UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

MDAPSP - Uma arquitetura modular distribuída para auxílio à predição de estruturas de proteínas

Edvard Martins de Oliveira

Tese de Doutorado do Programa de Pós-Graduação em Ciências de
Computação e Matemática Computacional (PPG-CCMC)

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Edvard Martins de Oliveira

MDAPSP - Uma arquitetura modular distribuída para auxílio à predição de estruturas de proteínas

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Júlio César Estrella

USP – São Carlos
Junho de 2018

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

O48m Oliveira, Edvard Martins de
 MDAPSP - Uma arquitetura modular distribuída para
 auxílio à predição de estruturas de proteínas / Edvard
 Martins de Oliveira; orientador Júlio César
 Estrella. -- São Carlos, 2018.
 202 p.

 Tese (Doutorado - Programa de Pós-Graduação em
 Matemática) -- Instituto de Ciências Matemáticas e
 de Computação, Universidade de São Paulo, 2018.

 1. Sistemas distribuídos. 2. Arquitetura
 orientada a serviços. 3. Predição de estruturas de
 proteínas. 4. Gateways científicos. 5. WorkflowSim.
 I. Estrella, Júlio César, orient. II. Título.

Edvard Martins de Oliveira

**MDAPSP - A modular distributed architecture to support
protein structure prediction**

Doctoral dissertation submitted to the Institute of
Mathematics and Computer Sciences – ICMC-USP, in
partial fulfillment of the requirements for the degree of
the Doctorate Program in Computer Science and
Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and
Computational Mathematics

Advisor: Prof. Dr. Júlio César Estrella

USP – São Carlos
June 2018

*Dedico esse trabalho àqueles que lutam pela ciência e difusão do conhecimento.
Aos educadores, tanto os das salas de aula quanto os dos lares.*

AGRADECIMENTOS

Agradeço primeiramente à minha família pela paciência, compreensão e apoio desde sempre. Ao prof. Júlio pelas oportunidades e pela orientação nessa caminhada acadêmica. Agradeço aos demais professores que contribuíram para minha formação e desenvolvimento dessa pesquisa, àqueles que integram o LaSDPC, ao prof. Alexandre Delbem pela colaboração e ao prof. Stephan Reiff-Marganiec pela orientação no período na Universidade de Leicester - Inglaterra. Aos meus amigos, dos quais destaco quatro que auxiliaram diretamente na construção dessa tese, Alexandre Defelicibus, Fausto Costa, Mário Pardo e Juliano Ferreira. Aos companheiros de laboratório pelas conversas, risadas e trocas de conhecimento. Aos amigos com quem convivi nesses anos, compartilhando teto, experiências, alegrias e dificuldades. Aos amigos de toda a vida de Machado, Poços de Caldas, São Carlos, Leicester e demais lugares que tive o prazer de conhecer. Muito obrigado.

*“O que sabemos é uma gota,
o que ignoramos é um oceano.”
(Isaac Newton)*

RESUMO

OLIVEIRA, E.M. MDAPSP - Uma arquitetura modular distribuída para auxílio à predição de estruturas de proteínas. 2018. 202 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2018.

A predição de estruturas de proteínas é um campo de pesquisa que busca simular o enovelamento de cadeias de aminoácidos de forma a descobrir as funções das proteínas na natureza, um processo altamente dispendioso por meio de métodos *in vivo*. Inserida no contexto da Bioinformática, é uma das tarefas mais computacionalmente custosas e desafiadoras da atualidade. Devido à complexidade, muitas pesquisas se utilizam de *gateways* científicos para disponibilização de ferramentas de execução e análise desses experimentos, aliado ao uso de *workflows* científicos para organização de tarefas e disponibilização de informações. No entanto, esses *gateways* podem enfrentar gargalos de desempenho e falhas estruturais, produzindo resultados de baixa qualidade. Para atuar nesse contexto multifacetado e oferecer alternativas para algumas das limitações, esta tese propõe uma arquitetura modular baseada nos conceitos de *Service Oriented Architecture* (SOA) para oferta de recursos computacionais em *gateways* científicos, com foco nos experimentos de *Protein Structure Prediction* (PSP). A Arquitetura Modular Distribuída para auxílio à Predição de Estruturas de Proteínas (MDAPSP) é descrita conceitualmente e validada em um modelo de simulação computacional, no qual se pode identificar suas capacidades, detalhar o funcionamento de seus módulos e destacar seu potencial. A avaliação experimental demonstra a qualidade dos algoritmos propostos, ampliando a capacidade de atendimento de um *gateway* científico, reduzindo o tempo necessário para experimentos de predição e lançando as bases para o protótipo de uma arquitetura funcional. Os módulos desenvolvidos alcançam boa capacidade de otimização de experimentos de PSP em ambientes distribuídos e constituem uma novidade no modelo de provisionamento de recursos para *gateways* científicos.

Palavras-chave: Arquiteturas orientadas a serviço, predição de estruturas de proteínas, *gateways* científicos, *workflows* científicos, WorkflowSim.

ABSTRACT

OLIVEIRA, E.M. **MDAPSP - A modular distributed architecture to support protein structure prediction**. 2018. 202 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2018.

PSP is a scientific process that simulates the folding of amino acid chains to discover the function of a protein in live organisms, considering that its an expensive process to be done by *in vivo* methods. PSP is a computationally demanding and challenging effort in the Bioinformatics state-of-the-art. Many works use scientific gateways to provide tools for execution and analysis of such experiments, along with scientific workflows to organize tasks and to share information. However, these gateways can suffer performance bottlenecks and structural failures, producing low quality results. With the goal of offering alternatives to some of the limitations and considering the complexity of the topics involved, this thesis proposes a modular architecture based on SOA concepts to provide computing resources to scientific gateways, with focus on PSP experiments. The *Modular Distributed Architecture to support Protein Structure Prediction* (MDAPSP) is described conceptually and validated in a computer simulation model that explain its capabilities, detail the modules' operation and highlight its potential. The performance evaluation presents the quality of the proposed algorithms, a reduction of response time in PSP experiments and prove the benefits of the novel algorithms, establishing the basis for a prototype. The new modules can optimize the PSP experiments in distributed environments and are a innovation in the resource provisioning model for scientific gateways.

Keywords: Service oriented architectures, protein structure prediction, scientific gateways, scientific workflows, WorkflowSim.

LISTA DE ILUSTRAÇÕES

Figura 1 – Ilustração dos passos do processo de predição de proteína.	27
Figura 2 – Diagrama das referências da tese.	34
Figura 3 – Histograma trabalhos referenciados na tese.	34
Figura 4 – Ilustração de uma estrutura de proteína.	38
Figura 5 – Ilustração das estruturas que as proteínas podem assumir	39
Figura 6 – Ilustração do processo de <i>folding</i> de uma proteína.	40
Figura 7 – Pilha conceitual dos <i>Web Services</i>	45
Figura 8 – Níveis de abstração em <i>Cloud</i> de acordo com o tipo de estrutura.	48
Figura 9 – Tipos de <i>workflow</i> base para construção de grafos complexos.	51
Figura 10 – DAG do <i>workflow</i> científico <i>Epigenomics</i> , usado em pesquisas genéticas. . .	52
Figura 11 – Fluxograma do ciclo de vida de um <i>workflow</i> científico.	54
Figura 12 – Diagrama de módulos do <i>WorkflowSim</i>	61
Figura 13 – Mensagens no <i>WorkflowSim</i> . Fase de inicialização	62
Figura 14 – Fluxo de mensagens no <i>WorkflowSim</i>	62
Figura 15 – Interface do Koala, implementação baseada no Galaxy	84
Figura 16 – Diagrama do Koala - Primeiros passos de um experimento.	85
Figura 17 – Diagrama do Koala - Preparação de dados.	85
Figura 18 – Diagrama do Koala - Predição.	86
Figura 19 – Diagrama do Koala - Minimização.	86
Figura 20 – Diagrama do Koala - Disponibilização de dados.	87
Figura 21 – Diagrama de sequência do planejamento de um experimento no Koala. . . .	88
Figura 22 – Organização da estrutura do Koala em instalação local.	90
Figura 23 – Gráficos com o resultado da regressão exponencial. Primeira amostra. . . .	95
Figura 24 – Gráficos com o resultado da regressão exponencial. Segunda amostra. . . .	96
Figura 25 – Gráficos com o resultado da regressão exponencial. Terceira amostra. . . .	97
Figura 26 – <i>Workflow</i> do 2PG-Mono, para a proteína 1VII.	99
Figura 27 – <i>Workflow</i> do 2PG-Mono, para a proteína 1AGY.	100
Figura 28 – <i>Workflow</i> do 2PG-Mono, para a proteína 1EOD.	101
Figura 29 – Gráficos do monitoramento do Koala com uso do Ganglia	103
Figura 30 – Organograma dos módulos gerais de <i>gateways</i> científicos.	105
Figura 31 – Ilustração do ambiente heterogêneo de recursos da infraestrutura.	106
Figura 32 – Levantamento dos tamanhos das cadeias processadas no último CASP, em 2016.	108

Figura 33 – Gráficos com os resultados dos modelos de regressões.	111
Figura 34 – Gráfico dos resultados gerados pelo <i>Decision Maker</i> . Primeiro cenário. . . .	112
Figura 35 – Gráfico dos resultados gerados pelo <i>Decision Maker</i> . Segundo cenário. . . .	113
Figura 36 – Gráfico dos resultados gerados pelo <i>Decision Maker</i> . Terceiro cenário. . . .	113
Figura 37 – Gráfico dos resultados gerados pelo <i>Decision Maker</i> . Quarto cenário. . . .	114
Figura 38 – Ilustração abstrata dos módulos da MDAPSP e sua contribuição em <i>e-science</i> .118	
Figura 39 – Organização em camadas dos SWMS	120
Figura 40 – Organização das infraestruturas de HPC.	122
Figura 41 – Organização dos módulos da MDAPSP.	123
Figura 42 – Organização completa dos módulos da MDAPSP	124
Figura 43 – Diagrama de sequência de um experimento na MDAPSP.	126
Figura 44 – Ilustração dos módulos da MDAPSP considerados no modelo simulado. . .	130
Figura 45 – Ilustração do módulo de provisionamento da MDAPSP no modelo simulado.	133
Figura 46 – Ilustração dos módulos cliente e PDPAT modelados no <i>WorkflowSim</i>	136
Figura 47 – Diagrama de classes das extensões implementadas para o desenvolvimento da MDAPSP.	138
Figura 48 – Ilustração dos módulos centrais modelados em simulação.	139
Figura 49 – Comparação de todos os cenários avaliados.	149
Figura 50 – Resultados para o <i>workflow</i> 1VII, em três cenários de avaliação.	150
Figura 51 – Resultados para o <i>workflow</i> 1AGY, em três cenários de avaliação.	152
Figura 52 – Resultados para o <i>workflow</i> Montage em três cenários de avaliação.	153
Figura 53 – Taxa de utilização das VMs, com 3 <i>datacenters</i>	154
Figura 54 – Taxa de utilização das VMs, com 4 <i>datacenters</i>	155
Figura 55 – Taxa de utilização das VMs, com 5 <i>datacenters</i>	155
Figura 56 – Tarefas do <i>workflow</i> relacionadas à sua estrutura.	156
Figura 57 – Tarefas do <i>workflow</i> relacionadas à sua estrutura.	157
Figura 58 – Tarefas do <i>workflow</i> relacionadas à sua estrutura.	157
Figura 59 – Comparação sobreposta dos <i>Workflows</i>	158

LISTA DE CÓDIGOS-FONTE

Código-fonte 1 – Configuração do <i>cluster</i> Cosmos	131
Código-fonte 2 – Configuração de Máquina Virtual no <i>cluster Cosmos</i>	131
Código-fonte 3 – Instanciação dos <i>datacenters</i>	132
Código-fonte 4 – Seleção dos algoritmos de Escalonamento e Planejamento	133
Código-fonte 5 – Vinculação dos <i>datacenters</i> ao escalonador.	134

LISTA DE TABELAS

Tabela 1 – Princípios de <i>design</i> de orientação a serviço.	43
Tabela 2 – Comparação de infraestruturas de computação	46
Tabela 3 – Comparação de valores entre os provedores de nuvem.	49
Tabela 4 – Exemplos de <i>gateways</i> científicos.	58
Tabela 5 – Amostra de trabalhos em PSP e suas necessidades principais.	69
Tabela 6 – Tabela comparativa dos SWMS mais difundidos.	70
Tabela 7 – Amostra de trabalhos em <i>gateways</i> científicos e suas necessidades principais.	75
Tabela 8 – Amostra do levantamento de trabalhos relacionados	80
Tabela 9 – Especificações dos nós do <i>Cluster Andromeda</i>	89
Tabela 10 – Especificações dos nós do <i>Cluster Halley</i>	89
Tabela 11 – Capacidade da Máquinas do Koala	89
Tabela 12 – Parâmetros para o experimento inicial.	91
Tabela 13 – Principais problemas encontrados no Galaxy Koala	92
Tabela 14 – Fatores de avaliação do procedimento <i>Build Population</i>	93
Tabela 15 – Variáveis de resposta do cálculo de regressão.	94
Tabela 16 – Amostra das variáveis de saída das regressões lineares.	94
Tabela 17 – Exemplo de saída do Gprof com as funções de um preditor.	98
Tabela 18 – Desenho de experimentos de teste do Ganglia Monitor	102
Tabela 19 – Parâmetros para os experimentos preliminares.	108
Tabela 20 – Capacidades de máquina de cada classe	109
Tabela 21 – Amostra dos resultados obtidos.	109
Tabela 22 – Resultados da regressão.	110
Tabela 23 – Configuração de <i>links</i> de rede entre componentes da arquitetura.	137
Tabela 24 – Condições estáticas do escalonador MDSA.	140
Tabela 25 – Especificações da máquina tipo <i>workstation</i>	144
Tabela 26 – Especificações dos nós do <i>Cluster Cosmos</i>	144
Tabela 27 – Especificações dos nós do <i>Cluster Andromeda</i>	144
Tabela 28 – Especificações dos nós do <i>Cluster Halley</i>	145
Tabela 29 – Especificações da Plataforma Bare Metal	145
Tabela 30 – Parâmetros de cenários para carga de trabalho	146
Tabela 31 – Parâmetros dinâmicos do MDSA.	146
Tabela 32 – Desenho de experimentos da arquitetura	147
Tabela 33 – Combinação de Fatores e Níveis dos Experimentos	148

