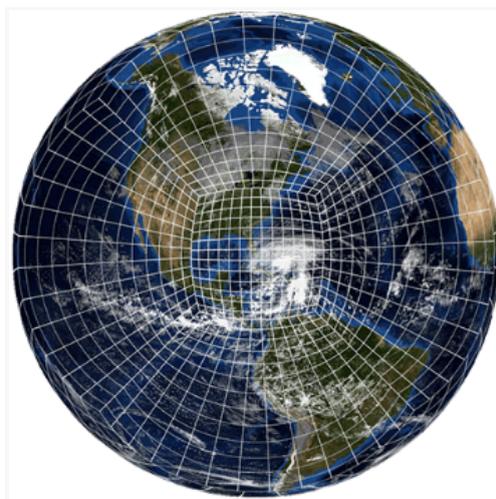
Figura 2.2 - Aplicações do modelo SHIELD.



Fonte: GFDL (2022a)

Como pode ser visto na Figura 2.3, o FV3 utiliza uma grade cúbica esférica flexível, que permite variar a resolução espacial em locais específicos de interesse do globo terrestre, enquanto que uma resolução mais grossa pode ser utilizada concorrentemente no restante do globo.

Figura 2.3 - Ilustração da grade não estruturada do FV3.



Fonte: GFDL (2022a)

2.1.3.1 Código e Artefatos

O pacote completo utilizado para a Previsão Global de Tempo, em sua resolução de 13km (C768), foi obtido a partir de:

- I. https://zenodo.org/record/5090895/files/shield_container.tar
- II. https://zenodo.org/record/5090124/files/GFS_fix.zip
- III. https://zenodo.org/record/5090124/files/global_scalability.zip

O primeiro link contém um pacote com todo o código do SHIELD e FV3. O segundo link contém os arquivos fixos de condição de contorno do modelo atmosférico global GFS utilizado em diversos experimentos. O terceiro link contém dados do experimento SHIELD para previsão de tempo Global.

2.1.3.2 Configurações utilizadas nos experimentos

Os principais parâmetros, fixos ou variáveis, usados nos experimentos deste relatório, são configurados no arquivo input.nml (GFDL, 2022b) e estão descritos na Tabela 2.3 abaixo.

Tabela 2.3 - Configurações utilizadas nos experimentos - SHIELD.

Parâmetro	Descrição	Valor
layout	Layout de processadores em cada bloco. O número de processadores atribuídos a um domínio deve ser igual a $layout(1)*layout(2)*tiles$	Variado entre (3,4), (4,6), (6,8),(8,12),(12,16) e (16,24) para atingir o número de processadores desejado nos experimentos
npx	Número de cantos da grade na direção x em um bloco do domínio	769
npz	Número de cantos da grade na direção y em um bloco do domínio. Este valor deve ser idêntico ao npx em uma malha de esfera cúbica	769
npz	Número de níveis verticais. Cada escolha de npz vem com um conjunto predefinido de níveis de pressão sigma híbridos e topo do modelo	63

(Continua)

Parâmetro	Descrição	Valor
n tiles	Número de blocos no domínio. Para a esfera cubada, deve ser 6, um bloco para cada face da esfera cubada; para a maioria dos outros domínios (incluindo grades aninhadas), normalmente deve ser definido como 1	6
hours	Tempo total de integração. Nos experimentos, o tempo total de integração foi extrapolado para 24h	12

Fonte: Produção dos autores.

2.2 Recursos Computacionais

Os recursos computacionais disponíveis para a realização do estudo de Portabilidade e Performance, Escalabilidade e Eficiência dos modelos (PEE), dos modelos MPAS e SHIELD (FV3) foram:

a) Cluster Egeon (CPTEC)

- 33 nós
- 2 CPU AMD EPYC 7H12 64-Core (128 núcleos), por nó
- 503 GBytes de RAM (total)

b) Cluster Minerva (Dell)

- 64 nós
- 2 CPU AMD EPYC 7713 64-Core (128 núcleos), por nó
- 251 GBytes de RAM (total)

c) Cluster Rattler (Dell)

- 7 nós com GPU
- 2 CPU AMD EPYC 7543 32-Core (64 núcleos), por nó
- 503 GBytes de RAM (total)
- 4 GPU NVIDIA A100-SXM4, por nó
- 40 GBytes de RAM (cada GPU)

As máquinas da DELL foram gentilmente cedidas para os experimentos. Apenas o Cluster Egeon esteve disponível em todo o tempo necessário. As demais máquinas foram cedidas com tempos parciais para uso. O uso das

máquinas permitiu os experimentos nas resoluções espaciais selecionadas. Entretanto, devido a limitações nos recursos, não foi possível realizar experimentos mais longos, rodadas envolvendo clima sazonal ou mudanças climáticas. Além disso, é necessário informar que as rodadas de previsão de tempo foram limitadas a 12h, posteriormente extrapoladas para 24h. Contudo, para analisar o proposto, isto é, o ND, tal tempo de rodada foi suficiente.

2.3 Avaliação da Qualidade de Software

A Qualidade de Software dos NDs foram analisadas em três etapas:

Na primeira etapa, o código foi analisado quanto a Característica de Manutenibilidade, ou seja, foram analisados a facilidade de manutenção e os possíveis impactos (seção 2.3.1)

A segunda etapa consistiu em avaliar a portabilidade dos modelos, onde os modelos foram devidamente instalados nos ambientes computacionais, juntamente com todas as bibliotecas, pacotes e dados necessários para sua compilação e execução (seção 2.3.2).

Na terceira etapa da avaliação foram feitos testes de PEE, onde os modelos foram executados utilizando paralelismo computacional, variando-se a quantidade de cores computacionais. O tempo de execução de cada rodada foi medido e analisado (seção 2.3.3).

2.3.1 Manutenibilidade

A Característica de Manutenibilidade diz respeito à facilidade de modificação e impactos sobre modificações. Esta seção exhibe os resultados da avaliação da característica de Manutenibilidade dos NDs, cuja metodologia está descrita no Apêndice A.

Manutenibilidade: identifica a capacidade (ou facilidade) do produto de software ser modificado, incluindo tanto as melhorias ou extensões de funcionalidade quanto às correções de defeitos, falhas ou erros. Suas sub características são:

- a) **Analisabilidade:** representa a facilidade em se diagnosticar eventuais problemas e identificar as causas das deficiências ou falhas.

- b) Modificabilidade: caracteriza a facilidade com que o comportamento do software pode ser modificado.
- c) Reusabilidade: identifica a capacidade de se reutilizar o software por completo ou partes dele.
- d) Estabilidade: avalia a capacidade do software de evitar efeitos colaterais decorrentes de modificações introduzidas.
- e) Testabilidade: representa a capacidade de se testar o sistema modificado, tanto quanto as novas funcionalidades quanto as não afetadas diretamente pela modificação.

2.3.2 Portabilidade

A portabilidade deve ser avaliada em referência a 5 requisitos básicos:

a) Arquitetura de máquinas

O modelo deve ser capaz de compilar no maior número de arquiteturas de máquinas distintas. Não deve apresentar disfunções ou perda de funcionalidade.

O modelo deve apresentar bons resultados nas diferentes arquiteturas, seja na funcionalidade quanto em uma performance mínima.

b) Processadores

O modelo deve mostrar comportamento estável em diferentes tipos de processadores e não apresentar performance inesperada para esses processadores.

c) Pacotes de software necessários para seu uso

O modelo deve ser capaz de funcionar com o mínimo de bibliotecas de software distintas sem que o resultado seja comprometido.

d) Bibliotecas distintas MPI, OpenMP e OpenACC

O modelo deve ser capaz de responder bem, funcionar sem erros, ao uso de diferentes implementações do MPI, e de paralelismo com diretivas OpenMP e OpenACC.

e) Compiladores

O modelo deve ser capaz de ser compilado por pelo menos 3 compiladores distintos: GNU, INTEL e PGI/NVIDIA

Para cada um dos modelos, as bibliotecas necessárias, ou seja, as suas dependências, são:

I. MPAS

- NetCDF-C/Fortran: NetCDF (network Common Data Form) é um conjunto de bibliotecas de software e de formatação de dados que independe de arquitetura de máquina e que permite a criação, acesso e compartilhamento de dados científicos
- Parallel-NetCDF: PnetCDF (Parallel netCDF) é uma biblioteca de I/O paralelo de alta performance para acesso a dados em formato compatível com o NetCDF, especificamente os formatos CDF-1, 2, e 5.
- HDF5: HDF5 é um modelo de dados, biblioteca e formato para arquivamento e gerência de armazenamento. Suporta uma grande variedade de tipos de dados (data types) e foi projetado para I/O flexível e eficiente quando está trabalhando com grande volume de dados.
- ParallelIO: Parallel IO libraries (PIO) é uma biblioteca de I/O de alto-nível para aplicações C e Fortran que precisam de NetCDF para um número elevado de processadores em sistemas de processamento de alto desempenho.

II. SHIELD (FV3)

As mesmas bibliotecas requeridas pelo MPAS (exceto ParallelIO) mais as seguintes:

- Bacio: Biblioteca BACIO realiza I/O binário para modelos do NCEP e processa registros de dados (data records) endereçados por bytes e formatado com N dimensões e faz a transformação entre dados/arquivos little-endian e big-endian.
- ESMF: Earth System Modeling Framework (ESMF) é um software de infra-estrutura de alta performance e flexível para a construção de modelos acoplados de tempo, clima e aplicações de ciências da Terra. O

ESMF define uma arquitetura para compor sistemas de modelagem complexos e acoplados e inclui estruturas de dados e utilidades para o desenvolvimento de modelos individuais.

- NEMSIO: NOAA Environmental Modeling System I/O (NEMSIO). É uma biblioteca de funções básicas e fornece conjunto de dados para leitura e escrita para todas as aplicações NEMS(NOAA Environmental Modeling System).
- spLIB: Biblioteca com diversas funções em subprogramas Fortran para transformada espectral. É parte das bibliotecas do projeto NCEPLIBS.
- w3nco: Biblioteca com rotinas Fortran90 para codificar/decodificar formato GRIB edição 1 com as mudanças da NCO (NCEP Central Operations).

Para instalação dos softwares que são pré-requisito dos modelos, foi empregado o gerenciador de pacotes *Spack* (<https://spack.io/>). Dentre os pacotes disponíveis no *Spack*, incluem-se os modelos MPAS (*mpas-model*) e UFS (*ufs-weather-model*). Assim como o modelo SHIELD, o modelo UFS também utiliza o FV3 no núcleo da dinâmica, e possui praticamente os mesmos pré requisitos de software.

Ao proceder a instalação de um pacote, tais como o *mpas-model* e o *ufs-weather-model*, conseqüentemente, o *Spack* providencia também a instalação de todas as suas dependências, ou seja, os respectivos softwares e bibliotecas pré-requisito. Assim foi feito em todos os ambientes computacionais.

O *Spack* provê suporte para uso de diferentes compiladores para instalação dos pacotes. Portanto, para todas as bibliotecas, com o *Spack* foram também geradas instalações para cada um dos três compiladores utilizados nos testes de portabilidade: GNU, INTEL e PGI/NVIDIA.

Foram utilizados critérios de pontuação para avaliar os requisitos. Esses critérios são mostrados na Tabela 2.4.

Tabela 2.4 - Critérios de pontuação de acordo com as métricas de avaliação de portabilidade.