Tabela 34 – Cenários de avaliação. 148

LISTA DE ABREVIATURAS E SIGLAS

aaS	<i>-as-a-Service</i>
2PG	<i>Prot-Pred-Gromacs</i>
3D	<i>Tridimensional</i>
ACCS	<i>Army Command and Control System</i>
ANFIS	<i>Adaptative Neural Fuzzy Inference System</i>
API	<i>Application Programming Interface</i>
APMD	<i>VM Allocation Policy for Multi Datacenters</i>
BPEL	<i>Business Process Execution Language</i>
CASP	<i>Critical Assessment of protein Structure Prediction</i>
CPU	<i>Central Processing Unit</i>
CUDA	<i>Compute Unified Device Architecture</i>
DAG	<i>Directed Acyclic Graph</i>
DAGMan	<i>Directed Acyclic Graph Manager</i>
DAO	<i>Data Access Object</i>
DAX	<i>Directed Acyclic Graph in XML</i>
DCG	<i>Directed Cyclic Graph</i>
DCI	<i>Distributed Computing Infrastructure</i>
DNA	<i>Deoxyribonucleic Acid</i>
EC2	<i>Elastic Cloud Computing</i>
EIPS	<i>Enterprise Integration Patterns</i>
ESB	<i>Enterprise Service Bus</i>
FCFS	<i>First Come First Served</i>
FPGA	<i>Field Programmable Gate Array</i>
GB	<i>GigaByte</i>
Gprof	<i>Gnu Profiler</i>
GPU	<i>Graphics Processing Unit</i>
GROMACS	<i>GRoningem MACHine for Chemical Simulations</i>
GUI	<i>Graphical User Interface</i>
HP	<i>Hidrofóbico Polar</i>
HPC	<i>High-performance Computing</i>
HTML	<i>HyperText Markup Language</i>

HTTP	<i>Hypertext Transfer Protocol</i>
I-TASSER	<i>Iterative Threading ASSEmbly Refinement</i>
IaaS	<i>Infrastructure as a Service</i>
ICMC	Instituto de Ciências Matemáticas e de Computação
IoT	<i>Internet of Things</i>
IP	<i>Internet Protocol</i>
iPFPi	<i>Improving Protein Function Prediction Iteratively</i>
LASDPC	Laboratório de Sistemas Distribuídos e Programação Concorrente
MB	<i>MegaByte</i>
MCBR	<i>Multi-Criteria Biased Randomized</i>
MDAPSP	<i>Modular Distributed Architecture to support the Protein Structure Prediction</i>
MDSA	<i>MultiDatacenter Scheduling Algorithm</i>
MI	Milhões de Instruções
MIPS	<i>Millions of Instructions per Second</i>
MPI	<i>Message Passing Interface</i>
NGS	<i>Next Generation Sequencing</i>
NSGA-II	<i>Non-dominated Sorting Genetic Algorithm-II</i>
PaaS	<i>Platform as a Service</i>
PDB	<i>Protein Data Bank</i>
PDPAT	<i>Protein Data Pattern</i>
PDS	<i>Protein Data Storage</i>
PS	<i>Protein Selector</i>
PSP	<i>Protein Structure Prediction</i>
QoE	<i>Quality of Experience</i>
QoS	<i>Quality of Service</i>
RAM	<i>Random Access Memory</i>
RDS	<i>Results Data Storage</i>
REST	<i>Representational State Transfer</i>
RMSE	<i>Root Mean Squared Error</i>
RNA	<i>Ribonucleic Acid</i>
RO	<i>Research Object</i>
RP	<i>Random Projection</i>
SaaS	<i>Software as a Service</i>
SCC	<i>Single-Chip Cloud Computer</i>
SCUFL	<i>Simple Conceptual Unified Flow Language</i>
SD	Sistema Distribuído
SGCI	<i>Science Gateway Community Institute</i>

SHIWA	<i>SHaring Interoperable Workflows for large-scale scientific simulations on Available DCIs</i>
SLA	<i>Service Level Agreement</i>
SMB	<i>Small and Medium Business</i>
SO	<i>Sistema Operacional</i>
SOA	<i>Service Oriented Architecture</i>
SOAD	<i>Service-Oriented Architecture Decision</i>
SOAP	<i>Simple Object Access Protocol</i>
SPI	<i>Service Provider Interface</i>
SPQoS	<i>Service Provider Quality of Service</i>
SPT	<i>Service Provider Type</i>
STAPL	<i>Standard Template Adaptive Parallel Library</i>
SVM	<i>Support Vector Machines</i>
SVR	<i>Epsilon-Support Vector Regression</i>
SWfMS	<i>Scientific Workflow Management System</i>
SWMS	<i>Scientific Workflow Management System</i>
UDDI	<i>Universal Description, Discovery and Integration</i>
USP	<i>Universidade de São Paulo</i>
VM	<i>Virtual Machine</i>
WEP	<i>Workflow Execution Planner</i>
WLSA	<i>Workload Length Scheduling Algorithm</i>
WSDL	<i>Web Services Description Language</i>
XML	<i>eXtensible Markup Language</i>

SUMÁRIO

1	INTRODUÇÃO	27
1.1	Contextualização	28
1.2	Motivação e objetivos	31
1.3	Estrutura da tese	33
2	REFERENCIAL TEÓRICO	37
2.1	Considerações iniciais	37
2.2	Predição de Estruturas de Proteínas	37
2.3	Arquitetura Orientada a Serviço	42
2.4	Computação em Nuvem	45
2.5	Workflow Científico	50
2.6	Gateways Científicos	54
2.7	Simulação Computacional	59
2.8	Considerações Finais	63
3	REVISÃO DA LITERATURA	65
3.1	Considerações iniciais	65
3.2	Estado da Arte de PSP	65
3.3	Estado da Arte de <i>gateways</i> científicos	69
3.4	Trabalhos Relacionados	74
3.5	Considerações finais	80
4	ESTUDO EXPLORATÓRIO DO KOALA	83
4.1	Considerações iniciais	83
4.2	O Galaxy Koala	83
4.2.1	<i>Estudo exploratório do Galaxy</i>	87
4.2.1.1	<i>Avaliação funcional</i>	91
4.2.2	<i>Avaliação da geração de dados intermediários</i>	92
4.2.3	<i>Mapeamento da carga de trabalho</i>	96
4.2.4	<i>Monitoramento da Arquitetura</i>	99
4.2.4.1	<i>Estudo de Monitoramento</i>	102
4.3	Decision Maker	103
4.3.1	<i>Algoritmo do Decision Maker</i>	105
4.3.2	<i>Avaliação dos algoritmos de regressão</i>	107

4.3.3	<i>Avaliação do modelo SVR</i>	111
4.4	Considerações Finais	115
5	MDAPSP	117
5.1	Considerações iniciais	117
5.2	A Arquitetura Conceitual	117
5.3	Organização geral	122
5.4	Considerações finais	128
6	MODELO DE SIMULAÇÃO	129
6.1	Considerações iniciais	129
6.2	Ferramentas	129
6.3	Descrição dos algoritmos	129
6.3.1	<i>Workflows de carga de trabalho</i>	135
6.3.2	<i>Delays de Rede</i>	136
6.3.3	<i>Extensão do framework WorkflowSim</i>	137
6.4	Algoritmos de Escalonamento	138
6.4.1	<i>MultiDatacenter Scheduling Algorithm (MDSA)</i>	139
6.4.2	<i>Workload Length Scheduling Algorithm (WLSA)</i>	141
6.5	Considerações finais	142
7	AVALIAÇÃO EXPERIMENTAL	143
7.1	Considerações iniciais	143
7.2	Ambiente de Teste	143
7.3	Desenho de Experimentos	147
7.4	Resultados	147
7.5	Considerações finais	159
8	CONCLUSÃO	161
8.1	Considerações iniciais	161
8.2	Argumentação geral	161
8.3	Contribuições	162
8.4	Publicações	163
8.5	Dificuldades	164
8.6	Trabalhos Futuros	164
REFERÊNCIAS		167

INTRODUÇÃO

A predição de estruturas de proteínas (do inglês PSP) é um problema bem conhecido em Bioinformática. A identificação da formação nativa da proteína torna possível prever a sua função dentro dos organismos, além de auxiliar no desenvolvimento de medicamentos e na compreensão de mecanismos biológicos (DORN *et al.*, 2014). Devido ao seu elevado custo financeiro, diversas técnicas têm sido desenvolvidas em busca de modelos simplificados e que alcancem bons resultados (KALEGARI; LOPES, 2013) (ALM; BAKER, 1999). A Figura 1 mostra uma simulação do processo de *folding* de uma estrutura de proteína, cuja função é encontrar a estrutura tridimensional a partir de uma cadeia de aminoácidos. A manipulação de proteínas é um processo caro e complexo, que enfrenta dificuldades para produzir bons resultados, necessitando de recursos humanos e computacionais especializados, além de longos períodos de experimentação (KUNHI; RAJU; CHIPLUNKAR, 2016).

Figura 1 – Ilustração dos passos do processo de predição de proteína.

Fonte: Ziv e Haran (2009).

PSP é um dos temas de pesquisa mais desafiadores da atualidade (BORNICH *et al.*, 2016). Mesmo com emprego de métodos variados, são observados pequenos ganhos em alguns objetivos (FACCIOLI, 2012). Dentre esforços na tentativa de acelerar os resultados destaca-se o *WeFold*, um *gateway* que reúne as soluções de diversos pesquisadores em um ambiente de trocas de informações, comparação de modelagens e disputa pelos melhores resultados (WEFOLD,

2013). Assim como em PSP, a complexidade e as demandas de procedimentos científicos *in silico* têm crescido de maneira constante, alcançando novos níveis de magnitude nos últimos anos (FARD *et al.*, 2012).

A capacidade interdisciplinar da Ciência da Computação, juntamente com as necessidades de outros campos de pesquisa evoluiu para um novo patamar de colaboração e resolução de problemas avançados (STEIN, 2008), favorecendo o surgimento de *e-science*, que provê a integração de tecnologias, divulgação de métodos, publicação de resultados e compartilhamento de conhecimento (GÖRLACH *et al.*, 2011).

Nesse aspecto, os *gateways* científicos surgem como uma ferramenta poderosa para gerenciamento de ambientes computacionais complexos, como computação de alto desempenho, supercomputadores ou sistemas distribuídos e paralelos (GESING *et al.*, 2017). Eles oferecem meios de acesso e integração para uma comunidade diversificada que se beneficia da capacidade de produzir novas soluções de forma estruturada. A necessidade de padronização e organização de tarefas e procedimentos é atendida com *workflows* científicos, essenciais no processo experimental moderno (OGASAWARA *et al.*, 2013). Nos *workflows* são registrados os passos detalhados de um procedimento, bem como as informações de cada tarefa, tráfego de rede, dependência de dados, prioridade de execução, e etc. Com um *workflow* bem definido é possível avaliar em detalhes um processo de experimento, identificando requisitos e propondo melhorias de desempenho (MATTOSO *et al.*, 2013). Os *gateways* científicos destacam-se pelo crescente interesse e bons resultados produzidos, inclusive em auxiliar usuários inexperientes na manipulação de experimentos.

Em busca de métodos que ofereçam flexibilidade, robustez e capacidade de evolução na construção desses *gateways*, as capacidades de Arquiteturas Orientadas a Serviço (do inglês SOA) despontam como uma alternativa viável. A composição de aplicações e sistemas colaborativos é um dos objetivos de SOA, que permite o desenvolvimento e publicação de soluções de forma incremental e fracamente acopladas. Serviços pequenos podem se conectar para a formação de ambientes maiores, nos quais a descoberta de recursos são fundamentais (CLASSE *et al.*, 2016).

1.1 Contextualização

A determinação da estrutura proteica é altamente dispendiosa (financeiramente) em ambientes reais, por meio de técnicas como cristalografia, microscopia eletrônica ou ressonância magnética nuclear, além de ser um processo que leva longo tempo de execução (GÜNTERT, 2004). PSP oferece uma alternativa de descoberta dessas proteínas por meio de simulações computacionais, de forma a reduzir esses custos. Entretanto, os procedimentos são complexos e as soluções incipientes, desafiando os pesquisadores a desenvolver novas abordagens (DORN *et al.*, 2014).

PSP envolve processamento pesado, experimentos longos e sequência variável de passos.

Pesquisas que exigem análises detalhadas, visualização dos passos e reprodutibilidade se utilizam de *workflows* científicos para especificar e automatizar a execução de experimentos, com especial importância em um Sistema Distribuído (SD) (QASHA; CAIA; WATSON, 2016). *Workflows* são capazes de gerar um modelo que define as estruturas das tarefas computacionais, assim como a gestão dos processos. Profissionais de diversas disciplinas podem contribuir para a análise das tarefas e a definição de suas estruturas, sendo úteis em diversos contextos (DEELMAN *et al.*, 2009).