Métrica	Descrição	Impactos Positivos	Impactos Negativos	Ferramentas	Unidade de medida
Arquitetura Dependência de máquina	Verifica diferentes arquiteturas que o modelo roda. Deve ser capaz de rodar pelo menos em 1 delas: Cluster Massivo Paralelo, GPU	- Adaptabilidade - Capacidade para ser Instalado	-	Rodada com avaliação de saídas. Avalia-se confiabilidade e portabilidade	Pontos por status funcional: +1.0
Processadores Capacidade de funcionar em diferentes processadores	Verifica se o core dinâmico é funcional e estável em processadores X86 (AMD ou INTEL)	- Adaptabilidade - Capacidade para ser Instalado	-	Rodada com avaliação de saídas. Avalia-se confiabilidade e portabilidade	Pontos por status funcional: +1.0
Pacotes Verifica a necessidade de uso de pacotes de software extra para uso com o modelo	Deve usar o mínimo de pacotes e bibliotecas adicionais para funcionar		- Coexistência - Capacidade de substituir	Verificação da exigência	Pontos negativos por pacote extra: -0.2
Bibliotecas de comunicação Verifica a capacidade do software funcionar com diversos tipos de biblioteca de comunicação	Deve ser capaz de funcionar com implementação de MPI, OpenMP e OpenACC	- Coexistência - Capacidade de substituir	-	Rodada Teste de funcionalidade	Pontos por biblioteca +1.0
Compiladores Verifica a capacidade de compilar em diversos compiladores	Avalia se modelo compila com compilador GNU, NVIDIA e INTEL	- Adaptabilidade - Capacidade para ser Instalado	-	Compilação e Rodada Teste de funcionalidade	Pontos por compilador +1.0

Fonte: Produção dos autores.

2.3.3 Performance, Escalabilidade e Eficiência (PEE)

Para avaliar os NDs, utilizou-se os ambientes computacionais Minerva e Rattler. Outros ambientes computacionais foram considerados e testados. Entretanto, foram descartados para finalidade das avaliações dos NDs. A razão

do descarte é que os demais ambientes testados tem arquitetura e sistemas semelhantes aos dois sistemas já mencionados, Minerva e Rattler.

Cada ND foi submetido a cinco execuções de 12 horas de previsão cada, a média das cinco execuções é a medida do tempo final considerada para o cálculo das métricas de performance “*Speedup*” e “Eficiência”. O *speedup* (S_p) (ou aceleração) é a relação entre o tempo gasto para executar uma tarefa com um único processo e o tempo gasto com “p” processos. Quanto mais próximo de “p” for o *speedup*, melhor, pois se aproxima do *speedup* linear (p), considerado ideal. Neste caso, considerou-se como sequencial as execuções utilizando um a dois nós do ambiente computacional, dependendo do modelo e da arquitetura (CPU/GPU). O *speedup* é definido pela Equação 2.1:

$$S_p = \frac{T_1}{T_p} \quad (2.1)$$

Onde:

S_p - *Speedup* utilizando “p” processos

T_1 - Tempo sequencial

T_p - Tempo utilizando “p” processos

Outra métrica utilizada para avaliar a performance dos NDs foi a “Eficiência” (E_p), que é o *speedup* normalizado pelo número de processos, ou seja, quanto mais próximo de 1 (ou 100%) melhor, à semelhança do *speedup* linear se aproximando de “p”. A eficiência é definida pela Equação 2.2:

$$E_p = \frac{S_p}{p} \quad (2.2)$$

Onde:

E_p - Eficiência utilizando “p” processos

S_p - *Speedup* utilizando “p” processos

p - Número de processos utilizados.

Utilizou-se o gerenciador de pacotes *Spack* (SPACK, 2022), especialmente desenvolvido para facilitar a instalação de softwares científicos em ambientes de supercomputadores. Os códigos do modelo MPAS, por exemplo, foram obtidos e instalados, primeiramente na sua versão 7.5, através do Spack. A compilação do sistema SHIELD foi facilitada pela utilização do spack, pois sua base de bibliotecas pôde ser gerada previamente com a compilação do UFS usando o *Spack*.

3 RESULTADOS

3.1 Manutenibilidade

Foram avaliados os núcleos dinâmicos do FV3 (SHIELD), GEF e MPAS. Os códigos dos núcleos dinâmicos avaliados estão disponíveis no repositório do *GitHub* em https://github.com/monanadmin/monan/tree/main/codigos_originais. Nesta primeira avaliação automatizada, seguindo os procedimentos documentados, o modelo GEF obteve um melhor desempenho em algumas métricas como documentação total, documentação de rotinas, tamanho médio dos módulos, *Fan-In*, *Fan-out* e aninhamento média de rotinas, como pode ser visto na Tabela 3.1.

Tabela 3.1 - Análise das métricas e pontuação dos modelos FV3, GEF e MPAS.

Métricas de Manutenibilidade	Avaliação			Pontuação		
	FV3	GEF	MPAS	FV3	GEF	MPAS
Comprimento de código	42.014,00	107.221,00	264.016,00	3	1	1
Complexidade Ciclomática Média (McCabe)	16,82	18,08	12,62	1	1	3
Documentação total	0,24	2,09	0,25	1	3	1
Documentação de rotinas	0,36	0,77	0,55	1	3	1
Documentação de arquivos	0,40	0,39	0,32	3	1	1
Documentação de estruturas de controle	0,02	0,31	0,11	1	3	1
Tamanho médio das rotinas	75,95	158,76	97,85	3	1	1
Tamanho médio dos módulos	1.623,31	41,26	916,98	1	3	1
Tamanho médio do nome das variáveis	5,24	4,09	9,45	1	1	3
Razão de <i>only</i> em <i>uses</i>	92,77	3,56	47,22	3	1	1
Razão de <i>"goto"</i> e <i>"continue"</i> por laço	2,50	13,76	0,00	1	1	3

(Continua)

Métricas de Manutenibilidade	Avaliação			Pontuação		
	FV3	GEF	MPAS	FV3	GEF	MPAS
Razão de "exit" e "cycle" por laço	0,63	0,25	2,17	1	1	3
Razão do uso de "implicit"	4,43	8,19	0,32	1	3	1
Total de "equivalence" ou "common"	0,00	44,00	0,00	3	1	3
Profundidade média de laços	7,81	13,56	13,26	3	1	1
Aninhamento médio de laços	0,30	0,38	0,19	1	1	3
Fan-in (mesma rotina sendo chamada)	4,31	86,00	4,66	1	3	1
Fan-out (rotinas chamadas por rotina)	8,35	3,00	16,50	1	3	1
Aninhamento médio de rotinas	1,89	1,21	1,93	1	3	1
TOTAIS				31	35	31

Fonte: Produção dos autores.

No entanto, ao aplicar uma análise qualitativa do código, verificou-se alguns pontos (destacados em negrito na Tabela 3.1) onde houve benefício indevido a um dos modelos. Isso foi devido a:

- A documentação gerada automaticamente para todo o código não é informativa, isto é, foram gerados os cabeçalhos automáticos mas não foram inseridas informações explicativas sobre o conteúdo dos códigos. Com isso, houve benefício nos itens relacionados à documentação.

Obs: O software de avaliação usado conta o número de linhas de cabeçalho. Não verifica o conteúdo do mesmo.

- Do total de módulos, 75 deles não contêm rotinas, os quais colaboraram para gerar uma maior pontuação nos itens "Tamanho médio dos módulos" e "Razão do uso de 'implicit'".

Obs: Nesses casos muitos módulos são apenas memórias de outros módulos, isto é, como se fossem *includes*.

- Rotinas criadas dentro de arquivos sem módulos, o que é desencorajado pelos padrões de codificação, geraram imprevistos de cálculo na ferramenta *FortranMakeUtils*, aumentando significativamente as métricas *Fan-in* e *Fan-out* e diminuindo a métrica "Aninhamento médio de rotinas".

Tabela 3.2 - Análise das métricas e pontuação dos modelos FV3, GEF e MPAS sem os critérios relativos a documentação, tamanho médio dos módulos, "Implicit", "Fan-in" e "Fan-out".

Métricas de Manutenibilidade	Núcleos Dinâmicos						ANA	MOD	REU	EST	TST	ANA	MOD	REU	EST	TST	ANA	MOD	REU	EST	TST	
	FV3	GEF	MPAS	FV3	GEF	MPAS	FV3					GEF					MPAS					
Comprimento de código	42.014	107.221	264.016	3	1	1	3	3	3	3	3	1	1	1	1	1	1	1	1	1	1	1
Complexidade Ciclomática Média (McCabe)	16,82	18,08	12,62	1	1	3	1	1	1	1	1	1	1	1	1	1	3	3	3	3	3	3
Tamanho médio das rotinas	75,95	158,76	97,85	3	1	1	3	3	3	3	3	1	1	1	1	1	1	1	1	1	1	1
Tamanho médio do nome das variáveis	5,24	4,09	9,45	1	1	3	1	1			1	1	1			1	3	3				3
Razão de only em uses	92,77	3,56	47,22	3	1	1	3	3		3		1	1		1		1	1			1	
Razão de "goto" e "continue" por laço	2,50	13,76	0,00	1	1	3	1	1		1		1	1		1		3	3			3	
Razão de "exit" e "cycle" por laço	0,63	0,25	2,17	1	1	3	1	1		1		1	1		1		3	3			3	
Total de "equivalence" ou "common"	0,00	44,00	0,00	3	1	3		3		3			1		1			3			3	
Profundidade média de laços	7,81	13,56	13,26	3	1	1	3	3	3	3	3	1	1	1	1	1	1	1	1	1	1	1
Aninhamento médio de laços	0,30	0,38	0,19	1	1	3	1	1	1	1	1	1	1	1	1	1	3	3	3	3	3	3
TOTAIS	-	-	-	20	10	22	17	20	11	19	12	9	10	5	9	6	19	22	9	19	12	12

ANA, Analisabilidade; MOD, Modificabilidade; REU, Reusabilidade; EST, Estabilidade e TST, Testabilidade.

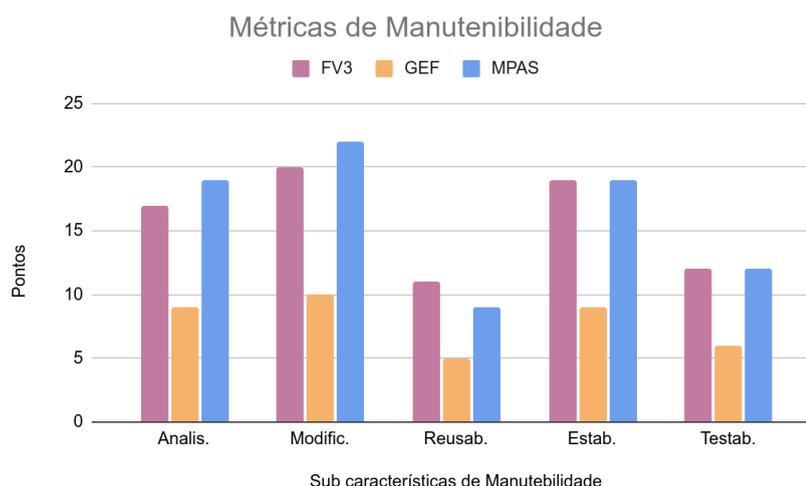
Fonte: Produção dos autores.

Levando em consideração os três pontos citados, os critérios destacados em negrito na Tabela 3.1 foram removidos desta avaliação com os três modelos. Seguindo os critérios de pontuação definidos no Apêndice B, foram obtidos os resultados compilados na Tabela 3.2.

Os melhores resultados das métricas receberam três pontos, e os piores um ponto. Esta pontuação foi associada às colunas com os nomes dos modelos e às colunas com sub características associadas às métricas pontuadas.

As maiores pontuações, totalizadas na última linha, representam as características que foram mais consideradas. A Figura 3.1 ilustra graficamente os resultados dos totais apresentados na Tabela 3.2.

Figura 3.1 - Métricas das sub características da Manutenibilidade com os modelos FV3, GEF e MPAS.



Fonte: Produção dos autores.

Ao analisar a pontuação do modelo GEF, utilizando as métricas possíveis de comparação para os três modelos, verificou-se que todas suas pontuações em métricas obtiveram valor mínimo (1 ponto). Por esse motivo, o modelo foi considerado insatisfatório com relação ao desempenho em Manutenibilidade e uma nova avaliação foi feita somente com os modelos FV3 e MPAS, incluindo-se todas as métricas, de onde se obtiveram os resultados apresentados na Tabela 3.3.

Tabela 3.3 - Análise do conjunto total de métricas e pontuação dos modelos considerados satisfatórios.

Métricas de Manutenibilidade	Pontuação		ANA	MOD	REU	EST	TST	ANA	MOD	REU	EST	TST
	FV3	MPAS	FV3					MPAS				
Comprimento de código	3	1	3	3	3	3	3	1	1	1	1	1
Complexidade Ciclomática Média (McCabe)	1	3	1	1	1	1	1	3	3	3	3	3
Documentação total	1	3	1	1				3	3			
Documentação de arquivos	3	1	3	3				1	1			
Documentação de estruturas de controle	1	3	1	1				3	3			
Tamanho médio das rotinas	3	1	3	3	3	3	3	1	1	1	1	1
Tamanho médio dos módulos	1	3	1	1	1	1	1	3	3	3	3	3
Tamanho médio do nome das variáveis	1	3	1	1			1	3	3			3
Razão de <i>only</i> em <i>uses</i>	3	1	3	3		3		1	1		1	
Razão de <i>goto</i> e <i>continue</i> por laço	1	3	1	1		1		3	3		3	
Razão de <i>exit</i> e <i>cycle</i> por laço	1	3	1	1		1		3	3		3	
Razão do uso de <i>implicit</i>	3	1	3	3		3		3	3		3	
Total de <i>equivalence</i> ou <i>common</i>	3	3		3		3			3		3	
Profundidade média de laços	3	1	3	3	3	3	3	1	1	1	1	1
Aninhamento médio de laços	1	3	1	1	1	1	1	3	3	3	3	3
<i>Fan-in</i> (mesma rotina chamada)	1	3	1	1	1	3	1	3	3	3	1	3
<i>Fan-out</i> (rotinas chamadas por rotina)	3	1		3	3	3	3		1	1	1	1
Aninhamento médio de rotinas	3	1		3	3	3	3		1	1	1	1
TOTAIS	37	41	28	37	20	32	21	38	43	20	28	23

ANA, Analisabilidade; MOD, Modificabilidade; REU, Reusabilidade; EST, Estabilidade e TST, Testabilidade.

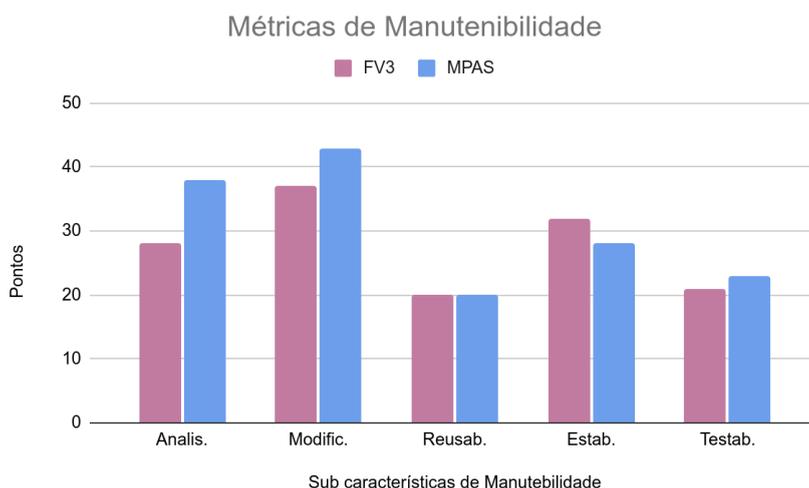
Fonte: Produção dos autores.

Ao incluir critérios removidos na avaliação com o GEF, a Analisabilidade passa a ser mais importante que a Estabilidade, como se pode observar nos totais da Tabela 3.3. O modelo FV3 totalizou $28 + 37 + 20 + 32 + 21 = 138$ pontos e o MPAS 152 pontos na somatória geral da Manutenibilidade.

A Figura 3.2 ilustra graficamente os totais da Tabela 3.3, onde se verifica uma superioridade do modelo MPAS nas características de Analisabilidade, Modificabilidade e Testabilidade, enquanto o FV3 supera o MPAS na

característica de Estabilidade. Os modelos empataram em termos de Reusabilidade.

Figura 3.2 - Métricas das sub características da Manutenibilidade dos modelos considerados satisfatórios.



Fonte: Produção dos autores.

3.2 Portabilidade

Nesta seção é avaliada a portabilidade dos códigos referentes aos códigos do núcleo da dinâmica do MPAS e do FV3 (SHIELD/UFS), nos ambientes computacionais disponíveis.

3.2.1 MPAS

A portabilidade do MPAS foi avaliada para todos os requisitos básicos descritos na seção 2.3.2, seguindo critérios definidos na Tabela 2.4. Na Tabela 3.4 está a pontuação obtida para cada uma das métricas dos requisitos para o MPAS, onde são indicadas as tarefas (*issues*) no projeto do repositório *GitHub* do MONAN (MONAN, 2022), que validam os modelos para as métricas de Arquitetura e Compiladores.

Tabela 3.4 - Pontuação do modelo MPAS de acordo com as métricas de avaliação de portabilidade.

Métrica	MPAS	Pontos
Arquitetura Dependência de máquina +1.0 pt / arquitetura	- Funciona na Egeon e na Minerva (issues #44 e #63) - Funciona em GPU no cluster Rattler (issue #53)	2.0
Processadores Funcionar em diferentes processadores: +1 pt.	- Funciona em X86 (AMD e INTEL)	1.0
Pacotes Pacotes extra para uso com o modelo: -0.2 pt / pacote	- HDF5, netCDF, Parallel-NetCDF, Parallelio (4 pacotes)	-0.8
Bibliotecas de comunicação Tipos de biblioteca de comunicação: +1 pt / bib.	- MPI - OpenMP - OpenACC	3.0
Compiladores Compiladores: +1 pt / comp.	- GNU: gfortran, gcc, g++ (issue #56) - INTEL: ifort, icc, icpc (issue #61) - PGI/NVIDIA: pgf90, pgcc, pg++ (issue #45)	3.0
Pontuação Geral		8.2

Fonte: Produção dos autores.

3.2.2 FV3

A portabilidade do FV3 foi avaliada para todos os requisitos básicos descritos na seção 2.3.2, seguindo critérios de pontuação definidos na Tabela 2.4. Na Tabela 3.5 está a pontuação obtida para cada uma das métricas dos requisitos para o FV3, onde são indicadas as tarefas (*issues*) no projeto do repositório GitHub do MONAN, que validam os modelos para as métricas Arquitetura e Compiladores.

Para os testes de portabilidade foi utilizado o modelo UFS, com resolução C96 (~100km), tendo a finalidade de verificar se o código do FV3 compilava e executava corretamente nas arquiteturas disponíveis. O pacote utilizado para a previsão global com 24h de integração nesta resolução foi obtido a partir de

<https://ftp.emc.ncep.noaa.gov/EIB/UFS/simple-test-case.tar.gz>, onde a configuração dos parâmetros é dada a seguir:

- layout = 1,1
- io_layout = 1,1
- npx = 97
- npy = 97
- ntiles = 6
- npz = 64

Tabela 3.5 - Pontuação do FV3 de acordo com as métricas de avaliação de portabilidade.

Métrica	FV3	Pontos
Arquitetura Dependência de máquina +1.0 pt / arquitetura	- Funciona na Egeon e na Minerva: issues #38 e #60	1.0
Processadores Funcionar em diferentes processadores: +1 pt.	- Funciona em X86 (AMD e INTEL)	1.0
Pacotes Pacotes extra para uso com o modelo: -0.2 pt / pacote	- HDF5, netCDF, Parallel-NetCDF, Bacio, ESMF, NEMSIO, spLIB, w3nco (8 pacotes)	-1.6
Bibliotecas de comunicação Tipos de biblioteca de comunicação: +1 pt / bib.	- MPI - OpenMP	2.0
Compiladores Compiladores: +1 pt / comp.	- GNU: gfortran, gcc, g++: issue #37 - INTEL: ifort, icc, icpc: issue #41	2.0
Pontuação Geral		4.4

Fonte: Produção dos autores.

3.3 Performance, Escalabilidade e Eficiência (PEE)

Nesta seção são mostrados e avaliados os resultados de desempenho paralelo do núcleo da dinâmica dos modelos MPAS e SHIELD (FV3). Os testes de desempenho foram conduzidos sobre configurações específicas de rodada, detalhadas a seguir para cada um dos modelos.

3.3.1 MPAS

Para a avaliação de desempenho do núcleo dinâmico do modelo MPAS, foi escolhida uma das parametrizações do benchmark mantido pelos desenvolvedores do modelo, disponibilizado no endereço <https://www2.mmm.ucar.edu/projects/mpas/benchmark/>. São para três versões do modelo (v5.2, v6.x, v7.0). Contém os dados de condição inicial para diferentes resoluções de malha, onde a mais grossa é de 120km e a mais fina de 10km. Os arquivos de namelist com a parametrização espacial e física de execução, são também disponibilizados. Para os testes de desempenho mostrados neste relatório, foi escolhido o benchmark relativo a malha com resolução espacial de 15 km. A motivação para escolha de malhas mais grossas se deve ao fato da quantidade de memória disponível nos nós computacionais das máquinas usadas no teste.

O tempo de execução para 12 horas de integração do modelo é mostrado na Tabela 3.6, em rodada realizada com 2 nós do cluster Minerva, usando 64 núcleos por nó. É também mostrado o tempo de execução em diferentes sub-rotinas do MPAS, sendo que na cor azul estão destacadas aquelas relativas ao núcleo da dinâmica. O tempo total obtido foi de 5930.33s, e o tempo somado das sub-rotinas do núcleo da dinâmica foi de 3019.63s. Extrapolando-se para 24h de integração, este tempo de processamento é o dobro, ou seja, 6039.26s.

Tabela 3.6 - Tempos de execução retornados pela saída do modelo MPAS.

Timmer_Name		Total (s)
1	total time	5920.33105
2	initialize	32.38702
2	time integration	5886.15186
3	physics driver	2033.52197
4	calc_cldfraction	0.58155

(Continua)

Timmer_Name		Total (s)
4	RRTMG_sw	756.18988
4	RRTMG_lw	429.19873
4	Monin-Obukhov	4.56684
4	Noah	12.75321
4	YSU	168.60727
4	GWDO_YSU	36.13308
4	New_Tiedtke	512.27197
3	atm_rk_integration_setup	46.28673
3	atm_compute_moist_coefficients	23.35006
3	physics_get_tend	344.41339
3	atm_compute_vert_imp_coefs	31.42596
3	atm_compute_dyn_tend	934.06744
3	small_step_prep	111.46519
3	atm_advance_acoustic_step	311.87317
3	atm_divergence_damping_3d	81.90127
3	atm_recover_large_step_variables	348.64746
3	atm_compute_solve_diagnostics	493.36813
3	atm_rk_dynamics_substep_finish	102.01469
3	atm_advance_scalars	196.52936
3	atm_advance_scalars_mono	420.60504
3	microphysics	438.77209
4	WSM6	389.99307

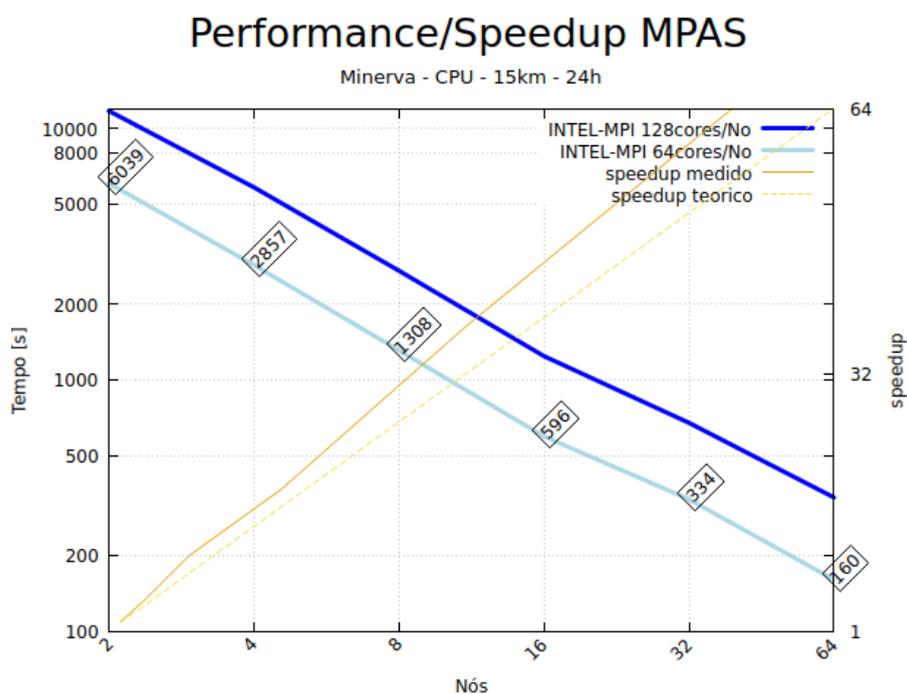
Rodada rodada com 12 horas de integração do benchmark de 15km. Destacados em azul as sub-rotinas relativas ao núcleo da dinâmica.