A fim de facilitar o acesso a mecanismos computacionais avançados, os *gateways* científicos são constituídos de um conjunto de ferramentas, aplicativos e dados, integrados em uma interface comum. O objetivo dos *gateways* é permitir a realização de experimentos, mesmo que o usuário não tenha habilidades no ambiente computacional. Um primeiro passo nesse sentido é o uso de sistemas de gerenciamento de *workflow* (do inglês *Scientific Workflow Management System* (SWMS)) para permitir a reprodutibilidade e tornar o processo de execução mais simples (SANDES; BOUKERCHE; MELO, 2016). Exemplos de SWMS são: Taverna (WOLSTENCROFT *et al.*, 2013), gUse (BALASKO; FARKAS; KACSUK, 2013), GenePattern (REICH *et al.*, 2006), IOBio (MILLER *et al.*, 2014), Epiviz (CHELARU *et al.*, 2014), Pegasus (DEELMAN *et al.*, 2015) e Swift (WILDE *et al.*, 2011). Cada um dos SWMS citados tem aplicações e finalidades distintas, sendo que alguns são multiúso. Esses sistemas geralmente são integrados com bancos de dados para armazenar os passos dos *workflows*, dados de entrada e resultados de processamento (GESING *et al.*, 2016).

Um sistema complexo como um *gateway* científico deveria ter capacidade para atender qualquer tipo de solicitação, independentemente da complexidade do processamento, porém, esses sistemas frequentemente apresentam limitações que interferem em seu desempenho. Como esses *gateways* oferecem uma solução de alto nível, o usuário não tem acesso às informações das máquinas que o hospedam (GESING *et al.*, 2017), situação que torna problemas de simples resolução em limitações do ambiente. E como os experimentos têm requisitos variáveis, pode haver subutilização dos recursos em um cenário e sobrecarga em outro, conforme o modelo de dados, dificultando a constituição de infraestruturas de máquinas.

Dos *gateways* científicos disponíveis, um dos mais proeminentes é o Galaxy (AFGAN *et al.*, 2016), voltado para aplicações em Bioinformática, tem grande aceitação e é largamente utilizado. Por ser de código aberto e permitir a adição de novos componentes, é adaptável a diversas necessidades e aplicações, provendo uma flexibilidade que atende uma população ampla de pesquisadores (AFGAN *et al.*, 2016). A simplicidade de uso e a transparência devem ser uma das características fundamentais desses sistemas, para facilitar o acesso e a manipulação. O Galaxy tem a capacidade de atender diversos usuários, entretanto carece de modelos de disponibilização de recursos de forma eficiente (GOECKS *et al.*, 2010). Esse e outros *gateways* podem se beneficiar de sistemas modulares de organização encontradas em SOA, bem como de alternativas heterogêneas de infraestrutura.

Com um paradigma bem estabelecido, SOA oferece ferramentas para o desenvolvimento independente e incremental de aplicações, favorecendo a integração entre sistemas (BENHADDI; BAINA; HASSAN, 2013). Permite o compartilhamento de dados entre diferentes estruturas por meio da troca de mensagens e auxilia na modularização de sistemas grandes, como jogos de múltiplos participantes, por exemplo (ARSLAN, 2012). Mecanismos de SOA podem ser eficientes em ambientes com diversas trocas de contexto, comunicação e camadas de rede (CHEN *et al.*, 2010) (HUANG, 2016). Modelagens cada vez mais eficientes têm sido disponibilizadas tanto para analisar quanto para atuar em ambientes heterogêneos, cuidando de aspectos como *Quality of Service* (QoS), integração e modularização (BAHRAMI; SINGHAL, 2015) (ELIASSON *et al.*, 2013) (LU; KOGA, 2015). A fácil integração de múltiplas tecnologias em SOA proporciona inovações e evoluções. A partir de uma arquitetura de módulos integráveis e disponíveis via modelo de serviços, é possível prover recursos computacionais de forma otimizada e oferecer variações no modelo de uma infraestrutura.

A capacidade requerida em um experimento computacional é difícil de ser determinada de antemão. O problema com a oferta de recursos é a busca pela melhor configuração, com o menor investimento e sobrecarga mínima (KIMOVSKI *et al.*, 2016). Conforme a característica de um experimento, uma máquina do tipo *workstation* é suficiente, na qual o usuário tem o controle do *hardware*, não tem limitações com a largura de rede e responsabiliza-se pelos dados. Entretanto, a variedade de procedimentos que podem ser atendidos por tais máquinas é restrita (GARZA; AICHELER; KOHLBACHER, 2016). Uma alternativa comum são *clusters* de computadores, uma infraestrutura confiável para realizar experimentos diversos (DEAN; GHEMAWAT, 2008), atendendo acesso concorrente e processamentos complexos (MACEDO *et al.*, 2013). As vantagens de um *cluster* são a gestão local, a liberdade de configuração, o armazenamento confiável e as transferências de dados rápidas, oferecendo recursos de alta capacidade por um preço reduzido. Os *clusters* por sua vez ficam limitados ao número de nós e à capacidade instalada.

Por sua vez, a computação em nuvem provê disponibilização e consumo de serviços sob demanda (CHARD *et al.*, 2015), sendo reconhecida por sua capacidade escalável, faixas de atendimento diferenciadas e gerenciamento oferecido pelos provedores (GUPTA; AZHARUDDIN; JANA, 2016). Contudo, é preciso lidar com restrições de prazo, custos, recursos e limitações de interoperabilidade (SENTURK *et al.*, 2016). A localização física dos servidores também pode ser problemática, já que dados sensíveis têm limitações legais de armazenamento em países estrangeiros (BUYA *et al.*, 2009). Impedir a migração de dados, controle contra falhas de conexão, baixa velocidade de transmissão, ambientes inseguros e limites de armazenamento são desafios em aberto (KSHETRI; MURUGESAN, 2013) (MEIYAN, 2013) (CRANDELL, 2017). O supracitado *WeFold* é um exemplo de sistema disponibilizado via servidores em nuvem, o que é uma boa escolha para grupos de pesquisa com poder de investimento ou devidamente patrocinados. É sabido que instâncias mais poderosas de nuvem têm preço elevado e que experimentos em Bioinformática podem levar várias semanas para concluir (FARD; FAHRINGER; PRODAN,

2013). Em PSP o uso de recursos de forma ponderada é indispensável.

Essas abordagens mostram a variedade de esforços que podem ser empregados na resolução de problemas de provisionamento e gerenciamento de experimentos. Os métodos computacionais são vários, as metodologias divergem e a aplicação pode ter particularidades que dificultam a reutilização de ferramentas. Além disso, os meios de processamento podem variar dentro de um mesmo contexto, o que pode ser desafiador. Os *gateways* científicos oferecem meios para suavizar esse contexto, mas esses também requerem planejamento cuidadoso, análise adequada e distribuição de carga para constituir um ambiente produtivo. Em um mundo onde as plataformas são cada vez mais dinâmicas e elásticas, são necessários esquemas de interação descentralizada e dinâmica (FERNANDEZ; TEDESCHI; PRIOL, 2016). Por isso é importante discutir opções com relação à provisão de recursos computacionais.

1.2 Motivação e objetivos

As pesquisas em PSP carecem de integração de métodos e análises, além de infraestrutura computacional adequada. O individualismo dos grupos de cientistas leva a um isolamento que é prejudicial para o avanço do campo de PSP e as primeiras experiências de cooperação se mostraram bastante promissoras. Tendo esse quadro em vista, considera-se as arquiteturas orientadas a serviço podem ser a peça-chave para a associação desejada. Um ambiente SOA pode proporcionar as ferramentas adequadas para a troca de informações e a disponibilização de recursos para o processamento dos dados envolvidos nas pesquisas.

O crescimento da *e-science* e das opções de *gateways* científicos têm contribuído na padronização de procedimentos e no acesso às ferramentas. Entretanto, esses portais são incapazes de identificar automática e corretamente as necessidades de um experimento e oferecer recursos adequados, o que pode influenciar negativamente nos resultados obtidos.

Dessa forma propõe-se MDAPSP, uma arquitetura para oferecer um ambiente de integração de diferentes técnicas para execução de experimentos, balanceamento de carga, provisionamento otimizado de recursos e desacoplamento de módulos. Procedimentos de PSP têm grande variabilidade de requisitos e necessidade de escalonamento de tarefas para ambientes com capacidade adequada. A princípio, a arquitetura é construída em um modelo de simulação com auxílio do *framework WorkflowSim* (CHEN; DEELMAN, 2012), para fins de extrapolar os dados de análise, comparação de tecnologias e validação do conceito. Como ponto fundamental da MDAPSP, apresenta-se o *Decision Maker*, módulo responsável por identificar qual o melhor ambiente de processamento para um experimento científico em PSP, ao analisar os requisitos inerentes a tal procedimento e definir um modelo baseado em modelos de regressão. Considerando um ambiente heterogêneo de máquinas, o módulo distribui a carga adequadamente, alcançando redução de tempo de resposta. A arquitetura completa é descrita em forma conceitual, que estabelece os módulos gerais. Sua estrutura inclui a proposta de escalonadores de tarefas de

workflows científicos além de fornecer um modelo para estabelecimento de *gateways* científicos, que vise o aumento de desempenho e o acesso de múltiplos usuários.

Os resultados alcançados pelo *Decison Maker* mostram que o modelo de aprendizado de máquina baseado em *Support Vector Machines* (SVM) é efetivo para identificar o ambiente computacional necessário para a execução de um experimento. Baseado em histórico de execuções, esse mecanismo cria um modelo que guia a tomada de decisão com relação à capacidade de máquina. No modelo de simulação, são descritos os módulos fundamentais da MDAPSP, com a proposta de dois escalonadores de *workflows* científicos para o modelo da infraestrutura. A análise dos escalonadores mostra sua boa capacidade, superando algoritmo disponível na literatura.

A pesquisa buscou identificar os principais aspectos dos *gateways* científicos e os gargalos que podem afetar o desempenho de tais sistemas. Considera-se que uma arquitetura SOA pode oferecer a modularização e integração necessárias para a oferta de soluções. Junto a isso, é fundamental distribuir adequadamente a carga de trabalho em um ambiente heterogêneo de computadores, capaz de reduzir o tempo de processamento e otimizar a utilização dos recursos. Para atingir essa meta, é constituído um ambiente de provisionamento que contempla uma infraestrutura de recursos variada (computadores *workstation*, *clusters* e servidores de computação em nuvem) e indica em qual conjunto de máquinas realizar os experimentos, conforme parâmetros de entrada e máquinas disponíveis.

Portanto, os objetivos desta tese são os seguintes:

- Otimização da capacidade funcional de *gateways* científicos em uma arquitetura modular,
- Identificação dos requisitos de recursos computacionais em um modelo de *workflow* científico e
- Distribuição de tarefas de *workflow* de PSP em recursos computacionais heterogêneos.

A hipótese proposta por esta tese é:

Uma arquitetura modular, escalável e de fácil utilização é capaz de otimizar a execução de experimentos *in silico*, como PSP. A identificação dos requisitos e o provisionamento de recursos de forma otimizada permite o escalonamento em uma infraestrutura heterogênea, melhorando o desempenho de tais experimentos em *gateways* científicos.

O procedimento de avaliação e validação dessa tese seguiu os seguintes passos:

- Estudo de caso no Galaxy, com identificação de pontos críticos;
- Estruturação da arquitetura conceitual da MDAPSP;

- Simulação no *WorkflowSim*, descrevendo o comportamento geral da arquitetura e o comportamento da carga de trabalho;
- Comparação com outros algoritmos da literatura disponíveis para validação dos resultados.

A afirmação proposta por esta pesquisa e confirmada experimentalmente é:

As pesquisas científicas são compostas por processos computacionais complexos e exigentes. Eles podem ser estruturados em um padrão de *workflow* que permite a compreensão e a distribuição da carga de trabalho. A provisão de recursos computacionais deve ser adequada para evitar gargalos e resultados de baixa qualidade. Este trabalho fornece módulos para distribuição otimizada em um ambiente heterogêneo, com o objetivo de melhorar o desempenho em *gateways* científicos e aprimorar a ocupação dos recursos. Esses módulos integram uma arquitetura que oferecerá características complementares.

A evolução científica tem caminhado para integração de campos de pesquisa com foco em ampliar o alcance das soluções técnicas e aumentar o desempenho em outros domínios. Devido à interdisciplinaridade dessa tese, o levantamento de trabalhos relacionados envolve diferentes áreas e suas relações. A Figura 2 ilustra a relação das referências utilizadas nesse trabalho, indicando tanto a quantidade quanto a relação entre elas. A contribuição do trabalho é representada pelo círculo amarelo. O círculo “Outros” indica ferramentas de suporte e referências isoladas utilizadas dentro de um contexto.

Os vários pontos de análise da pesquisa e o aumento do interesse pelos tópicos tratados, levaram a um extenso número de referências utilizadas nessa tese. A Figura 3 apresenta um histograma das referências utilizadas nesse trabalho, relacionando a quantidade de publicações por ano.

1.3 Estrutura da tese

Este trabalho está organizado da seguinte maneira:

- No Capítulo 2 é apresentado todo o referencial teórico envolvido nessa pesquisa;
- O Capítulo 3 descreve o estado da arte do tema de pesquisa, apresentando trabalhos relacionados;
- O Capítulo 4 apresenta um estudo geral do *framework* Koala, uma versão personalizada do Galaxy, especializada em PSP. Esse estudo permite apontar limitações do modelo atual e propor soluções na arquitetura;

Figura 2 – Diagrama das referências da tese.

Fonte: Elaborada pelo autor.

Figura 3 – Histograma trabalhos referenciados na tese.

Fonte: Elaborada pelo autor.

- O Capítulo 5 traz informações sobre a concepção da arquitetura MDAPSP, as tecnologias utilizadas e a modelagem;

- O Capítulo 6 descreve os módulos desenvolvidos, os algoritmos principais e a organização do fluxo de dados;
- O Capítulo 7 traz os experimentos e a avaliação dos resultados;
- Finalmente, o Capítulo 8 traz as conclusões do trabalho, as contribuições, as publicações derivadas da pesquisa, dificuldades enfrentadas e os trabalhos futuros.

REFERENCIAL TEÓRICO

2.1 Considerações iniciais

Este Capítulo apresenta o embasamento teórico no qual essa tese está estabelecida. São descritas em linhas gerais as teorias que envolvem PSP, SOA, computação em nuvem, *workflows* científicos, *gateways* científicos e simulação computacional. Essa fundamentação é necessária para demonstrar o posicionamento da pesquisa, o histórico da evolução de cada um dos conceitos e como eles se integram na solução proposta.

2.2 Predição de Estruturas de Proteínas

Proteínas são cadeias poliméricas lineares que executam funções biológicas, fundamentais para a manutenção da vida (ALM; BAKER, 1999). O processo de "*folding*" (em português enovelamento) é um método físico-químico no qual uma cadeia encontra sua configuração funcional em uma estrutura tridimensional específica, sendo essa a condição para realizar sua função biológica (AMATO; DILL; SONG, 2002). A Figura 4 traz a representação de uma estrutura de proteína, semelhante a um fio de pérolas, enovelado no estado fornecido pela compressão de muitas ligações entre as pérolas (resíduos), que são próximas espacialmente, mas distantes ao longo da sequência (linhas vermelhas tracejadas) (KALEGARI; LOPES, 2013). As pesquisas em predição de estruturas de proteínas vêm esbarrando em grandes obstáculos, apesar da evolução da tecnologia. De acordo com ISLAM; CHETTY mesmo com modelos simplificados a busca pela solução ótima consiste em um problema NP-completo (em princípio é necessário tempo exponencial para investigar o espaço de busca para encontrar a estrutura correta) e os *frameworks* disponíveis geralmente são utilizados em contextos que resultam em modelos de proteínas em baixa resolução (ISLAM; CHETTY, 2013). Também é preciso considerar que as pesquisas por algoritmos de PSP tentam buscar eficiência computacional, para reduzir o tempo de processamento necessário (BRASIL; DELBEM; SILVA, 2013).

Figura 4 – Ilustração de uma estrutura de proteína.

Fonte: [Kalegari e Lopes \(2013\)](#).

As moléculas de proteínas são cadeias heterogêneas (contém heteroátomos, átomos diferentes do Carbono em sua constituição) não ramificadas de aminoácidos. Os aminoácidos são formados por uma amina e um grupo carboxílico. Na maioria das vezes são quartenários, sendo compostos por Carbono (C), Nitrogênio (N), Hidrogênio (H) e Oxigênio (O), contando também com o Enxofre (S) em alguns casos. São ligados a uma cadeia lateral, da qual é classificado como polar, apolar ou neutro ([DEFELICIBUS, 2016](#)):

- **Polar** ou Hidrofilico são compostos que dissolvem em água.
- **Apolar** ou Hidrofóbico são compostos que não se dissolvem na água.
- **Neutro** apresentam grupos químicos que tendem a formar ligações de hidrogênio.

Uma cadeia de aminoácidos é denominada peptídeo, que em suas formações mais complexas dão origem às proteínas. Ao dobrar e enovelar-se de maneira a assumir uma forma tridimensional as proteínas se tornam capazes de realizar a sua função biológica. Há também o processo contrário, chamado desnaturação (ou *unfolding*), em que uma proteína torna-se uma cadeia amorfa e não funcional de aminoácidos, perdendo a sua configuração funcional. Esse processo é irreversível na maioria das vezes ([AGOSTINI et al., 2006](#)). As estruturas das proteínas podem ser classificadas em primária, secundária, terciária e quartenária. Estrutura primária refere-se à sequência dos aminoácidos que compõe a proteína, da qual as informações que podem ser extraídas são a quantidade de resíduos e como eles se ligam ([STANDLEY et al., 2008](#)). A estrutura secundária representa a organização dos aminoácidos na sequência primária da proteína, definida pela capacidade de rotação das ligações entre os carbonos alfa dos aminoácidos e dos seus grupos amina e carboxilo. A estrutura terciária refere-se à organização tridimensional

da proteína no espaço e à estabilização fornecida pelas ligações. Essa estrutura é definida como a forma nativa da proteína (FACCIOLI, 2012). Por último, a estrutura quaternária é o resultado da união de várias estruturas de proteína em um complexo multi-proteico (STANDLEY *et al.*, 2008). Um exemplo dessas estruturas pode ser visto na Figura 5.

Figura 5 – Ilustração das estruturas que as proteínas podem assumir

Fonte: Adaptada de Lehninger, Nelson e Cox (2008).

O problema de *folding* de proteínas busca entender como a proteína se dobra dinamicamente ao seu estado nativo - uma configuração energeticamente estável. A interação entre os diferentes grupos químicos (o esqueleto da cadeia, envolvendo a sucessão de ligações peptídicas e também as cadeias laterais) na estrutura da proteína permite a formação das estruturas 3D, determinada pela sequência de resíduos de aminoácidos e essencial para a função da proteína. O *folding* de uma proteína se dá também pelas interações com o meio, geralmente aquoso, de forma que o resultado difere se uma proteína é analisada isoladamente em solução ou no ambiente celular, em que a concentração de outras macromoléculas é elevada (SCOTT; ROGER; SILVA, 2008). PSP busca a determinação da configuração 3D de uma proteína obtida a partir da sua sequência de aminoácidos. No entanto, a qualidade das previsões relatadas na literatura é relativamente baixa, sendo que boas previsões são encontradas mais frequentemente para proteínas menores (BRASIL; DELBEM; SILVA, 2013). Compreender este processo é de grande importância prática, uma vez que algumas doenças devastadoras como Alzheimer e encefalopatia espongiforme bovina (popularmente conhecida como vaca louca) estão associados com erros na predição de proteínas (*misfolding*) (THOMAS *et al.*, 2005). A incapacidade das proteínas de estabelecer uma configuração estável e funcional é originada por diferentes patologias, sejam elas mutações ou alterações ambientais que interfiram no enovelamento. Em outros casos, o organismo tem problemas em desempenhar suas funções por conta da ausência de alguma

proteína, como em doenças neurodegenerativas como Parkinson (DAHIMÈNE *et al.*, 2013). A Figura 6 mostra um exemplo do processo de *folding*.

Figura 6 – Ilustração do processo de *folding* de uma proteína.

Fonte: Alberts *et al.* (2002).

A dificuldade em determinar a estrutura 3D das proteínas gera uma discrepância entre o volume de dados (sequências de resíduos de aminoácidos) gerados pelo Projeto Genoma¹ e o número de estruturas de proteínas atualmente conhecidas. Apenas uma pequena porção de sequências de proteínas tem soluções tridimensionais resolvidas experimentalmente (DORN *et al.*, 2014). Os métodos computacionais para a PSP podem ser classificados em quatro grupos (FLOUDAS, 2007): modelagem comparativa, reconhecimento de *fold*, primeiros princípios sem base de dados e primeiros princípios com base de dados.

A modelagem comparativa baseia-se no princípio de que sequências relacionadas evolutivamente exibem estruturas tridimensionais semelhantes. A precisão das previsões por modelagem comparativa depende do grau de similaridade da sequência. O reconhecimento de *fold* é baseado no princípio de que o número de proteínas enoveladas diferentes é significativamente menor do que as sequências diferentes geradas nos projetos genômicos. Estima-se que o *Protein Data Bank* (PDB) contenha dados suficientes para cobrir estruturas proteicas até um comprimento de cerca de 100 resíduos. As abordagens incluem métodos avançados de comparação de sequência e predição de estrutura secundária com comparação de sequências (FLOUDAS, 2007).

Para os métodos de primeiros princípios, há as duas variações. A forma sem base de dados não utiliza informações armazenadas para identificar a estrutura, empregando apenas leis

¹ <http://genomics.energy.gov/>

da Física e a sequência de aminoácidos. Com uma ampla gama de alvos, esse método está entre os mais difíceis de calcular. O modelo com base de dados é sub-classificado em categorias que contemplam comparações de fragmentos, métodos híbridos e combinação de estruturas. Estão entre os modelos mais consistentes para previsões bem-sucedidas (FLOUDAS, 2007).

Dentre as categorias citadas em PSP destacam-se *AB Initio* e homologia. O processo *Ab Initio* utiliza-se das propriedades físico-químicas dos aminoácidos para definir as funções de energia e, a seguir, algoritmos de busca tentam definir as funções que a proteína pode assumir. Já por homologia a construção dos modelos é baseada na estrutura tridimensional de proteínas homólogas, o que é considerado uma forma confiável de fazer essa definição (GERALDO; VALENTE, 2014). Os experimentos são atrelados ao objetivo da predição. Isso refere-se às interações possíveis das proteínas, sejam entre seus resíduos, seus átomos ou da estrutura com o meio, e elas são eletrostáticas, em sua maioria (FACCIOLI, 2012). O efeito mais importante no processo de *folding* é o hidrofóbico, porque ele não necessita de interações específicas, além das já presentes. Sua função é descrever a termodinâmica do comportamento de substâncias apolares em um solvente polar (DILL; BROMBERG, 2011). Desse efeito se origina a principal força indutora do processo de *folding*, além do fato de que classificar os aminoácidos em hidrofóbico e hidrofílico seja suficiente para prever a estrutura composta por hélices de uma proteína pequena (FACCIOLI, 2012). As forças indutoras do processo de *folding* são de natureza eletromagnética e podem ser covalentes e não-covalentes. Algoritmos evolutivos mono e multiobjetivo são usados para fazer a predição *Ab-Initio* da estrutura de proteínas, nos quais a opção pelo tipo e quantidade de objetivos influencia na qualidade da estrutura obtida (DEFELICIBUS, 2016).

Uma das mais importantes fontes de informação sobre proteínas é o PDB, que contém coleções de dados semi-estruturados desses componentes biológicos (PONMARY; RAJ; SHARMILA, 2007). Os arquivos PDB detêm informações sobre coordenadas atômicas de proteínas no espaço tridimensional (SOVIC *et al.*, 2010). Após a extração de dados do PDB é preciso reorganizar os dados para torná-los inteligíveis em uma sequência de texto descrita no formato FASTA (MANDAL; DAS; BHATTACHARJEE, 2012). A informação estrutural e funcional de uma proteína pode ser adicionada como campos de anotação. Dados de PSP muitas vezes se encontram em servidores dispersos, em um cenário onde informações importantes podem ser perdidas.

As técnicas testadas em PSP incluem Dinâmica Molecular, Redes Neurais, Algoritmos Evolutivos, Algoritmos de Avaliação de Distribuição, entre outros (BRASIL; DELBEM; SILVA, 2013). Algoritmos Evolutivos mostram resultados relevantes para problemas complexos e progressos importantes foram alcançados na predição de estrutura secundária usando redes neurais artificiais (JIAN-WEI *et al.*, 2013). Durante o processo de *folding* são realizadas montagens e sub-montagens das estruturas, em um processo que é comum a ocorrência de *overheads* de processamento, sendo fundamental a utilização de dados padronizados para reduzir as perdas (PHUOC; KIM, 2011).

O advento dos processadores *multi-core* contribuiu para o avanço dos processos de PSP e têm sido consideradas novas técnicas desde então. A utilização de diferentes tecnologias de SD busca acelerar os processos envolvidos e extrair a maior eficiência das plataformas (SHARMA; PAPANIKOLAOU; MANOLAKOS, 2013) (SOVIC *et al.*, 2010). O projeto *Folding@Home* (FOLDING@HOME, 2013) é um exemplo, que consiste em formar uma grade de computadores domésticos para auxiliar o processamento. O *Folding@Home* obteve o melhor resultado absoluto de predição para uma determinada proteína (ZAGROVIC *et al.*, 2002). Outra abordagem, o *FoldIt* (FOLDIT, 2013) foi desenvolvido em formato de jogo, buscando a interação humana para o reconhecimento de formas tridimensionais das proteínas (COOPER *et al.*, 2010). Os resultados obtidos foram bastante significativos, e o número de participantes foi superior a 300 mil jogadores (KHATIB *et al.*, 2011). O jogo permitia a formação de equipes de jogadores para cooperação e troca de resultados. Outro exemplo é o *WeFold* (WEFOLD, 2013), um *gateway* que reúne as soluções de diversos pesquisadores em um ambiente de trocas de informações, comparação de modelagens e disputa pelos melhores resultados. É um portal utilizado pela comunidade para integração *online* dos grupos de pesquisadores em um sistema de *coopetition*, onde há a cooperação e a competição, para verificar qual grupo apresenta a melhor resposta (ou a mais rápida), dentre outros fatores. O *WeFold* conseguiu resultados excelentes, inclusive abrindo novos campos de estudo, focados na cooperatividade entre grupos de pesquisa (WEFOLD, 2013). Outra iniciativa de integração é o *Iterative Threading ASSEmbly Refinement* (I-TASSER), um servidor *online* para processamento de PSP e anotação de proteínas (ZHANG, 2008). Esse sistema hierárquico proporciona a automatização dos procedimentos de predição e auxilia na criação de comunidades de pesquisadores, aumentando a reprodutibilidade e oferecendo ambientes de processamento.

Devido à complexidade desses procedimentos bem como a intensividade dos dados envolvidos, a exigência é grande e o prazo de execução dos experimentos é variável. O uso de ferramentas de apoio é fundamental para a entrega de bons resultados e para a disponibilização de dados, que podem se encontrar dispersos em servidores não relacionados. Considera-se que uma organização modular de SOA, usando as tecnologias dos *gateways* científicos e a padronização dos *workflows* científicos seja capaz de fornecer uma arquitetura que atenda aos requisitos da área, fornecendo os recursos computacionais de forma adequada.

2.3 Arquitetura Orientada a Serviço

SOA é uma estrutura de *software* na qual as funcionalidades devem ser disponibilizadas como serviço (ERRADI; MAHESHWARI; TOSIC, 2006). O conjunto de serviços tem baixo acoplamento, que se comunicam por meio de interfaces e protocolos padronizados (CHEN; ZHOU, 2014). O serviço independe do contexto em que se encontra ou de outros serviços, mesmo que correlatos. Esses serviços devem ser disponíveis, integráveis, confiáveis e respeitar requisitos de QoS. Como uma arquitetura de *software*, SOA é um modelo conceitual que diz

Tabela 1 – Princípios de *design* de orientação a serviço.