Fonte: Produção dos autores.

A Figura 3.3 mostra os tempos de processamento paralelo. A figura apresenta os resultados obtidos em duas simulações distintas. Uma rodada submetida usando 128 núcleos (cores) por nó e outra utilizando 64 núcleos. Observa-se que as rodadas que usam 64 núcleos por nó tem melhor desempenho (performance). Não foram verificados os motivos que levam a essa melhora de desempenho, contudo é esperado que tal variação seja resultado de menor uso de memória em cada nó. A investigação para esse comportamento será realizada em experimentos futuros.

Nas curvas de *speedup* (ganho de desempenho), teórico x medido, é observado que o modelo apresentou um comportamento super linear. Esse comportamento é determinado quando $Sp > p$ (ver equação 2.1). Comportamentos de super-escalabilidade (super linear) são sintomas patológicos. Em geral estão associados à topologia de memória, formas de acesso à memória (*cache, page fault, etc*), uso concomitante de OpenMP e MPI ou outras causas. A investigação para esse comportamento deverá ser realizada em experimentos futuros.

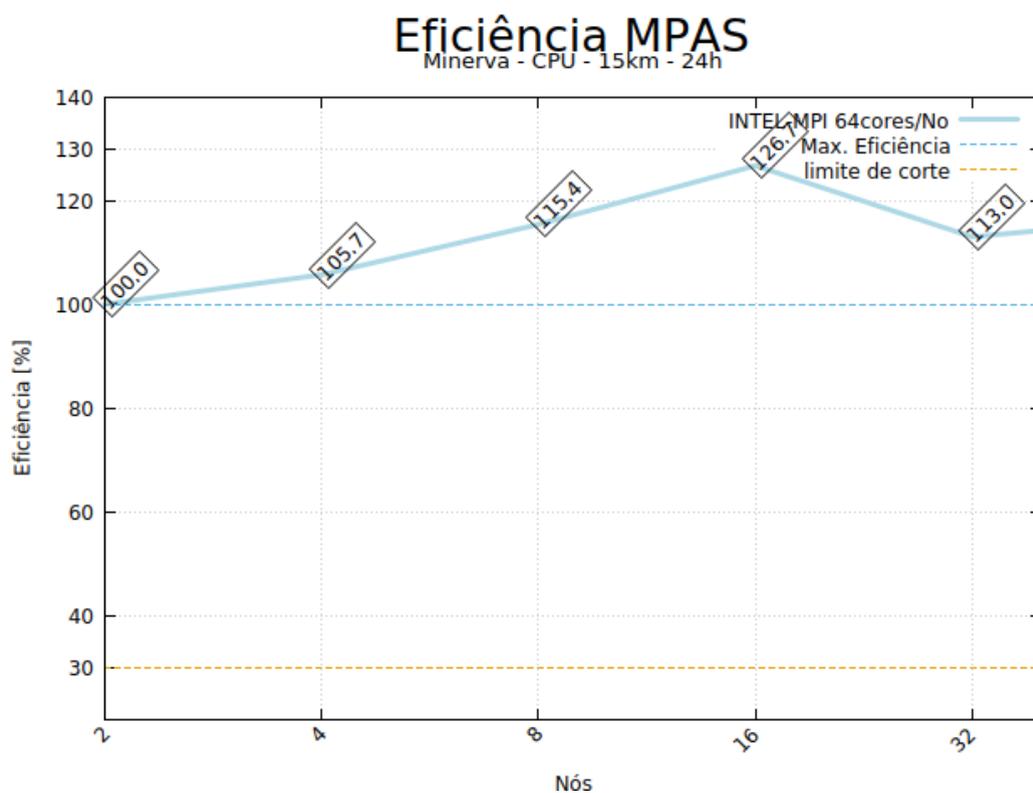
Figura 3.3 - Tempo de processamento paralelo (MPI) e *speedup* (Ganho de desempenho) do núcleo da dinâmica do MPAS no cluster Minerva.



Fonte: Produção dos autores.

Na Figura 3.4 vemos a eficiência do paralelismo (2). Devido ao comportamento super linear a eficiência medida é maior que a eficiência máxima esperada, isto é, maior que 100%. Isso fere a lei de *Amdahl* que trata da relação entre a fração paralelizável e a fração escalar de um código. Contudo os casos de super-escalabilidade levam a esse comportamento e devem ser investigados.

Figura 3.4 - Eficiência do processamento paralelo (MPI) do núcleo da dinâmica do MPAS no cluster Minerva.



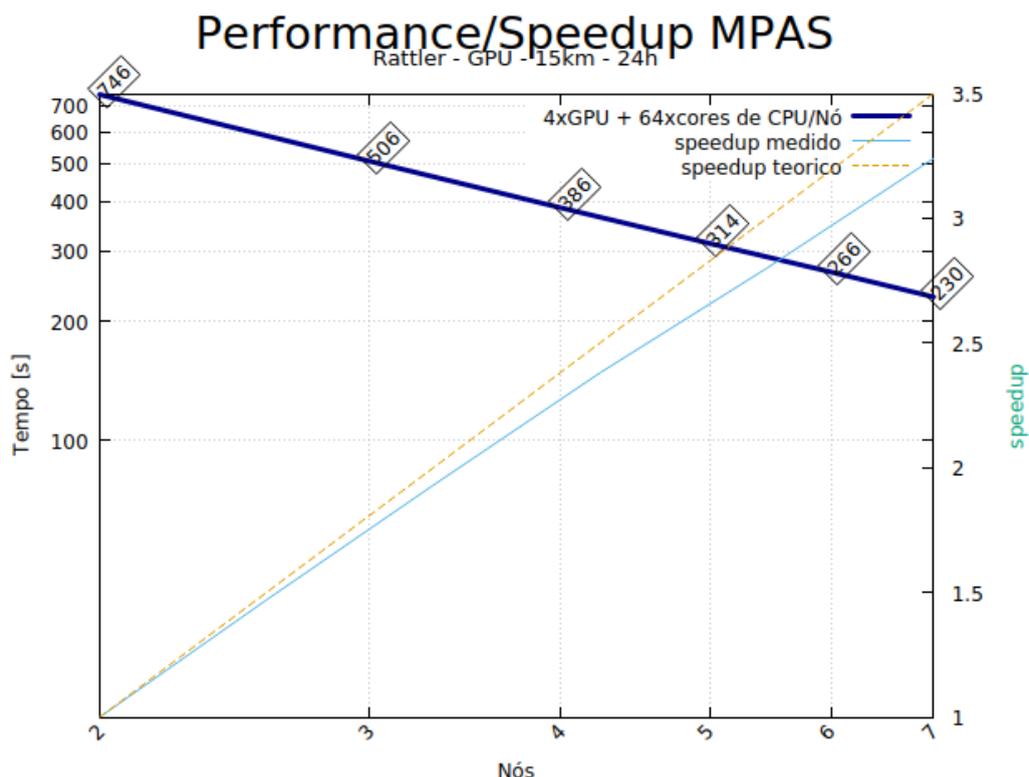
Observe a super linearidade com eficiência > 100%.

Fonte: Produção dos autores.

O modelo MPAS foi submetido para rodar na máquina *Rattler*. O modelo está preparado para rodar com GPU utilizando diretivas *OpenACC*, padrão de programação para computação paralela (OpenACC, 2022). Essa máquina dispõe apenas de 7 nós (448 núcleos) na configuração adequada para a

rodada do modelo. Portanto o experimento foi realizado até esse limite. A Figura 3.5 mostra a performance e *speedup* do núcleo dinâmico do modelo nesse sistema. Observa-se um comportamento excelente de performance e *speedup*.

Figura 3.5 - Tempo de processamento paralelo (MPI) e *Speedup* do núcleo da dinâmica do MPAS no cluster Rattler.



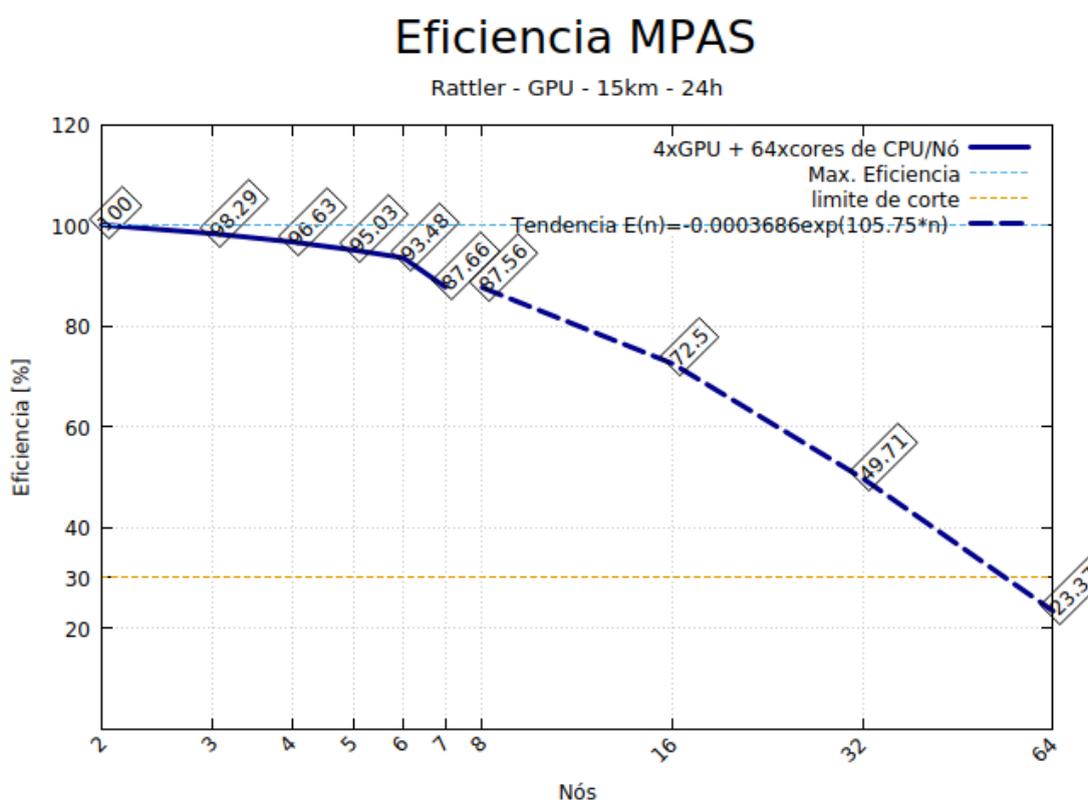
Uso de 4 GPUs NVIDIA A100 por nó.

Fonte: Produção dos autores.

Na Figura 3.6 é apresentado o resultado da eficiência do modelo MPAS no sistema *Rattler*. Foi determinada uma equação para representar a tendência da eficiência nos 7 nós disponíveis e aplicada a mesma equação para determinar a eficiência para número de núcleos acima de 448. Essa simulação será útil na comparação dos núcleos dinâmicos em diferentes sistemas computacionais. A aplicação da equação e os resultados são teóricos.

Em geral, o comportamento real dos modelos em sistemas paralelos não acompanham curvas com estimação baseadas em equações de tendência. Mas por falta de sistemas maiores isso se tornou necessário. Aplicando a equação observa-se que a eficiência cai abaixo de 30% para 4096 núcleos computacionais. Esse limite de 30% foi adotado como limite mínimo aceitável.

Figura 3.6 - Eficiência do processamento paralelo (MPI) do núcleo da dinâmica do MPAS no cluster Rattler.