Princípio	Descrição
Contrato	Expressa seu propósito e capacidade em documentos em forma de contratos
Fraco acoplamento	Minimiza a dependência, um componente não tem conhecimento de toda a arquitetura
Abstração	O serviço publica seu contrato e oculta a lógica interna
Reusabilidade	Serviços podem ser utilizados por outros para construção de outras soluções
Autonomia	O serviço controla o ambiente subjacente e os recursos para se manter independente
Ausência de estado	Gerência de estados é deixada a cargo dos clientes, sem desrespeitar a autonomia
Descoberta	Descrições e metadados para permitir a descoberta e compreensão dos contratos, promovendo a reusabilidade
Composição	Serviços podem ser compostos por outros, independentemente do tamanho e complexidade do sistema

Fonte: Adaptada de [Erl \(2007\)](#).

respeito a organização e as ligações entre esses componentes, não se ocupando propriamente com o desenvolvimento de programas operacionais ([CHEN; ZHOU, 2014](#)). SOA pode assumir diversas roupagens, conforme os desenvolvedores ou o público alvo. Alguns dos princípios de *design* de SOA são definidos por [ERL](#) conforme a Tabela 1.

Em linhas gerais uma SOA deve conter um registrador de serviços, um provedor de serviços e um cliente. O cliente é a parte interessada em uma tarefa ao sistema, aquele que irá consumir os recursos disponibilizados. O provedor de serviços é componente encarregado do processamento computacional das solicitações do cliente e conta com serviços publicados em um catálogo, os quais realizam as tarefas oferecidas pela arquitetura. O registrador de serviços é o responsável por fazer a integração entre o cliente e o provedor. Ao receber a requisição inicial do cliente, procura um provedor capaz de atendê-la, encaminha a requisição e retorna a resposta. Esse registro é conhecido como *Broker* e geralmente é o elemento central de uma arquitetura ([ESTRELLA, 2010](#)). Há ainda *Internet Brokers*, disponíveis em rede, no quais os usuários podem localizar serviços ([D'MELLO et al., 2012](#)), tecnologia que tem se diversificado para aplicações com diversos níveis de serviços ([GHASEMI; MATTA, 2016](#)) e também atuado em *Internet of Things* (IoT) ([KUMRAI et al., 2017](#)). Além desses elementos básicos, uma SOA pode conter um repositório de serviços, cuja função é indicar ao *Broker* a localização dos provedores capazes de atender a uma determinada solicitação. Os serviços são independentes de plataforma, podendo ser publicados em diretórios ou repositórios públicos ou privados, para que desenvolvedores e usuários tenham a capacidade de compor suas aplicações ([KRAFZIG;](#)

BANKE; SLAMA, 2004). A ligação dinâmica entre os componentes e o fraco acoplamento são algumas de suas características principais (MANSOUR; DILLON, 2011). Uma implementação de SOA é denominada *Web Service*, aplicações autônomas que podem ser disponibilizadas, localizadas e invocadas na *Internet* (ERRADI; MAHESHWARI; TOSIC, 2006).

Tecnologias, componentes e rotinas experimentais são desenvolvidas cada vez mais em forma de serviços, fortalecendo esse paradigma (TAN; ZHANG; FOSTER, 2010). As interações entre *Web Services* ocorrem tipicamente com o uso de *Simple Object Access Protocol* (SOAP), um protocolo de comunicação baseado em *eXtensible Markup Language* (XML) para a interação de aplicações (PAPAZOGLU, 2003). SOAP funciona como *backbone* para aplicações de computação distribuída, independente de plataforma, linguagens e Sistema Operacional (SO). Podem ser utilizados também os conceitos arquiteturais de *Representational State Transfer* (REST). SOAP é mais pesado, mas capaz de realizar verificações de tipo rígido, enquanto REST é mais leve, sem marcações XML desnecessárias (TAN; ZHANG; FOSTER, 2010). As interfaces dos *Web Services* são definidas em *Web Services Description Language* (WSDL), uma linguagem de descrição definida para esse contexto (THOMAS; THOMAS; GHINEA, 2003). A teoria de *Web Services* inclui ainda um protocolo para o diretório responsável por manter as descrições dos serviços oferecidos, denominado *Universal Description, Discovery and Integration* (UDDI). Em uma arquitetura SOA esse protocolo é um componente de grande importância, especialmente nos primórdios dessa abordagem (PAPAZOGLU, 2003). Atualmente já são empregados outras formas de armazenar as descrições e localização dos provedores de serviços, ainda que o UDDI seja uma opção viável.

A pilha conceitual dos *Web Services* envolve três camadas principais: camada de rede, camada de descrição e camada de descoberta. Na camada de descoberta fica o repositório de serviços, responsável por armazenar e disponibilizar as informações dos serviços. Na camada de descrição se encontram os detalhes dos *Web Services*, as linguagens utilizadas e suas interações. A camada de rede conta com especificações de roteamento, criptografia, qualidade de serviço, anexos e gerenciamento de transações (KRAFZIG; BANKE; SLAMA, 2004). Abaixo desta estão os demais protocolos de *Internet*, como é mostrado na Figura 7.

A abordagem de SOA oferece vários benefícios, como reutilização, desenvolvimento e implantação independentes, autonomia de plataforma, transparência e flexibilidade (FELJAN *et al.*, 2015). A confiabilidade e a disponibilidade são dois pontos importantes em SOA. A confiabilidade trata dos cuidados para que a execução de um serviço retorne o resultado desejado assim como mantém o QoS acordado (prioridade de atendimento, tempo de resposta, etc.) (SINGH; HUHNS, 2005). Já a disponibilidade é a probabilidade de que um serviço esteja acessível e operacional para atender as requisições dos usuários (FACI *et al.*, 2011).

Aplicações em SOA se aplicam em integração de dados (WANG *et al.*, 2009), serviços de informação (VAIDYANATHAN *et al.*, 2011), convergência de tecnologias (YANG; ZHANG, 2012) (YANG, 2013), grades inteligentes (MALARVIZHI; KALYANI, 2013) e agricultura

Figura 7 – Pilha conceitual dos *Web Services*.

Fonte: Adaptada de Estrella (2010).

(SRINIVASULU *et al.*, 2017), por exemplo. As definições de SOA a tornam uma modelagem bastante flexível e apta a atuar em contextos de integração de tecnologias e pesquisas. Permite inclusive selecionar quais de suas metodologias podem integrar um modelo funcional. No esquema da arquitetura MDAPSP serão integrados os conceitos de *Broker*, provedores de serviço independentes, módulos integráveis e repositório de serviços. Além disso, tem como foco o modelo de provisão de recursos computacionais, área em que a computação em nuvem revolucionou a forma de oferta. SOA também tem relevância nesse aspecto, sendo conceitos complementares que podem construir ambientes flexíveis e robustos.

2.4 Computação em Nuvem

Os recursos de infraestrutura computacional disponíveis atualmente são diversos e os requisitos os mais variados. Níveis altos de processamento e produção de grandes volumes de dados exigem cada vez mais das capacidades instaladas. É preciso identificar o tipo de máquina a ser utilizado em cada contexto, diversificando os esforços (SANDES *et al.*, 2016a) (SANDES *et al.*, 2016b). Para SINGH "a nuvem é o ambiente mais relevante para processamento de grandes

Tabela 2 – Comparação de infraestruturas de computação

Plataforma	Vantagens	Desvantagens
Cluster	Alto desempenho de forma econômica.	Requer instalação especializada, suporte de <i>hardware</i> e de TI.
Nuvem	Virtualização traz flexibilidade. Bom para tarefas únicas de HPC quando não são necessários recursos persistentes.	Preocupações com privacidade. Menor controle de processos. Largura de banda limitada. Difícil movimentar grandes volumes de dados.
Grid	Recursos computacionais em grande escala com baixo ou nenhum custo (voluntários).	Grandes transferências de dados são difíceis ou impossíveis. Controle mínimo de <i>hardware</i> , incluindo disponibilidade.
Heterogêneo	Fração do custo de um cluster. Otimizado para paralelismo de grão fino, computacionalmente intensivo. Controle local de dados.	Experiência significativa e tempo de programador necessários para implementar aplicativos; geralmente indisponível em casos baseados em <i>cluster</i> e na nuvem.

volumes de dados genômicos"(SINGH, 2016). As crescentes necessidades e interesses em Bioinformática tendem a empurrar os limites do ambiente computacional e demandar cada vez mais capacidade. As principais arquiteturas que formam plataformas de *Distributed Computing Infrastructure* (DCI) são listadas a seguir (BUYA *et al.*, 2009) e na Tabela 2 é feita uma comparação, apontando vantagens e desvantagens.

1. **Cluster** - Múltiplos computadores interligados, geralmente por meio de rede de alta velocidade, que efetivamente funcionam como um único computador.
2. **Computação em Nuvem** - Abstrai as arquiteturas de hardware subjacentes (servidores, armazenamento e rede), permitindo acesso conveniente e sob demanda a um ambiente compartilhado de recursos de computação que podem ser facilmente provisionados e publicados.
3. **Computação em Grade** - Uma combinação de computadores em rede fracamente acoplados de diferentes centros administrativos que trabalham juntos em tarefas computacionais comuns. Tipificado por esforços de computação voluntária (como o Folding@Home (FOLDING@HOME, 2013)).
4. **Computação Heterogênea** - Computadores que integram aceleradores especializados - por exemplo, *Graphics Processing Unit* (GPU) ou lógica reconfigurável como *Field Programmable Gate Array* (FPGA).

A computação em nuvem surge com as necessidades crescentes de processamento e armazenamento de empresas e usuários, que ao mesmo tempo buscavam a redução dos gastos

com recursos computacionais. Ela permite a disponibilização, via *Internet* desses recursos, de forma rápida, elástica e sob demanda. Além disso o gerenciamento e manutenção dos sistemas fica a cargo do provedor de serviços, sem necessidade de intervenção do usuário final. Outro ponto favorável é a diversidade de ofertas, concorrência de preços, suporte e tecnologias. Um dos pontos fortes é a capacidade elástica de alterar a quantidade de recursos para mais ou menos conforme necessidades de curto e médio prazo (ARMBRUST *et al.*, 2009). Por outro lado para uma boa experiência de utilização é fundamental que a conexão com a rede seja estável e rápida para que o meio de transmissão não se torne um gargalo e suas vantagens sejam perdidas. Os desafios envolvem o uso colaborativo de diversas tecnologias – algumas emergentes – para permitir a integração de múltiplos projetos de pesquisa (ROSENTHAL *et al.*, 2010).

A computação em nuvem proporciona facilidades além de processamento e armazenamento de dados. A manutenção de servidores nas instalações de uma instituição é um trabalho que dispense mão de obra, vigilância, investimento e consumo de energia. Seja como *mainframes* (PAULSON, 2001), computadores de grande porte ou *clusters*, o objetivo é ampliar a capacidade de máquinas simples e distribuir tarefas em vários centros de processamento para acelerar a obtenção de resultados. Esse foi o ambiente disponível por muitos anos, que expandiu os horizontes das áreas de ciência e negócios. O advento dos processadores de múltiplos núcleos ampliou grandemente a capacidade de computadores pessoais e abriu oportunidade de paralelismo nessas máquinas. Entretanto, as demandas computacionais seguem em crescimento, exigindo recursos cada vez mais poderosos. A computação em nuvem busca atender tais requisitos, provendo o gerenciamento de recursos com elasticidade, balanceamento de carga, tolerância a falhas, segurança e acompanhamento de desempenho (VELEV; ZLATEVA, 2011). Os serviços denominados *pay-as-you-go* fornecidos por empresas como Amazon e Google permitem contratar diferentes configurações de máquinas de acordo com a necessidade (MENON *et al.*, 2012). A computação em nuvem, fornece meios para processar, armazenar ou compartilhar informações e estende o horizonte em áreas da ciência e de negócios. Desenvolvimento de *softwares Web* e computação em nuvem baseada em SOA representam uma prática recente em computação paralela e distribuída (CHEN; ZHOU, 2014) e capacitam a integração de tecnologias.

A computação em nuvem pode ser classificada em três categorias principais: *Software as a Service* (SaaS), *Platform as a Service* (PaaS) e *Infrastructure as a Service* (IaaS) (VECCHIOLO; PANDEY; BUYYA, 2009). SaaS refere-se aos *softwares* oferecidos por meio de provedores de serviço, os quais tiveram um aumento de mercado conforme a largura da banda de *Internet* aumentou. PaaS fornece meios aos usuários para criação de aplicações remotamente, inclusive implementação e operação. Conhecidos como *Application Programming Interface* (API), têm vasta utilização e são ferramentas úteis no auxílio ao desenvolvimento de sistemas e divulgação tecnológica. Por fim, IaaS fornece recursos de *hardware* e sistemas operacionais para usuários, que podem implementar seus ambientes de trabalho e integrar tecnologias para variados usos (VELEV; ZLATEVA, 2011). Os desdobramentos desses modelos são vários e atualmente já existe o conceito de **-as-a-Service* (*aaS) (SCHAFFER, 2009), multiplicando as ofertas. A

Figura 8 mostra os diferentes níveis de abstração conforme o modelo de computação em nuvem oferecido.

Figura 8 – Níveis de abstração em *Cloud* de acordo com o tipo de estrutura.

Fonte: Adaptada de Cambiucci (2014).

Pesquisas em genética de alto rendimento geram uma grande quantidade de dados que nem sempre podem ser tratados por computadores de menor capacidade. Este gargalo pode ser superado pelo uso de computação em nuvem para armazenamento e processamento, além de demais tarefas inerentes. Algumas das ferramentas já disponíveis nesse sentido são o *Hadoop* (WHITE, 2009) e *MapReduce* (DEAN; GHEMAWAT, 2008) do Google, que oferecem ferramentas de computação paralela para grandes volumes de dados, além de aplicações para análise de imagens, mineração de dados e sequenciamento genético que podem ser disponibilizados para pesquisa colaborativa. Para abordar os desafios envolvidos no processamento de dados intensivos é necessário um mapeamento adequado dos recursos e suas limitações tais como dinheiro, energia, armazenamento e recursos pessoais (SANSOM, 2010). Além disso é preciso compreender a natureza dos dados e analisar os diversos algoritmos. Fatores que podem influenciar na decisão e na estrutura do ambiente são o tamanho e a complexidade de dados, o transporte de informações, a possibilidade de paralelização e a complexidade dos algoritmos (RITTINGHOUSE; RANSOME, 2009).

As vantagens proporcionadas pelo modelo de nuvem são várias, porém algumas limitações impedem uma maior utilização. Características que são críticas atualmente são armazenamento e largura de banda. Problemas com relação à privacidade de dados e hospedagem em países com legislação deficiente são frequentemente listadas entre os motivos para a não adoção dessa infraestrutura (MEIYAN, 2013). A intermitência da rede de transmissão é outro problema recorrente, principalmente quando consideradas as divergências entre um território e outro, limitando o acesso de usuários de regiões menos desenvolvidas (TOOSI; SINNOTT;

Tabela 3 – Comparação de valores entre os provedores de nuvem.

Provedor	CPU	RAM	Armazenamento	Preço (US\$)
Dreamhost	4 CPU	8 GB	100 GB	\$48.00
Exoscale	4 CPU	8 GB	200 GB	\$95.00
Google	4 CPU	15 GB	200 GB	\$131.00
Ecloud	7 CPU	8 GB	50 GB	\$132.00
Amazon C5	4 CPU	16 GB	50 GB	\$135.00
Atlantic.net	4 CPU	8 GB	160 GB	\$80.00
Zettagrid	4 CPU	8 GB	50 GB	\$251.00
Windows A3	4 CPU	7 Gb	285 Gb	\$175.00
M5	4 CPU	8 GB	50 GB	\$72.00

Fonte: Dados da pesquisa.

Nota – Preços por mensalidade.

BUYA, 2017). A impossibilidade de um acesso às máquinas também pode ser problemática para usuários avançados, que exijam um controle amplo do conjunto de recursos (YUAN *et al.*, 2013).

Há ainda que se considerar o investimento necessário para contratação dos seus serviços. Um leque de opções é oferecido aos usuários, que podem alugar instâncias prontas de máquina ou personalizar as configurações, com custos variáveis em poder de processamento, quantidade de memória *Random Access Memory* (RAM), tamanho de disco, entre outros. Alguns provedores oferecem amostras grátis de *Virtual Machine* (VM) com configurações simples, entretanto as experiências científicas geralmente exigem máquinas de maior capacidade. Isso eleva os preços consideravelmente e a cobrança pode ser por hora de uso, sendo impeditivo para execuções muito longas. Na Tabela 3, é mostrado um orçamento de diferentes provedores de nuvem. Em muitos casos, as opções são caras, especialmente no longo prazo. Em geral, a amostra considera máquinas virtuais com 4 núcleos, 8 *GigaByte* (GB) de RAM, 50 GB de armazenamento e SO gratuito, com variação nos requisitos em alguns provedores. Os valores são calculados em mensalidades e foram obtidos na página dos provedores em março de 2018. Não foram consideradas ofertas específicas por contratos de longo prazo e os custos de transferência.

A área de computação em nuvem tem, como qualquer outra, suas limitações e dificuldades, porém os ganhos com sua utilização são diversos, capazes de atender muitas necessidades. As pesquisas em PSP também podem se beneficiar desse ambiente, utilizando a capacidade de processamento e armazenamento, além da acessibilidade distribuída aos recursos. A integração que se espera conseguir neste projeto passa por ferramentas conhecidas tanto em SOA quanto em computação nuvem, que devem trabalhar em conjunto, agregando ainda outras alternativas de infraestrutura. Alternativas de padronização, reexecução e reprodutibilidade são fundamentais nesses ambientes distribuídos e em procedimentos extensivos.

2.5 Workflow Científico

A complexidade dos estudos científicos cada vez mais requer apoio de computação especializada. Um experimento típico é um ciclo de repetições de mover os dados para o ambiente computacional, definir entre análise e simulação, execução dos cálculos e armazenamento dos resultados (DEELMAN *et al.*, 2009). Um *workflow* pode ser definido como uma sequência lógica de atividades ou tarefas de processamento de dados, que funciona com base em regras predefinidas, com objetivo de automatizar qualquer processo (MATTOSO *et al.*, 2013). Há dois tipos principais: os científicos e os comerciais. Os *workflows* científicos encontram aplicações no campo da computação científica para automatizar experimentos e processos científicos, enquanto os comerciais são usados para automatizar processos de negócios (SONNTAG; KARASTOYANOVA; DEELMAN, 2010). O conceito foi definido no domínio de negócio pela *Workflow Management Coalition* como "A automação de um processo de negócio, no todo ou em parte, durante o qual os documentos, informações ou tarefas são passadas de um participante para outro por ação, de acordo com um conjunto de regras processuais" (WFMC, 2014).

Workflows científicos são usados para descrever, gerenciar e compartilhar análises e aplicações científicas complexas. Definem múltiplos passos para agilizar a composição de tarefas e o fluxo de dados entre elas. Fornecem alto nível declarativo para especificar o objetivo de determinada experiência, mas não como será executado. Tal modelo é apoiado por uma estrutura de grafo, em que os nós representam tarefas e as ligações representam o fluxo de dados (RUSINKIEWICZ; SHETH, 1994). A transição desde a iniciação até a conclusão é denominada como o ciclo de vida do *workflow* (MATTOSO *et al.*, 2010).

A preservação dos materiais e métodos de experiências computacionais com anotações claras é essencial para a compreensão e reprodutibilidade de experimentos (HETTNE *et al.*, 2014). *Workflows* científicos se sobrepõem em alguns requisitos e características, ainda que existam diferenças fundamentais entre esses (WASSERMANN *et al.*, 2007). Uma delas é que os científicos são orientados a fluxo de dados, enquanto que o seu análogo de negócios centra-se em padrões e eventos de fluxo de controle. Consequentemente, muitos grupos de pesquisa têm explorado modelos e linguagens baseadas em fluxo de dados (TAN; ZHANG; FOSTER, 2010), resultando em ferramentas como o Taverna², o Kepler³ (LUDÄSCHER *et al.*, 2006) e o Triana⁴ (VAHI *et al.*, 2013).

A Figura 9 mostra exemplos de estruturas de *workflow*. A Figura 9a mostra o exemplo de um processo, consistindo de um comando simples que gera um arquivo de saída. A Figura 9b mostra o exemplo de um *pipeline*, que nesse caso consiste na união de duas tarefas: a primeira faz a busca de um arquivo *online* e o armazena. A seguir a segunda tarefa processa o arquivo e gera o resultado. A Figura 9c mostra o exemplo de uma divisão. Um primeiro processo faz a busca do

² www.taverna.org.uk

³ www.kepler-project.org

⁴ www.trianacode.org

arquivo de entrada, o qual é dividido por uma segunda tarefa, que distribui suas partes. Cada parte é processada individualmente, gerando arquivos de saída próprios. A Figura 9d mostra o exemplo de uma união, em que tarefas prévias separadas produziram arquivos de resultados, sobre os quais é executada uma rotina de concatenação, que produz um arquivo único, a ser utilizado posteriormente (DEELMAN *et al.*, 2015). Esses modelos básicos são depois combinados para formar *workflows* complexos. Um exemplo de *workflow* completo pode ser visto na Figura 10, que representa as tarefas de automação de operações de processamento em sequenciamento genético. Nele são divididos arquivos de sequências em várias partes e convertidos para o formato apropriado. Em seguida, sequências ruidosas e contaminantes são filtradas, para mapeá-las para seus locais genômicos. A partir dos passos de mapeamento individuais, é formado um único mapa global (LIN *et al.*, 2017). As cores indicam as diferentes tarefas executadas no *workflow*.

Figura 9 – Tipos de *workflow* base para construção de grafos complexos.

Fonte: Adaptada de Deelman *et al.* (2015).

Os sistemas de gerenciamento de *workflows* científicos, ou SWMS, fornecem infraestrutura de configuração, execução e monitoração, em um ambiente em que experimentos podem ser definidos e executados, coordenando a operação dos componentes individuais que constituem o *workflow* - processo também referido como orquestração (LIU *et al.*, 2015). *Workflows* científicos e SWMS surgiram para resolver problemas de compatibilidade, manipulação de grandes volumes de dados e fornecer uma maneira fácil de usar e de especificar as tarefas de um período específico no experimento. O objetivo de um SWMS é oferecer uma linguagem geral para simplificar o trabalho de definir um *workflow* científico e orquestrar a execução de um experimento (DEELMAN *et al.*, 2009).

Os *workflows* tornaram-se úteis também no compartilhamento de conhecimento e de

Figura 10 – DAG do *workflow* científico *Epigenomics*, usado em pesquisas genéticas.

Fonte: Adaptada de [Lin et al. \(2017\)](#).

métodos experimentais, permitindo uma compreensão geral dos procedimentos envolvidos, assim como a construção de novas sequências de execução ([LIU et al., 2012](#)). Os *workflows* podem assumir diversos estados, conforme a fase de atuação em que se encontram. Uma execução pode ser incompleta ou podem ser necessárias alterações na estrutura. Isso passa pela estrutura do ciclo de vida de um *workflow*, que pode ser definido em quatro fases principais: composição, mapeamento, execução e proveniência ([GÖRLACH et al., 2011](#)).

A fase de composição descreve os passos e dependências que definem a análise de modo geral. O resultado desse trabalho é um grafo que pode ser descrito e armazenado em vários formatos. A especificação pode usar linguagens particulares como *Business Process Execution Language* (BPEL), *Simple Conceptual Unified Flow Language* (SCUFL), *Directed Acyclic Graph Manager* (DAGMan) and *Directed Acyclic Graph in XML* (DAX), ou mesmo em texto plano. Para simplificar o trabalho dos pesquisadores, alguns dos *gateways* científicos oferecem formas de especificação via interfaces gráficas ([LIU et al., 2012](#)). A representação mais comum é o grafo direto, seja por *Directed Acyclic Graph* (DAG) ou *Directed Cyclic Graph* (DCG) sendo esse último menos utilizado. DAG é um *workflow* executável (ou concreto), que pode ser processado em ambientes computacionais ([LIU et al., 2012](#)).

Mapeamento refere-se à fase de gerar um modelo executável baseado numa descrição independente dos recursos, denominada *workflow* abstrato ou instância. As tarefas são mapeadas aos recursos de acordo com as estratégias definidas, como troca de mensagens para sincronização, uso de sistemas de arquivos compartilhados ou mapeamento definido pelo usuário ([DEELMAN et al., 2009](#)). Em casos de *workflows* baseados em serviço, o mapeamento inclui a descoberta e ligação aos serviços apropriados, considerando requisitos de QoS. Para *workflows* baseados em

sistemas *stand-alone*, essa fase inclui a descoberta de recursos para execução e otimização, além de possíveis alterações na modelagem (LIU *et al.*, 2012). Os passos básicos do mapeamento podem ser descritos como: (i) definição e descoberta, (ii) mapeamento definido pelo usuário, (iii) mapeamento supervisionado (por escalonador ou *broker*) e (iv) otimizações no *workflow* (LIU *et al.*, 2012).

A execução do *workflow* é a parte do processamento das tarefas descritas. Os serviços são designados para os recursos disponíveis e executados seguindo a ordem de prevalência ou de dependência (DEELMAN *et al.*, 2009). Os SWMS têm modelos diferentes de atuação, conforme modelagem e estratégias de alocação. Interações individuais com as tarefas são mais raras e complexas do que alterações no *workflow* e suas sub-divisões (JUVE *et al.*, 2013). A escalabilidade entre muitas plataformas é uma preocupação que requer cuidados para evitar atrasos e sobrecargas. Para esse fim, há modelos baseados em nuvem e grade, que têm sido usados com sucesso há um bom tempo (TAYLOR *et al.*, 2003) (FAHRINGER *et al.*, 2005). A tolerância a faltas também tem seu papel nessa fase (COUVARES *et al.*, 2007), juntamente com o redirecionamento de fluxo. Os *workflows* adaptativos têm a capacidade de incorporar dados a uma aplicação e também manobrar durante a execução, sendo usados em sistemas de tempo real. Essa adaptação é difícil de ser tratada em outros sistemas, como as grades por exemplo (YU; BUYYA, 2005). As estratégias para manobras podem ser ramos de caminho distintos e re-escrita dinâmica de *workflow*, desde que evitadas ações ilegítimas (JUVE *et al.*, 2013).

A coleta de dados de proveniência está conectada à todas as fases, sendo definida pelo armazenamento do histórico de geração e tratamento das informações de um *workflow*. Especificamente, a cadeia de processos (incluindo instante de tempo, versão de programa, nó de execução, etc.), dados intermediários e o arquivo fonte inicial (DEELMAN *et al.*, 2009). Os dados podem ser armazenados interna ou externamente, de formas genéricas (LIU *et al.*, 2012). Além do rastreamento dos dados, é possível também manter os dados de *design* do *workflow* conforme o mesmo evolui e identificar origens em comum de diferentes ramos. Dados como esses são importantes tanto para reprodutibilidade como para processos educacionais. Adicionalmente, as modificações feitas em um *workflow* original podem dificultar a análise dos resultados, sendo que as anotações fornecem informações essenciais de sua estrutura. A manipulação inteligente de dados permite a extração de informação em larga escala, acelerando a evolução das pesquisas (GRUNZKE, 2016). Essas fases são representadas na Figura 11, na qual a fase de mapeamento é descrita como uma sub-divisão de uma fase denominada análise, que engloba o estudo dos dados gerados pela composição e execução (MATTOSO *et al.*, 2010).

Workflows científicos podem proporcionar ou demandar interoperabilidade. Como as aplicações podem ser compostas de partes heterogêneas com finalidades, requisitos e interações variadas, não há atualmente um sistema que acomode todas as descrições possíveis (DEELMAN *et al.*, 2009). Os princípios para interoperabilidade são o planejamento, a coordenação e a sincronização de comunicação (CHEN *et al.*, 2010). A padronização deve obedecer a estrutura

Figura 11 – Fluxograma do ciclo de vida de um *workflow* científico.