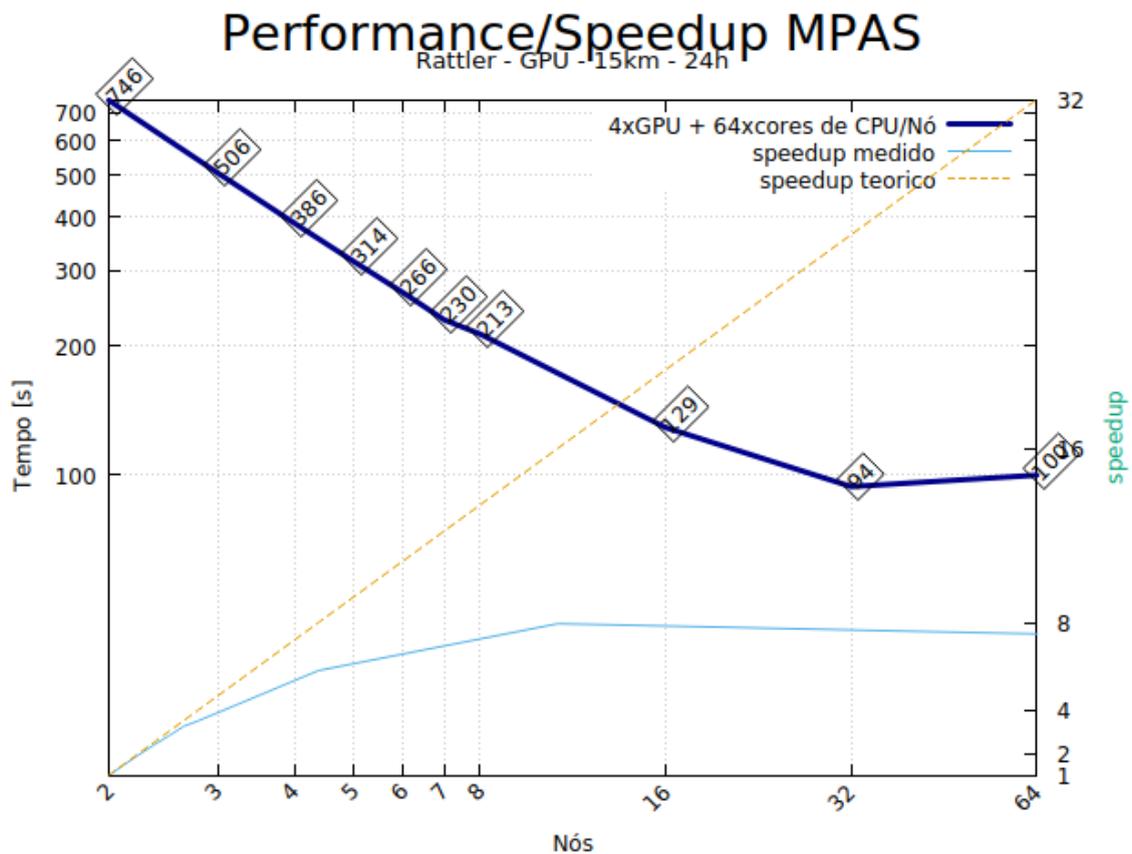


Uso de 4 GPUs NVIDIA A100 por nó.

Fonte: Produção dos autores.

Utilizando a equação de tendência levantada é possível extrapolar a performance e *speedup* até 32 nós. Observa-se que, até 16 nós, tem-se ganho aproveitável. A Figura 3.7 mostra um gráfico com os valores extrapolados.

Figura 3.7 - Performance e *speedup* medidos até 7 nós e teóricos (extrapolados) acima de 7 nós.



Fonte: Produção dos autores.

3.3.2 FV3

Esta seção exibe os resultados de Performance, Escalabilidade e Eficiência do FV3 através da execução do modelo SHIELD (FV3) no Cluster Minerva, onde foi possível avaliar os requisitos de até 64 nós de CPUs.

Até o momento de elaboração destes resultados, não foi possível concluir a compilação do Shield com GPU, que não está preparado para o compilador da NVIDIA. Futuros trabalhos poderão apresentar estes resultados.

A Figuras 3.8 e Figura 3.9 apresentam a Performance, *Speedup* e Eficiência do SHIELD (FV3) em duas linhas de execução: utilizando 96 e 72 processadores por core. Devido a grade cúbica do modelo, o número de núcleos para a rodada do Shield deve ser compatível com sua configuração de LayoutX,

LayoutY e ntilas=6. Isso força a escolha de um número de núcleos múltiplos destas configurações para cada rodada. Por esse motivo, foi estressado o máximo de processadores por nó (96) e também foram feitos testes com 72 processadores, número mais próximo para comparação com o número de processadores utilizado nos testes do MPAS (64).

Os resultados dos tempos do FV3 foram obtidos a partir do relatório de execução do SHIELD com 12h de integração. Na Tabela 3.7, observa-se o tempo total e o tempo específico do ND destacado em azul, onde foram utilizados 3072 processadores distribuídos em 32 nós (96 cores por nó):

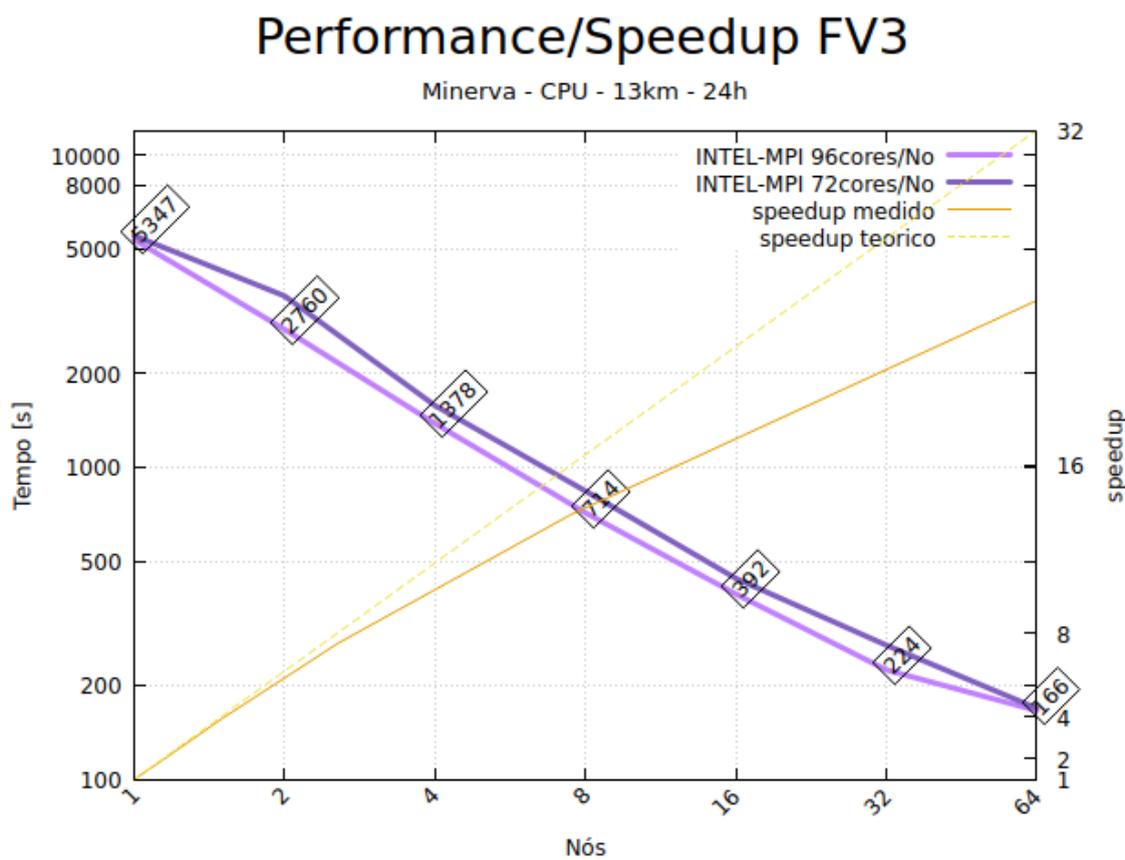
Tabela 3.7 - Especificação dos tempos da rodada FV3.

Tabulating mpp_clock statistics across 3072 PEs									
	hits	tmin	tmax	tavg	tstd	tfrac	grain	pemin	pemax
Total runtime	1	316.53299	316.537048	316.53479	0.000586	1	0	0	3071
Initialization	1	46.709026	47.009949	46.959171	0.110229	0.148	0	0	3071
FV dy-core	576	95.576279	137.230118	111.872803	10.009806	0.353	11	0	3071
FV subgrid_z	288	0.122203	0.368742	0.215803	0.055221	0.001	11	0	3071
FV Diag	288	1.965	2.972553	2.283237	0.156009	0.007	11	0	3071
GFS Step Setup	288	66.412865	100.922356	92.205933	9.730636	0.291	1	0	3071
GFS Radiation	288	2.741683	15.935287	8.426137	2.717996	0.027	1	0	3071
GFS Physics	576	15.045197	20.578415	16.700846	0.763495	0.053	1	0	3071
Dynamics get state	288	4.008958	4.928978	4.20271	0.081553	0.013	1	0	3071
Dynamics update state	288	9.64494	51.693634	26.468935	10.011416	0.084	1	0	3071
FV3 Dycore	576	99.676285	141.782654	116.474121	10.018317	0.368	1	0	3071
Main loop	1	238.051605	238.054733	238.052979	0.000446	0.752	0	0	3071
Termination	1	12.26896	12.269122	12.269039	2.5E-05	0.039	0	0	3071

Fonte: Produção dos autores.

No caso, o FV3 levou o tempo de 111.87s no total para a integração de 12h. Extrapolando-se para 24h, o tempo total foi de 223,74s, que pode ser observado na Figura 3.8. No gráfico os valores foram arredondados.

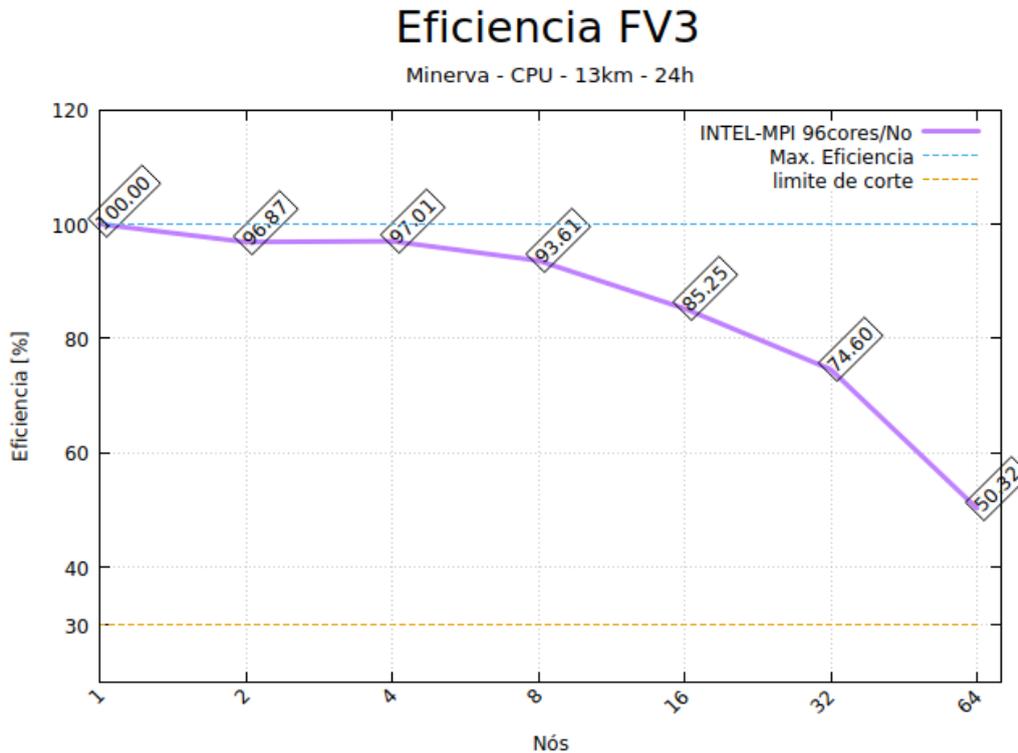
Figura 3.8 - Desempenho e speedup do core dinâmico do modelo FV3 na Minerva.