Fonte: Adaptada de Mattoso *et al.* (2010).

das bases já utilizadas em projetos anteriores, incluindo acesso, tráfego de dados e interoperação de aplicações. A integração deve permitir que a expansão e atualização dos *softwares* envolvidos seja simplificada e estimulada, o que permite extrair vantagens de escalonamento de tarefas para *datacenters* distribuídos (MAHESHWARI *et al.*, 2013). Em ambientes de DCI, os *workflows* científicos são utilizados para coordenar a execução de experimentos *in-silico*, enquanto a coleta de dados de proveniência auxilia na avaliação do desempenho geral (JRAD; TAO; STREIT, 2013) (POUCHARD *et al.*, 2017).

Gerência de recursos e dados são pontos essenciais em DCI. As simulações computacionais permitem explorações de realidades modeladas, enquanto métodos quantitativos melhoram a compreensão dos fenômenos observados. Contudo, tais abordagens requerem armazenamento de grande capacidade e recursos de computação de alto desempenho. Isso é alcançado com a combinação de métodos tecnológicos. Nesse aspecto os *gateways* científicos liberam os cientistas de lidar com as infraestruturas de computação, gerenciando suas demandas via recursos de processamento, interfaces e servidores de armazenamento. Neste ambiente tecnológico complexo, apresentam-se uma quantidade significativa de desafios (GESING *et al.*, 2017).

2.6 Gateways Científicos

E-science é definida por (COLLADO, 2017) como “ciência computacionalmente intensiva realizada em ambientes altamente distribuídos ou ciência que utiliza imensas quantidades de dados”. Nessa linha de pesquisa, os esforços se concentram em oferecer o acesso a recursos de qualidade e desempenho de forma transparente. *E-science* visa a integração de ferramentas,

provendo um ambiente genérico útil em diferentes domínios. É capaz de aumentar o reúso de componentes, reduzir a curva de aprendizado e fornecer um modelo consistente para usuários (ZHAO *et al.*, 2015), geralmente por meio de *gateways* científicos. As publicações na área se pulverizam entre diversos temas e há eventos científicos específicos⁵.

Os *gateways* científicos abordam as demandas de usuário adaptadas a uma comunidade específica. A gestão exige colaboração estreita com os especialistas do domínio, além de gerenciamento de projetos, licenciamento, composição de equipe, sustentabilidade, recursos de *High-performance Computing* (HPC), visualização e usabilidade (GESING *et al.*, 2017). O senso de comunidade é forte, conectando colaboradores e disponibilizando componentes em organizações como o *Science Gateway Community Institute* (SGCI). Existem muitos desafios ao trabalhar com um *gateway* científico, dentre os quais pode-se destacar: escolher um que seja ativamente mantido, verificar a possibilidade de adicionar serviços, capacidade de gestão remota, analisar o desempenho dos recursos e o suporte da comunidade de usuários (MADDURI *et al.*, 2015).

Os *gateways* científicos foram desenvolvidos para facilitar o uso de computação avançada, definição de padrões e a execução de aplicações científicas de forma natural (BALASKO; FARKAS; KACSUK, 2013). Neles, os nós e aplicações são apresentados como caixas-pretas, das quais os usuários não recebem informações detalhadas. Costumeiramente são conectados a servidores robustos para lidar com aplicações de larga escala e grandes volumes de informação (GESING *et al.*, 2017). Tendem a focar em um campo científico e agregam aplicações compartilhadas, as quais podem ser executadas separadamente ou de forma estruturada. As vantagens principais são acessibilidade, transparência, interface modificável e facilidade de uso (BALASKO; FARKAS; KACSUK, 2013). Uma comunidade de cientistas necessita de alguns serviços primordiais de uma infraestrutura de *e-science* (KACSUK, 2011):

- acesso transparente a recursos, ofertados em diferentes modelos de computação;
- repositório de aplicações acessível com diversidade de serviços executáveis;
- interface para execução, monitoramento e manipulação de aplicativos e dados;
- suporte a acesso concorrente;
- ambiente confiável, mesmo em infraestruturas inseguras (como grades); e
- apoio à colaboração por meio do compartilhamento de aplicativos e bancos de dados.

Os requisitos de uma comunidade de usuários são fundamentais no processo de desenvolvimento de um *gateway* científico (WOLSTENCROFT *et al.*, 2013). Seus membros podem ser divididos em cientistas regulares e técnicos de suporte. Os primeiros focam em suas pesquisas e

⁵ <https://sciencegateways.org/engage/annual-conference>

dados, sem se preocupar com detalhes técnicos como local de execução ou versões de *software*. Os últimos identificam as necessidades do ambiente e oferecem soluções, em forma de recursos computacionais, aplicações e *workflows* (GUGNANI *et al.*, 2016). A taxonomia básica de requisitos identifica três aspectos principais (GESING *et al.*, 2016):

- **Compartilhamento:** as aplicações podem ser instaladas em ambientes complexos como *clusters* ou na nuvem, além de proporcionar uso facilitado de *softwares* proprietários e interfaces pouco amigáveis;
- **Complexidade:** varia de aplicações *stand-alone* a composições de *workflow*, em que o *gateway* pode ter atuações separadas;
- **Interface:** varia de acordo com o interesse da comunidade. São subdivididas em três categorias.
 - **Existente:** interface simples, padronizada e imutável perante as versões disponíveis;
 - **Automática:** genérica e sem grande variação específica entre aplicações. Pode também ser gerada semi-automaticamente;
 - **Ajuste Fino:** permite a conexão de uma interface personalizada, que acesse as funcionalidades do sistema.

Um *gateway* científico pode ser construído de duas formas: partindo do zero ou usando mecanismos que ofereçam ferramentas, APIs e bases de dados para criar uma estrutura unificada (MADDURI *et al.*, 2015). A maioria dos *gateways* são criadas da segunda forma, porque reduz o tempo de construção e usa *frameworks* com boa aceitação e suporte. O *CIG Seismology Web Portal* (BASNEY *et al.*, 2008) é um exemplo de um *gateway* criado exclusivamente para estudo sismológicos, enquanto o EnginFrame (TORTEROLO *et al.*, 2009) é um portal em nuvem que integra recursos de HPC, aplicações, dados, e mesmo SWMS como Taverna e Kepler. A própria criação e desenvolvimento de *gateways* científicos conta com ferramentas de automatização. O Vine Toolkit provê APIs de criação de aplicações *Web* que ocultem complexidade (KUROWSKI *et al.*, 2014). O SAGA oferece uma especificação para criação de interfaces de integração com aplicações (GOODALE *et al.*, 2007).

Diferentes níveis de controle podem ser definidos em um *gateway*. Nas interfaces para o usuário final há menos maleabilidade, permitindo modificar propriedades básicas e a importação de *workflows* (FIORE *et al.*, 2017). O uso de interfaces auto-geradas não requer nenhuma habilidade específica. Uma segunda categoria é oferecida para o Desenvolvedor de Aplicação que define aplicações de *workflow* e seu uso por terceiros. Soluções permitem a criação de aplicações *Web* que produzam interfaces transparentes, além de limitar a interação do usuário final. Modelos, padronizações, linguagens e *frameworks* são disponibilizados para facilitar esse trabalho (BALASKO; FARKAS; KACSUK, 2013). Dentre os métodos que podem ser ofertados

por uma aplicação em um *gateway*, três categorias se distinguem: (i) gerenciamento de entrada e saída de dados, (ii) gerenciamento de atividades de usuário e (iii) cobertura para dificuldades e problemas (BALASKO; FARKAS; KACSUK, 2013).

A facilidade de uso é o papel crucial a ser desempenhado pelos *gateways*. Tipicamente, isso é conseguido nas camadas de serviço no topo da infraestrutura a que estão conectados (KUROWSKI *et al.*, 2014). Os primeiros *gateways* científicos focavam na computação em grade (WASSERMANN *et al.*, 2007), que compartilham recursos e criam comunidades por meio de organizações virtuais (KACSUK, 2011). Um portal de *e-science* para grade fornece acesso para recursos, bases de dados e aplicações. Grades privadas podem ser utilizadas, com as vantagens de maior segurança, confiabilidade nos recursos conectados, maior privacidade e menor necessidade de redundância (BASNEY *et al.*, 2008). A desvantagem principal é o menor número de computadores disponíveis (LIU; YANG; LI, 2008). Embora as grades permitam o compartilhamento de recursos em vários domínios administrativos, elas demandam parceria entre instituições, sendo particularmente úteis na pesquisa científica, porém limitada na âmbito comercial (KOVÁCS; KACSUK; LOMAKA, 2011).

A computação em nuvem expande as alternativas para hospedagem dos *gateways* científicos (PERERA; MARRU; GANNON, 2008). Para o caso das companhias privadas, a nuvem pode ser mais adequada pela terceirização da manutenção e uso de recursos de forma dinâmica. A interface do usuário é mais simples do que na grade e serviços de alto nível podem ser ofertados para não especialistas (LIU; YANG; LI, 2008). No entanto, quando se tratam de instituições de pesquisa, o uso de nuvens comerciais é menos atraente, já que o financiamento acadêmico pode não suportar a terceirização de infraestrutura. O uso de servidores locais e *clusters* de computadores tende a ser preferido nesses casos (KOVÁCS; KACSUK; LOMAKA, 2011). A virtualização reduz esforços de portabilidade de aplicações (MAROSI; KOVÁCS; KACSUK, 2013), o que é particularmente útil em *workflows* que sofrem alterações frequentes de parâmetros. Em um sistema heterogêneo, muitas vezes é preciso ter diferentes SOs e versões de aplicativos, o que aumenta a complexidade. É também necessário registrar as aplicações disponíveis no sistema, o que pode aumentar o tempo de busca (MAROSI; KOVÁCS; KACSUK, 2013). Soluções virtualizadas podem contornar esses problemas e oferecer um modelo genérico, disponível conforme a demanda.

Além da virtualização, a combinação de diferentes infraestruturas de serviços tem sido objeto de estudo recentemente, como a oferta de recursos heterogêneos (GUIMARAES *et al.*, 2014). A exploração de DCI em *gateways* científicos é abordada também no projeto *SHaring Interoperable Workflows for large-scale scientific simulations on Available DCIs* (SHIWA), que tem como foco a interoperabilidade de *workflows* em ambiente distribuído. Focado em execução de simulações computacionais, esse projeto evoluiu para o *ER-flow*⁶, que busca a construção de uma comunidade de usuários com troca de informações e processamento de experimentos. O

⁶ <https://www.erflow.eu/home>

Tabela 4 – Exemplos de *gateways* científicos.

Nome do <i>gateway</i>	Descrição	Abordagem
Protein Model Portal	Métodos comparativos de modelagem de diferentes grupos e acesso a serviços interativos para construção de modelos e avaliação de qualidade	Pesquisa e compartilhamento de métodos
Vector Base	Fornecer dados genômicos, fenotípicos e centrados na população para a comunidade científica de vetores invertebrados de patógenos humanos	Compartilhamento de métodos, código e arquivos
Brain Life	Plataforma aberta e acesso à infraestrutura de computação em nuvem e banco de dados de estudos do cérebro. Acesso para desenvolver e publicar métodos	Ambiente de processamento e compartilhamento de dados
GenomeSpace	Framework de interoperabilidade baseado em nuvem para suportar a análise de genômica integrativa através de uma interface <i>Web</i> .	Ambiente de processamento
Omics	Serviços e ferramentas de dados baseados na nuvem para análise de dados genômicos (DNA), transcriptômicos (ARN), proteômicos (proteínas) e metabolômicos (pequenas moléculas).	Ambiente de processamento e compartilhamento de dados
Genome DB Rosaceae	Banco de dados relacional integrado que fornece ferramentas de mineração de dados em genômica e genética	Compartilhamento de dados
Plant Genomics	Fornecer as ferramentas para armazenamento e análise da sequência de nucleotídeos	Pesquisa e compartilhamento de métodos
European Variation Arch	Banco de dados de acesso aberto de todos os tipos de dados de variação genética de todas as espécies	Compartilhamento de dados
Vega	Repositório de modelos de genes de alta qualidade produzidos pela anotação manual de genomas de vertebrados	Pesquisa e compartilhamento de métodos
OMICTools	Busca de ferramentas de análise de dados biológicos. Conecta usuários para compartilhar e discutir ferramentas	Compartilhamento de métodos, código e arquivos
SEAGrid	Uso de aplicativos científicos implantados em supercomputadores, clusters e computação em nuvem.	Ambiente de processamento e compartilhamento de dados

ER-Flow fornece diversos serviços focados nas necessidades dos grupos de pesquisa, em alguns casos de forma isolada e com recursos variáveis (KAROCZKAI; KERTESZ; KACSUK, 2015).

Há diversos sistemas de suporte à realização de experimentos científicos, baseados em diferentes implementações, necessidades e campos de pesquisa. Os usos e aplicações de tais *gateways* são os mais variados, dos quais alguns que podem ser destacados: DECIDE (ARDIZZONE *et al.*, 2012), *MosGrid* (KRÜGER *et al.*, 2014), *gUSE* (KACSUK; KISS; SIPOS, 2008), *WS-Pgrade* (BALASKO; FARKAS; KACSUK, 2013) e *Lunarc* (LINDEMANN; SANDBERG, 2005). Com relação à biologia computacional, há diversas arquiteturas, com abrangência também variável. Algumas que podem ser citadas são *GeneSifter* (BASKIN *et al.*, 2009), *Genomatix* (PATEL; PANCHAL; ANJARIA, 2012), *ChIPseeqer* (GIANNOPOULOU; ELEMENTO, 2011) *LabKey Server* (NELSON *et al.*, 2011), *BioCatalogue* (BHAGAT *et al.*, 2010) e *Galaxy* (AFGAN *et al.*, 2016). Estes, variam entre soluções pagas e gratuitas, qual o SO base, a forma de acesso principal e os níveis de personalização. Uma visão geral de alguns *gateways* científicos atualmente disponíveis são listados na Tabela 4. Mais exemplos podem ser encontrados na página do SGCI⁷.