Fonte: Produção dos autores.

A eficiência do modelo usando 96 núcleos é mostrada na Figura 3.9. Observa-se que o ND apresenta boa eficiência até o limite de núcleos testados, estando sempre acima do limite de 30%.

Figura 3.9 - Eficiência do core dinâmico do modelo FV3 na Minerva.



Fonte: Produção dos autores.

4 CONCLUSÃO E DISCUSSÕES

4.1 Manutenibilidade

Ao considerar a avaliação do conjunto total de métricas para os modelos FV3, GEF e MPAS, os resultados indicaram o modelo MPAS com o maior somatório de pontuações de sub características de Manutenibilidade. Tendo em vista que o modelo GEF foi beneficiado em alguns critérios indevidamente, os cálculos foram refeitos desconsiderando estas métricas, e este foi considerado insatisfatório, por não ter nenhuma pontuação máxima (3 pontos) em todas as sub características avaliadas. Em seguida, o GEF foi removido da análise e foram incluídas as métricas desconsideradas. Nesta segunda avaliação, o modelo FV3 totalizou 138 pontos e o MPAS 152. Os resultados indicaram que o MPAS obteve o melhor desempenho na maioria das características em ambas as situações.

Ao comparar os modelos FV3 e MPAS, considerando todas as métricas possíveis, o FV3 recebeu pontuação superior em Estabilidade, enquanto que o MPAS superou o FV3 em Analisabilidade, Modificabilidade e Testabilidade. Em termos de Reusabilidade a performance foi idêntica. Reforça-se que a Estabilidade, no contexto de Manutenibilidade, avalia a capacidade do software de evitar efeitos colaterais decorrentes de modificações introduzidas. Não se refere à estabilidade de execução dos modelos.

Destaca-se que, em termos da Manutenibilidade, os resultados aqui encontrados não são suficientes para eliminar o modelo FV3 da lista de modelos elegíveis para compor o ND do modelo MONAN. Com relação ao GEF, devido às pontuações consideradas insatisfatórias (1 ponto em todas sub características), considera-se que este modelo não é elegível para compor o ND do modelo MONAN se comparado com as duas outras opções.

4.2 Portabilidade

Os testes de portabilidade foram realizados apenas nos modelos FV3 e MPAS. Os dois modelos foram testados nas máquinas disponíveis, com os compiladores e bibliotecas de comunicação possíveis e, em geral, apresentaram bom comportamento. Não foi possível obter o modelo FV3 com portabilidade para uso em GPUs. Os critérios de pontuação foram aplicados garantindo que a instalação e compilação sejam feitas de forma mais simples e que exijam pouco dos usuários.

O resultado da pontuação aplicada mostra que o modelo MPAS teve nota 8.4 e o modelo FV3 teve nota 4.4. O FV3 tem 4 pontos desfavoráveis quando comparado com o MPAS:

- a) Falta de diretivas OpenACC (não portado);
- b) A não possibilidade de execução usando GPU (reflexo do item anterior);
- c) Exigir a instalação de uma grande quantidade de bibliotecas adicionais;

d) Não ter sido possível executá-lo com compilação NVIDIA/Portland.

4.3 Performance, Escalabilidade e Eficiência (PEE)

Os dois ND testados foram bem avaliados nos quesitos de desempenho, escalabilidade e eficiência. Contudo, a quantidade de nós disponíveis para a análise de escalabilidade poderia ser maior. Ainda, houve a indisponibilidade das máquinas, evitando que mais testes com mais CPUS e GPUS pudessem ser feitos.

A comparação de performance por número de nós permite que se entenda o comportamento e desempenho do modelo quando no uso de sistemas, ainda que os Jobs não ocupem um nó completo. Essa comparação é mais realista pois infere diretamente um custo financeiro que é dependente do total de nós de um sistema.

O gráfico apresentado na Figura 4.1 exhibe o comportamento comparado entre os modelos. Conforme discutido e apresentado na Figura 3.5, o modelo MPAS em GPU foi rodado com no máximo 7 nós. Para estimar o tempo do modelo para mais nós utilizou-se o speedup ideal e aplicou-se a tendência de eficiência mostrada na Figura 3.6.

Além disso, foi necessário inferir um valor do tempo para o MPAS fazendo-se uma aproximação pois o FV3 foi rodado com resolução de 13 km e o MPAS com resolução de 15 km. Ademais há uma diferença de número de níveis. O FV3 usou 63 níveis e o MPAS 56. Para obter o tempo estimado da rodada do MPAS com resoluções compatíveis ao FV3 usou-se obter uma razão $P_{fv3/mpas}$ (Equação 4.4) entre os dois modelos. A razão foi obtida através da Equação 4.1, Equação 4.2. e Equação 4.3:

$$A_t = 4\pi R_t^2 \quad (4.1)$$

onde:

A_t - Área da superfície da Terra

R_t - Raio da Terra (6378 km)

$$A_{fv3} = r_{fv3}^2 \quad (4.2)$$

$$A_{mpas} = r_{mpas}^2 \quad (4.3)$$

onde:

A_{fv3} - Área de um pixel do modelo FV3

A_{mpas} - Área de um pixel do modelo MPAS

r_{fv3} - Resolução do FV3

r_{mpas} - Resolução do MPAS

$$P_{fv3/mpas} = \frac{(A_t/A_{fv3}) L_{fv3}}{(A_t/A_{mpas}) L_{mpas}} \quad (4.4)$$

onde:

L_{fv3} - Número de Níveis do FV3

L_{mpas} - Número de Níveis do MPAS

Não foram consideradas as características da grade de Voronoi do MPAS ou especificidades da esfera cubada do FV3. Tratou-se cada pixel como um ponto quadrado de lado igual a resolução de cada um.

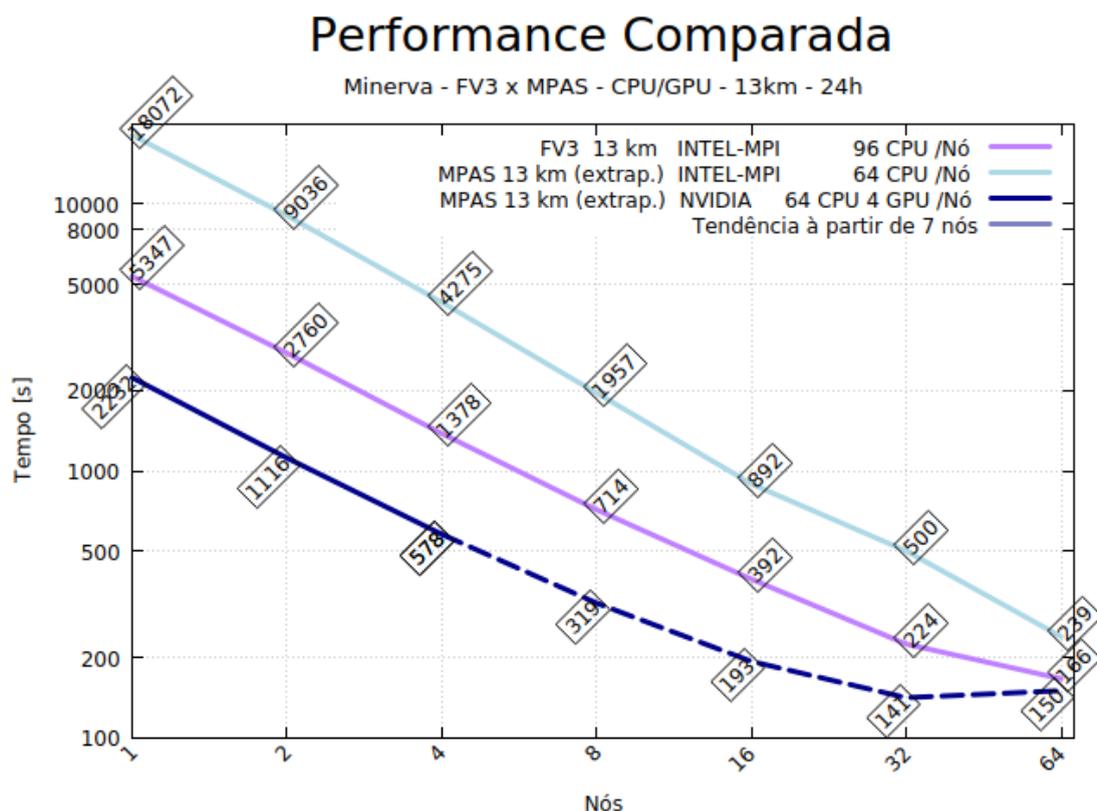
Obteve-se o valor 1,5 para essa razão. Os resultados de tempo obtidos nos experimentos com o modelo MPAS foram multiplicados por esse valor.

Há que se ressaltar que essa extrapolação pode não representar muito bem a realidade envolvendo GPUs pois, em geral, o aumento de resolução de modelos aplicadas a GPU costuma ter como resultado uma eficiência melhor. Quanto mais cálculos efetuados melhor a resposta das GPUs.

Observando os resultados apresentados nas Figura 4.1, verifica-se que há um desempenho melhor para o modelo FV3 usando CPU. O FV3 é aproximadamente 44% mais rápido que o modelo MPAS quando rodado com 64 nós.

Quando comparamos o modelo FV3 rodando em CPU com o modelo MPAS rodando em GPU observa-se que o modelo MPAS é 11% mais rápido para 64 nós. Para 32 nós, até onde o MPAS apresentou eficiência aceitável (estimada) a razão entre o MPAS e o FV3 é de 59%.

Figura 4.1 - Comparação da performance entre os modelos.



Fonte: Produção dos autores.

4.4 Discussão e Análise

O modelo GEF necessitou ser abandonado devido a algumas características de manutenibilidade. Além disso, o modelo, conforme informação de seus autores, não possui as características não-hidrostático e hidrostático simultâneas, o que é condição básica para o novo modelo MONAN.

Os resultados dos requisitos não funcionais, para os núcleos dinâmicos dos modelos FV3 e MPAS, obtidos nos experimentos nos permitem avaliar que o núcleo dinâmico do modelo MPAS apresenta vantagem na manutenibilidade e

na portabilidade. O MPAS vence com 152 pontos contra 132 do FV3 em manutenibilidade e 8,4 a 4,4 pontos em portabilidade.

A manutenibilidade é fundamental para garantir um longo ciclo de vida de software e garante mais facilidade em sua melhoria, manutenção e correção de bugs. A maturidade é diretamente dependente desses requisitos não funcionais. Com a evolução constante das arquiteturas e compiladores é fundamental que os softwares apresentem menos impedimentos e exigências de trabalhos extras para migrá-los quando da aquisição de novos sistemas computacionais ou do lançamento de novas versões de compiladores e bibliotecas. Dessa forma a vantagem apresentada pelo núcleo dinâmico do modelo MPAS é significativa e deve ser considerada na escolha do futuro núcleo dinâmico do modelo MONAN.

Um ponto importante a ser frisado é a dificuldade encontrada pelo grupo de computação científica em obter apoio na configuração, uso e documentação do núcleo dinâmico do modelo FV3. Solicitações de apoio enviadas ao grupo desenvolvedor não foram respondidas e muito do trabalho precisou ser feito de forma experimental e com todas as dificuldades inerentes a essa abordagem. O mesmo não foi observado quanto ao uso do modelo MPAS. As solicitações enviadas por e-mail foram todas respondidas e também aquelas submetidas nas listas de discussão do modelo disponível na internet (MPAS, 2022). A ausência de apoio no desenvolvimento de um software pré existente é de fundamental importância e, apesar de ser um critério embutido em parte das análises, há que ser destacado nesse parágrafo para subsidiar a escolha do ND. É possível que em um convênio formal com a NOAA o acesso seja ampliado, mas não foi possível notar um apoio mínimo durante o desenvolvimento dos testes.

A análise do desempenho dos dois núcleos dinâmicos mostra que o FV3 tem nítida vantagem em sistemas computacionais baseados em CPU. Já o núcleo dinâmico do modelo MPAS é superior que o FV3 quando comparamos rodadas em GPU e rodadas do FV3 em CPU. Infelizmente, não foi concedido o acesso

ao código do núcleo dinâmico do FV3 portado para GPU, o que poderia mostrar uma comparação direta sob a mesma arquitetura.

Há um ponto fundamental que deve ser considerado e é passível de testes, desde que com máquinas mais robustas. Em geral, a eficiência de um modelo de previsão de tempo é maior quando usado em resoluções maiores. Portanto é de se esperar que com 13 km, resolução usada no FV3, a eficiência do modelo MPAS, que foi rodado com resolução de 15 km, seja maior.

O documento “*FV3 on GPUs*” de Lucas Harris, Rusty Benson e Oli Fuhrer (Harris et al. 2021) mostra que existem esforços de migração do FV3 para GPU. Há um alerta na apresentação citada onde se verifica que o modelo necessita de algum trabalho para que possa usar adequadamente arquiteturas GPUs. Apesar dos esforços envidados junto a empresa NVIDIA para tentar receber esse código, não pudemos ser atendidos. Já o MPAS foi totalmente portado e tem excelente comportamento nessa arquitetura.

Não cabe aqui a escolha do melhor núcleo dinâmico a ser adotado pelo futuro modelo MONAN. Mas esse trabalho apresenta subsídios e aponta variáveis importantes para essa escolha.

4.5 Trabalhos futuros

Alguns pontos envolvendo os modelos ainda cabem ser elucidados. Um deles é tentar determinar a razão que leva a super linearidade observada com o modelo MPAS sendo executado em máquinas baseadas em arquitetura X86 (CPU). Outra futura investigação é tentar determinar o comportamento do modelo MPAS usando máquinas maiores baseadas em GPU, com mais memória e com as arquiteturas atualizadas.

Por fim, é fundamental que o ND escolhido seja executado de forma pré-operacional, de forma contínua, em máquinas com GPU, para uma melhor avaliação da PEE. O INPE dispõe de máquinas X86 (CPU) que permitem a rodada dos dois modelos para exames comportamentais e de estabilidade, mas não dispõe de máquinas com nós de GPU para estas tarefas, ainda que seja uma menor parte do supercomputador.

5 REFERÊNCIAS BIBLIOGRÁFICAS

- AVEC. AVEC Report: NGGPS Level-1 Benchmarks and Software Evaluation. Disponível em: <<https://www.weather.gov/media/sti/nggps/AVEC%20Level%201%20Benchmarking%20Report%2008%2020150602.pdf>>. Acesso em: 12 jun. 2023.
- CHENG, K.-Y.; HARRIS, L. M.; SUN, Y. Q. Enhancing the accessibility of unified modeling systems: GFDL System for High-resolution prediction on Earth-to-Local Domains (SHIELD) v2021b in a container, *Geosci. Model Dev.*, 15, 1097–1105 (2022). Disponível em: <<https://doi.org/10.5194/gmd-15-1097-2022>>. Acesso em: 15 jul. 2022.
- GFDL. Geophysical Fluid Dynamics Laboratory - FV3. Disponível em: <<https://www.gfdl.noaa.gov/fv3/>>. Acesso em: 20 jul. 2022 (a).
- GFDL. FV3 - Namelist Guide. Disponível em: <https://www.gfdl.noaa.gov/wp-content/uploads/2017/09/fv3_namelist_Feb2017.pdf>. Acesso em: 20 jul. 2022 (b).
- HARRIS, L.; ZHOU, L.; LIN, S.-J.; CHEN, J.-H.; CHEN, X., GAO, K., et al. GFDL SHIELD: A unified system for weather-to-seasonal prediction. *Journal of Advances in Modeling Earth Systems*, 12, e2020MS002223. (2020). Disponível em: <<https://doi.org/10.1029/2020MS002223>>. Acesso em: 16 jul. 2022.
- HARRIS, LUCAS; BENSON, RUSTY; FUHRER, OLI. FV3 on GPUs: Two approaches to accelerate global cloud-resolving modeling. 2021. Disponível em: <<https://ufsccommunity.org/wp-content/uploads/2021/04/Harris-UFS-Webinar-25-mar-21.pdf>>. Acesso em: 15 jul. 2022.
- LATINOVIĆ, D.; CHOU, S. C.; RANČIĆ, M. Seasonal range test run with Global Eta Framework, *Adv. Sci. Res.*, 14, 247–251 (2017). Disponível em: <<https://doi.org/10.5194/asr-14-247-2017>>. Acesso em: 20 jul. 2022.
- MESINGER, F.; CHOU, S.C.; GOMES, J.L. et al. An upgraded version of the Eta model. *Meteorol Atmos Phys* 116, 63–79 (2012). Disponível em: <<https://doi.org/10.1007/s00703-012-0182-z>>. Acesso em: 20 jul. 2022.
- MONAN. MONAN - Model for Ocean-IaNd-Atmosphere Prediction. GitHub Repositório. Disponível em: <<https://github.com/monanadmin/monan>>. Acesso em: 25 ago. 2022.
- MPAS. MPAS Home. Disponível em: <<http://mpas-dev.github.io>> Acesso em: 2 jan. 2023.
- MPAS. MPAS Users Group. Disponível em: <<https://groups.google.com/g/mpas-users>>. Acesso em: 10 ago. 2022.
- OPENACC. OpenACC Site. Disponível em: <<https://www.openacc.org/>>. Acesso em: 24 ago. 2022.
- PURSER, R. J. AND RANČIĆ, M. Smooth quasi-homogeneous gridding of the sphere, *Q. J. Roy. Meteorol. Soc.*, 124, 637–647, 1998.

RANČIĆ, M., PURSER, R. J., AND MESINGER, F. A global shallow-water model using an expanded spherical cube: Gnomonic versus con-formal coordinates, *Q. J. Roy. Meteorol. Soc.*, 122, 959–982, 1996.

RANČIĆ, M., PURSER, R. J., JOVIĆ, D., VASIĆ, R., AND BLACK, T. A nonhydrostatic multiscale model on the uniform Jacobian cubed-sphere, *Mon. Weather Rev.*, <https://doi.org/10.1175/MWR-D-16-0178.1>, in press, 2017.

SPACK. Spack Site. Disponível em: <<https://spack.io>>. Acesso em: 22 jun. 2022.

SADOURNY, R. Conservative finite-differencing approximations of the primitive equations on quasi-uniform spherical grids, *Mon. Weather Rev.*, 22, 1107–1115, 1972.

APÊNDICE A – CARACTERÍSTICAS DA QUALIDADE DE SOFTWARE

As características de qualidade selecionadas para a avaliação do Núcleo Dinâmico foram baseadas nas normas ISO e estão descritas abaixo.

Funcionalidade: a capacidade de um software prover funcionalidades que satisfaçam o usuário em suas necessidades declaradas e implícitas, dentro de um determinado contexto de uso. Suas sub características são:

- Adequação: Capacidade do produto de software de prover um conjunto apropriado de funções para tarefas e objetivos do usuário especificados.
- Acurácia: Capacidade do produto de software de prover, com o grau de precisão necessário, resultados ou efeitos corretos ou conforme acordados.
- Interoperabilidade: Capacidade do produto de software de interagir com um ou mais sistemas especificados.

Usabilidade: a capacidade do produto de software de ser compreendido, aprendido, operado e atraente ao usuário, quando usado sob condições especificadas. Note que este conceito é bastante abrangente e se aplica mesmo a programas que não possuem uma interface para o usuário final. Por exemplo, um programa batch executado por uma ferramenta de programação de processos também pode ser avaliado quanto a sua usabilidade, no que diz respeito a ser facilmente compreendido, aprendido, etc. Suas sub características são:

- Inteligibilidade: Capacidade do produto de software de possibilitar ao usuário compreender se o software é apropriado e como ele pode ser usado para tarefas e condições de uso específicas.
- Apreensibilidade: Capacidade do produto de software de possibilitar ao usuário aprender sua aplicação.
- Operacionalidade: Capacidade do produto de software de possibilitar ao usuário operá-lo e controlá-lo.

- Proteção frente a erros de usuários: como produto consegue prevenir erros dos usuários;

Eficiência: O tempo de execução e os recursos envolvidos são compatíveis com o nível de desempenho do software. Suas sub características são:

- Comportamento em Relação ao Tempo: avalia se os tempos de resposta (ou de processamento) estão dentro das especificações;
- Utilização de Recursos: mede tanto os recursos consumidos quanto a capacidade do sistema em utilizar os recursos disponíveis; exemplo: processador e memória.

Manutenibilidade: a capacidade (ou facilidade) do produto de software ser modificado, incluindo tanto as melhorias ou extensões de funcionalidade quanto às correções de defeitos, falhas ou erros. Suas sub características são:

- Analisabilidade: identifica a facilidade em se diagnosticar eventuais problemas e identificar as causas das deficiências ou falhas.
- Modificabilidade: caracteriza a facilidade com que o comportamento do software pode ser modificado.
- Reusabilidade: identifica a capacidade de se reutilizar o software por completo ou partes dele.
- Estabilidade: avalia a capacidade do software de evitar efeitos colaterais decorrentes de modificações introduzidas.
- Testabilidade: representa a capacidade de se testar o sistema modificado, tanto quanto as novas funcionalidades quanto as não afetadas diretamente pela modificação.

Portabilidade: a capacidade do sistema ser transferido de um ambiente para outro. Como "ambiente", devemos considerar todos os fatores de adaptação, tais como diferentes condições de infraestrutura (sistemas operacionais, versões de bancos de dados, etc.), diferentes tipos e recursos de hardware (tal como aproveitar um número maior de processadores ou memória). Além destes, fatores como idioma ou a facilidade para se criar ambientes de testes

devem ser considerados como características de portabilidade. Suas sub características são:

- Adaptabilidade: representando a capacidade do software se adaptar a diferentes ambientes sem a necessidade de ações e configurações adicionais.
- Capacidade para ser Instalado: identifica a facilidade com que se pode instalar o sistema em um novo ambiente.
- Coexistência: mede o quão facilmente um software convive com outros instalados no mesmo ambiente.
- Capacidade para Substituir: representa a capacidade que o sistema tem de substituir outro sistema especificado, em um contexto de uso e ambiente específicos. Este atributo interage tanto com adaptabilidade quanto com a capacidade para ser instalado.

APÊNDICE B – CRITÉRIOS DE PONTUAÇÃO DE SUBCARACTERÍSTICAS E CARACTERÍSTICAS

As características de Qualidade descritas no item 2 devem ser avaliadas no início do projeto para ajudar a escolher o ND.

Os critérios de pontuação definidos pelas normas ISO são:

- Três pontos para as subcategorias que atendem satisfatoriamente os requisitos de qualidade (excelente, bom e razoável).
- Um ponto para as subcategorias com pontuação Insatisfatória.

Esses critérios de pontuação serão utilizados como base para a avaliação que pode ser feita por métricas, da seguinte forma: Para cada métrica, os NDs que apresentarem:

- A melhor pontuação da métrica: recebem três pontos para cada subcategoria de qualidade relacionada à métrica (e.g. Tabela C.2).
- As piores pontuações da métrica: recebem um ponto para cada subcategoria de qualidade relacionada à métrica.

Em caso de empate dos melhores, os empatados recebem três pontos. Analogamente, os piores empatados recebem um ponto.

Ao final da pontuação de métricas, para cada subcategoria, serão somados os pontos. Um maior somatório de pontos de subcategorias, entre os NDs avaliados, indica uma melhor pontuação para uma Categoria, que recebe três pontos. Os menores somatórios, um ponto.

APÊNDICE C – AVALIAÇÃO DE CARACTERÍSTICAS DA MANUTENIBILIDADE

C.1 Introdução

Métricas de Software podem ser utilizadas para a avaliação de algumas características de Qualidade. Existem métricas para avaliação de código estruturado e de código Orientado a Objeto. Serão utilizadas somente as métricas de código estruturado, por se tratar da técnica de codificação comum entre os códigos analisados.

Diversos tipos de métricas foram criadas para avaliar a qualidade dos softwares. Algumas delas medem a complexidade do software e características da linguagem de programação, relacionados à Manutenibilidade.

C.1.1 Métrica de Complexidade Ciclométrica de McCabe

M McCabe desenvolveu uma métrica que permite aos desenvolvedores identificar módulos difíceis de testar ou manter. Como resultado, ele desenvolveu uma métrica de software que iguala a complexidade ao número de decisões em um programa. Os desenvolvedores podem usar essa medida para determinar quais módulos de um programa são excessivamente complexos e precisam ser recodificados. A métrica também pode ser usada para definir o número mínimo de casos de teste necessários para testar adequadamente os caminhos do programa. Os valores de referência são exibidos na Tabela C.1, abaixo:

Tabela C.1 - Valores de referência da Complexidade Ciclométrica de McCabe.

Complexidade	Avaliação
1-10	Método simples. Baixo risco.
11-20	Método razoavelmente complexo. Moderado risco.
21-50	Método muito complexo. Elevado risco.
51-N	Método de altíssimo risco e bastante instável.

Fonte: Produção dos autores.

C.1.2 Métricas de Software da RADC

A metodologia RADC consiste em elementos métricos, métricas, critérios e fatores. As pontuações dos elementos de métrica são combinadas para formar uma pontuação de métrica e as pontuações de métrica são combinadas para formar uma pontuação de critério. As pontuações de critérios são então combinadas de várias maneiras para produzir várias pontuações de fatores de qualidade. Elementos métricos são as medidas quantitativas de objetos de software. Métricas são os detalhes orientados ao software das características do software. Critérios são as características orientadas ao software que contribuem para vários tipos de qualidade.

C.2 Metodologia

A metodologia consistiu em utilizar métricas de software, baseadas na metodologia RADC, na qual todas as sub características relacionadas à métrica podem ser pontuadas negativamente ou positivamente. Essa relação está descrita na seção C.2.1. O cálculo das métricas é avaliado sobre os códigos de versões dos modelos que se deseja avaliar.

C.2.1 Métricas

As métricas abaixo foram selecionadas para avaliar as sub características de Manutenibilidade. O método de pontuação de métricas está descrito no Apêndice B deste, onde aqui se destaca os critérios de pontuação. A Tabela C.2 exibe todas as métricas e suas descrições utilizadas na avaliação (colunas “Métrica” e “Descrição”), as sub características que são impactadas positivamente e negativamente, com o aumento do valor da métrica (colunas “Impactos Positivos” e “Impactos Negativos”). A unidade de medida da métrica está disposta na última coluna “Unidade de medida”. As ferramentas utilizadas para o cálculo de cada métrica estão dispostas na coluna "Ferramentas".

Tabela C.2 - Métricas utilizadas na avaliação da Manutenibilidade.

Métrica	Descrição	Impactos Positivos	Impactos Negativos	Ferramentas	Unidade de medida
Comprimento de código	Essa é uma medida do tamanho de um programa. Geralmente, quanto maior o tamanho do código de um componente, mais complexo e sujeito a erros o componente é. O comprimento de código tem mostrado ser uma das métricas mais confiáveis para prever a propensão a erros em componentes	-	Analisabilidade, Modificabilidade, Estabilidade, Testabilidade	FortranAnalyser	Linhas de código
Complexidade Ciclométrica Média (McCabe)	Permite aos desenvolvedores identificar módulos difíceis de testar ou manter. Calcula o número de decisões em um programa, que pode ser usado para determinar quais módulos de um programa são excessivamente complexos e precisam ser recodificados.	-	Analisabilidade, Modificabilidade, Reusabilidade, Estabilidade, Testabilidade	FortranAnalyser	Ver tabela C.1
Documentação total	Mede a porcentagem de código comentado (todo o código) (vazios e "!" sem continuação estão excluídos).	Analisabilidade, Modificabilidade	-	Check.py	Razão
Documentação de rotinas	Mede a porcentagem rotinas comentadas. Facilita o entendimento do código e registra alterações históricas.	Analisabilidade, Modificabilidade, Reusabilidade, Testabilidade	-	FortranAnalyser	Razão
Documentação de arquivos	Mede a porcentagem arquivos com comentários no início. Facilita o entendimento do código e registra alterações históricas.	Analisabilidade, Modificabilidade	-	FortranAnalyser	Razão
Documentação de estruturas de controle	Mede a porcentagem de comentários em estruturas de controle. Facilita o entendimento do código.	Analisabilidade, Modificabilidade	-	FortranAnalyser	Razão
Tamanho médio das rotinas	Tamanho médio em linhas de rotinas, que afeta a compreensibilidade e Manutenibilidade	-	Analisabilidade, Modificabilidade, Reusabilidade, Estabilidade, Testabilidade	Check.py	Linhas de código
Tamanho médio dos módulos	Dado importante. Quando comparado com o número médio das rotinas, se menor, vai indicar que as rotinas não estão encapsuladas em módulos.	-	Analisabilidade, Modificabilidade, Reusabilidade, Estabilidade, Testabilidade	Check.py	Linhas de código
Tamanho médio do nome das variáveis	Considera-se rotinas e funções	Analisabilidade, Modificabilidade, Testabilidade	-	Check.py	Quantidade de caracteres
Razão de only em uses	A falta do only é problemática, pois propaga todas as variáveis para a estrutura e comumente é fonte de bugs.	Analisabilidade, Modificabilidade, Estabilidade	-	Check.py	Razão

(Continua)

Métrica	Descrição	Impactos Positivos	Impactos Negativos	Ferramentas	Unidade de medida
Razão de only em uses	A falta do only é problemática, pois propaga todas as variáveis para a estrutura e comumente é fonte de bugs.	Analisabilidade, Modificabilidade, Estabilidade	-	Check.py	Razão
Razão de "goto" e "continue" por laço	O uso de gotos e continues é desaconselhado, pois torna o código incompreensível e de difícil manutenção. Idealmente deve ser zero.	-	Analisabilidade, Modificabilidade, Estabilidade.	Check.py	Razão
Razão de "exit" e "cycle" por laço	O baixo uso de exit e cycle indica estruturas de laços mal formadas e podem apontar para códigos "macarrônicos"	Analisabilidade, Modificabilidade, Estabilidade	-	Check.py	Razão
Razão do uso de "implicit	Todas as rotinas e módulos deveriam ter "implicit" pois impedem erros por variáveis não declaradas e bugs potenciais. Idealmente deve ser 100%	Analisabilidade, Modificabilidade, Estabilidade	-	Check.py	Razão
Total de "equivalence" ou "common"	Equivalence ou common não são recomendados. Indicam estruturas de codificação antigas e riscos de bugs não mapeados. Essas duas keywords devem ser proibidas pois levam a erros graves quando partes do código são modificadas e vão afetar outras.	-	Modificabilidade, Estabilidade	Check.py	Soma de linhas com palavra chave
Profundidade média de laços	Mede a média de linhas em laços. Laços muito grandes devem ser evitados pois são de baixa compreensão	-	Analisabilidade, Modificabilidade, Reusabilidade, Estabilidade, Testabilidade	Check.py	Número de linhas
Aninhamento médio de laços	Número de laços aninhados médio. Grandes aninhamentos indicam complexidade.	-	Analisabilidade, Modificabilidade, Reusabilidade, Estabilidade, Testabilidade	Check.py	Número de laços aninhados médio
Fan-in	Média de chamadas por subrotina. Número de vezes que a mesma subrotina é chamada. Números altos indicam que erros podem ser corrigidos em área comum facilitando a manutenção, mas que pode afetar a estabilidade	Analisabilidade, Modificabilidade, Reusabilidade, Testabilidade	Estabilidade	FortranMakeUtils	Número de chamadas
Fan-out	Número de rotinas que são chamadas por cada rotina (digamos, X). Um valor alto para fan-out sugere que a complexidade geral do X pode ser alta, devido a complexidade da lógica de controle necessária para coordenar as rotinas chamadas		Modificabilidade, Reusabilidade, Estabilidade, Testabilidade	FortranMakeUtils	Número de chamadas
Aninhamento médio de rotinas	Um número alto pode indicar que a complexidade da rotina X pode ser alta, devido a complexidade da lógica de controle necessária para coordenar as rotinas chamadas		Modificabilidade, Reusabilidade, Estabilidade, Testabilidade	FortranMakeUtils	Número de rotinas aninhadas

Fonte: Produção dos autores.

C.3 Ferramentas para avaliação da Manutenibilidade

As ferramentas necessárias para a avaliação estão versionadas em https://github.com/monanadmin/monan/tree/main/tools/qas_eval. Algumas ferramentas foram adaptadas, como o *FortranAnalyser*, para possibilitar o uso sem interface gráfica. Outras utiliza-se somente o código binário, como no caso do *fortran-src*.

Uma imagem do *container singularity* (*qas_eval.sif*) foi criada para armazenar e executar as ferramentas binárias (*fortran-src* e *FortranAnalyser.jar*) sobre uma distribuição Linux compatível com as ferramentas. As ferramentas binárias estão disponíveis na pasta “/home/qas_files/tools/” dentro da imagem.

Ferramentas com código fonte no repositório:

- *check.py* (https://github.com/monanadmin/monan/tree/main/tools/qas_eval).
Programa desenvolvido para ler estatísticas geradas pelo programa (binário) *fortran-src*. Também contabiliza outras estatísticas através da análise direta do código, gerando um relatório sintetizado de métricas.
- *FortranAnalyser* (<http://fortrananalyser.ephyslab.uvigo.es/>).
Ferramenta escrita em java, modificada para ser executada em linha de comando. Gera um relatório em PDF contendo estatísticas e escores, que são gerados por arquivo e em forma sintética para todo o conjunto de arquivos.
- *FortranMakeUtils* (<https://github.com/deniseiras/fortranMakeUtils>).
Ferramenta desenvolvida inicialmente para gerar a hierarquia de dependência de módulos e gerar os makefiles automaticamente. As dependências geradas entre chamadas de rotinas foi utilizada no *check.py* para calcular métricas como Fan-In e Fan-Out.

Ferramentas com código binário:

- *fortran-src* (<https://github.com/camfort/fortran-src>).
Ferramenta escrita na linguagem Haskell que gera diversas estatísticas do código.

- FortranAnalyser.jar.

Binário gerado pelo código fonte. Para gerar o binário a partir do código fonte, baixe ou instale a ferramenta maven, entre na pasta principal do *FortranAnalyser* (onde se encontra o pom.xml) e execute "mvn clean install".

C.4 Geração das métricas

1. Use o módulo *singularity* no servidor Rattler (a) ou instale em outro local desejado (b). Obs: A versão do singularity disponível para Ubuntu 18.04 (2.6.1-dist - via apt install) não funciona com o pull do passo 3. A instalação manual de versões mais recentes, via passos de <https://sylabs.io/guides/3.0/user-guide/installation.html> , requer pacotes que não estão disponíveis na versão 18.04 do Ubuntu. Recomenda-se alguma das formas abaixo:

a) module load singularity # máquina Rattler

b) sudo apt install -y singularity-container

2. Crie uma pasta de nome qas_eval e entre nela

a) mkdir qas_eval;ND qas_eval

3. Baixe a imagem do singularity na pasta criada, com uma das opções abaixo. Use a opção b) caso não consiga usar o pull (ex. Ubuntu 18.04) em a):

a) singularity pull qas_eval.sif

library://denis.eiras/monan/qas_eval:dev

b) Clique no link de download disponível na página https://cloud.sylabs.io/library/denis.eiras/monan/qas_eval e baixe a imagem para a pasta atual (qas_eval) usando o nome qas_eval.sif

4. Baixe o código inteiro do monan (será usada a subpasta tools):

a) git clone <https://github.com/monanadmin/monan.git>

5. Execute o script principal. Todas as subpastas dos NDs dos códigos em monan/codigos_originais, serão avaliadas:

a) monan/tools/qas_eval/run_eval.sh

O último passo executará as ferramentas para os códigos de NDs e gerará os seguintes relatórios:

- QualityReport_[MODELO].pdf: Relatório gerado pela ferramenta *FortranAnalyser*. As estatísticas sintetizadas estão no final do arquivo
- Check_Report_[MODEL].txt : Relatório gerado pela ferramenta Check.py.

Os relatórios informam os valores das métricas obtidas por cada ND, que serão usados para pontuar as características de Qualidade mapeadas para cada métrica (Tabela C.2).