⁷ <https://catalog.sciencegateways.org/#/home>

Um dos mais importantes, o Galaxy é uma plataforma *Web* de código aberto e modificável. Amplamente utilizada em pesquisas Bioinformáticas, especialmente as intensiva em dados. Permite ter servidores disponíveis com instâncias gratuitas ou personalizadas, em que os usuários têm acesso às ferramentas de análise e mecanismos para executar e reproduzir *workflows*. O Galaxy foi projetado para facilitar o acesso a cientistas que não são familiarizados com informática avançada, sendo de propósito geral e usado em uma variedade de domínios. Algumas das suas principais características são acessibilidade, reprodutibilidade e transparência. É possível trabalhar com o Galaxy em vários tipos de instalação (servidor local ou em nuvem) e tem recursos que permitem a publicação de experimentos. Uma característica que o distingue é a possibilidade de mostrar as etapas de um *workflow*, como os resultados foram obtidos e a manutenção dos dados no mesmo local, simplificando o processo de reproduzir e verificar experimentos. Dentre as opções de *framework* disponíveis, o Galaxy foi eleito para os testes de embasamento dessa pesquisa, devido justamente ao seu alcance e capacidade.

O Galaxy conta com uma comunidade de usuários ativa e atuante, que constantemente propõe melhorias em métodos. Porém em sua maioria, os esforços são concentrados nos procedimentos ofertados ao usuário final, como novos algoritmos e técnicas de análise. As pesquisas em melhoria de desempenho computacional são menos frequentes, abrindo uma lacuna importante.

2.7 Simulação Computacional

Devido à complexidade dos procedimentos envolvidos nessa pesquisa e dos diversos pontos de análise, uma primeira versão da arquitetura é modelada em simulador. A simulação computacional auxilia a modelagem de sistemas grandes e com tráfego intenso de dados, entre outros objetivos. Diferentes aplicações têm múltiplos requisitos de composição, configuração e implantação. As infraestruturas podem ter demandas, padrões de fornecimento e tamanhos de sistemas variáveis, os usuários têm requisitos heterogêneos e conflitantes e os aplicativos desempenho e carga de trabalho incertos. O uso de infraestruturas reais para avaliações é restringido pela rigidez das mesmas, o que dificulta a reprodução dos resultados, além de reduzir a confiabilidade e escalabilidade. Além disso, é trabalhoso e custoso reconfigurar os parâmetros de *benchmarking* em uma infraestrutura de grande escala. Essas limitações são causadas por condições imperantes nos ambientes, especialmente naqueles onde não há controle por parte dos desenvolvedores, como na nuvem (CALHEIROS *et al.*, 2011). A execução de *workflows* científicos em ambientes distribuídos tem evoluído e as pesquisas consideram múltiplos aspectos, em um cenário adequado para utilizar a simulação, que permite a configuração de diversos *datacenters*, variações de capacidade, estudos de sobrecarga e etc. (BOCHENINA, 2014).

As ferramentas de simulação possibilitam a avaliação de hipóteses em um ambiente controlado, reprodutível e variável. Essa abordagem oferece benefícios em testes de serviços, análises de gargalos, cenários de carga de trabalho e desempenho de recursos em infraestruturas

diversas, antes da implantação em ambientes reais (BOCHENINA, 2014). A simulação permite ter uma estimativa precisa das capacidades do sistema, da influência dos fatores, de parâmetros gerais e tempo de processamento. Utilizado principalmente quando o sistema do mundo real é muito caro, de alta complexidade ou ainda não disponível. Diversos simuladores já foram propostos no decorrer dos anos, entre os quais o *CloudSim* destaca-se pela capacidade de gerenciar serviços e modelagem de infraestrutura de nuvem (LONG; YUQING; QINGXIN, 2013). Nesse simulador um usuário pode modelar *datacenters*, variedades de máquinas virtuais, consumo de energia e comportamento de rede (CALHEIROS *et al.*, 2011). Diversos estudos já foram conduzidos com o *CloudSim*, que alcançaram resultados notáveis, sendo um dos simuladores de maior reputação nos estudos de sistemas distribuídos.

A simulação é um dos métodos de avaliação mais utilizados em estudos de *workflows* científicos. No entanto, os simuladores de *workflow* existentes não fornecem uma estrutura que leve em consideração avaliações de custos e as falhas heterogêneas, por exemplo. Também não possuem o suporte para técnicas de otimização de *workflow*, algumas amplamente utilizadas como o agrupamento de tarefas (CHEN; DEELMAN, 2012). Ignorar sobrecargas do sistema e as falhas na simulação de *workflows* científicos podem causar imprecisões significativas, que acabem por invalidar o modelo. Por conta do contexto de *workflows* científicos foi escolhido o simulador *WorkflowSim*, que tem se tornando um dos mais importantes na área. Ele estende o *CloudSim*, uma ferramenta já bem estabelecida, documentada e testada (CALHEIROS *et al.*, 2011). As novidades trazidas pelo *WorkflowSim* são suporte para preparação e execução de *workflows*, pilha de *parsers*, *workflow engine* e escalonadores de tarefas. Outros simuladores que contemplam *workflows* são limitados, sem opções por exemplo para introdução de modelos falhas, modelos de custo, atrasos de rede e a heterogeneidade presente em sistemas reais (CHEN; DEELMAN, 2012). O *WorkflowSim* suporta um modelo multi-camadas de falhas e injeção atrasos em vários níveis do *workflow*. Os parâmetros são diretamente coletados em rastreamento de execuções reais em SWMS, como o Pegasus (DEELMAN *et al.*, 2015). Com essa ferramenta são desenvolvidos alguns dos algoritmos fundamentais da MDAPSP, de forma a avaliar um horizonte de possibilidades para a arquitetura.

WorkflowSim fornece modelagens de ambiente de nuvem e outros DCI, permitindo estudos em programação de grande escala, *clustering* de tarefas e provisionamento de recursos, inclusive em configurações de detalhes como quantidade de memória RAM por nó, largura de banda, representação de *Millions of Instructions per Second* (MIPS), variação de núcleos de *Central Processing Unit* (CPU) e tamanhos de arquivo gerados. Conta com diversos algoritmos de escalonamento, em que políticas de gerenciamento de recursos podem ser implementadas. O *WorkflowSim* ajuda pesquisadores a avaliar técnicas de otimização de *workflow* com precisão e suporte. Ainda em expansão, novas versões devem incluir outros tipos de falhas e técnicas como partição de *workflows*, avaliações de custos e métricas de utilidade de recursos (CHEN; DEELMAN, 2012).

O *WorkflowSim* conta com cinco entidades principais para conduzir gerenciamento dos experimentos: *Mapper*, *Merger*, *Engine*, *Scheduler* e *Datacenter*. *Mapper* é a entidade que inicializa o sistema e analisa o arquivo de *workflow* do tipo DAX para obter as tarefas individuais, chamadas *cloudlets* (CHEN; DEELMAN, 2012). A seguir, essas tarefas são enviadas para o *Merger* (também chamado de *Clustering Engine*) que agrupa diferentes tarefas em *jobs* (ou coleções de tarefas). A configuração padrão do simulador não prevê a realização de agrupamentos, tratando cada tarefa como uma entidade individual. Em seguida, as tarefas são enviadas para a *Engine* onde são selecionadas conforme a ordem especificada no DAX. (COTES-RUIZ *et al.*, 2017). *Engine* é a entidade encarregada de enviar ao escalonador as tarefas que podem ser processadas, inclusive verificando aspectos de dependência. Finalmente, o escalonador seleciona qual VM do *datacenter* irá realizar o processamento. A coordenação é baseada em eventos e os componentes mantêm filas de mensagens individuais. Um exemplo dessa organização pode ser observado na Figura 12.

Figura 12 – Diagrama de módulos do *WorkflowSim*.

Fonte: Adaptada de Chen e Deelman (2012).

Nota – A área entre as linhas vermelhas é suportada pelo *CloudSim*.

As comunicações são agrupadas em três etapas. As Figuras 13, 14a e 14b representam graficamente a troca de mensagens entre as entidades nas fases de inicialização, execução e finalização, respectivamente (COTES-RUIZ *et al.*, 2017). A Figura 13 ilustra a coleta de informações do arquivo DAX, para a construção dos *cloudlets*. O *Mapper* encaminha as instruções para o Mecanismo de Agrupamento, que pode formar conjuntos de tarefas. Desse ponto, é enviado para a *Engine*, que faz a preparação e encaminha *cloudlet* ao escalonador. Este pode contar com políticas distintas, conforme as necessidades de um experimento ou infraestrutura. As características são instanciadas nos *datacenters* configurados, onde são preparadas as VMs.

Nelas é realizado o processamento das instruções determinadas no *workflow* e as mensagens são retornadas confirmando o procedimento. A Figura 14a representa a fase principal de funcionamento do *framework*, ilustrando as trocas de mensagens entre os mecanismos centrais, resumando o tráfego de *cloudlets* até a conclusão do processamento. Finalmente a Figura 14b demonstra a sinalização da *Engine* para o escalonador do fim da simulação, que se encarrega de destruir as VMs instanciadas. A estrutura bem definida e organizada é outro ponto forte do *WorkflowSim*, permitindo o acompanhamento dos detalhes de implantação e execução de experimentos, facilitando melhorias e identificação de problemas.

Figura 13 – Mensagens no *WorkflowSim*. Fase de inicialização

Fonte: Adaptada de Cotes-Ruiz *et al.* (2017).

Figura 14 – Fluxo de mensagens no *WorkflowSim*.

Fonte: Adaptada de Cotes-Ruiz *et al.* (2017).

Por seus mecanismos inovadores no ambiente de simulação, sua flexibilidade e a estabilidade, o *WorkflowSim* foi a ferramenta selecionada para o desenvolvimento de alguns dos

módulos propostos para a MDAPSP, bem como para estudos de capacidade e de validação. Os módulos desenvolvidos servem como base para a evolução da arquitetura geral, além de apresentarem avanços nos mecanismos disponíveis no próprio simulador.

2.8 Considerações Finais

Este Capítulo abordou um levantamento do arcabouço teórico geral que sustenta esta tese. A arquitetura proposta baseia-se nos princípios gerais de SOA para construção de um ambiente para execução de experimentos de PSP, usando ferramentas de *workflows científicos* e *gateways científicos*, no contexto de *e-science*. Dos diversos *gateways científicos* disponíveis, o Galaxy é o escolhido como plataforma de análise, por conta de sua capacidade, importância e flexibilidade. Dentre as formas de provisionamento de recursos, a computação em nuvem é uma das principais, mas não deve ser a única, sendo considerado um ambiente heterogêneo de oferta de recursos. Apresenta-se por fim a opção pela simulação computacional no desenvolvimento e validação de um ambiente complexo como o desta pesquisa. O Capítulo a seguir mostra o estudo da literatura e a investigação de trabalhos relacionados.

REVISÃO DA LITERATURA

3.1 Considerações iniciais

Neste Capítulo são apresentados o estados da arte em PSP e em *e-science*. São destacadas as evoluções técnicas dessas áreas, para compreensão de seus pontos de convergência, capacidades atuais e necessidades. A seguir é apresentado um levantamento dos trabalhos relacionados e das propostas em sistemas distribuídos para solução dos desafios encontrados.

3.2 Estado da Arte de PSP

Apesar dos avanços alcançados, determinar as funções de proteínas utilizando experimentos biológicos *in vivo* continua sendo um processo demorado e caro. Por esse motivo, técnicas *in silico* despertam interesse de pesquisadores e produzem resultados de qualidade. Essa Seção mostra um levantamento dessas soluções, incluindo utilização de algoritmos de aprendizado de máquina para geração de modelos com bom desempenho, algoritmos multiobjetivo que buscam melhorar a qualidade da predição, o impacto de PSP em pesquisas na Biomedicina, as técnicas para redução de erros em *folding*, o problema de organização de dados de proteínas e as técnicas que envolvem processamento paralelo e distribuído.

Como os procedimentos em PSP envolvem diversas interações e cálculos custosos, os algoritmos genéticos estão entre os utilizados nas pesquisas com proteínas. O trabalho de (FABRE; PULIDO; TELLO, 2013) descreve o modelo *Hidrofóbico Polar* (HP) em *folding* de proteínas, com tratamento multiobjetivo de casos de teste, com bons resultados. Em (CHAMORRO *et al.*, 2013) é proposta uma abordagem evolutiva multiobjetivo para a previsão de propriedades estruturais dos aminoácidos, que gera um conjunto de regras de decisão, capazes de reduzir o tempo de processamento. O trabalho de (KRISHNAJI; RAO, 2013) apresenta uma atualização a um sistema com redes neurais, algoritmos genéticos e lógica *fuzzy* para tratar números eleva-

dos de entradas, alcançando maior precisão de predição. Em (WANG *et al.*, 2013) é proposta uma melhoria em predição com interação *Ribonucleic Acid* (RNA)-proteína, com algoritmo de redução de dimensão que extrai as características essenciais do espaço e que independe do tamanho da amostra. Em (JIAN-WEI *et al.*, 2013) utiliza-se uma rede neural *feedforward*, com reajuste dos pesos a cada rodada para reduzir a complexidade do modelo de treino. A precisão alcançada supera alternativas de *back propagation*. Um classificador para sequências generalizadas é apresentado em (RIZZI *et al.*, 2013) capaz de encontrar similaridades com técnicas como clusterização e mineração de dados, com resultados de boa acurácia. A solução em (OKADA *et al.*, 2014) trabalha com previsão das funções das proteínas baseada em aprendizado de máquina, com melhora da precisão disponível na literatura. O trabalho de (RASHID *et al.*, 2013) introduz um *framework* baseado em algoritmos genéticos com busca local nas populações para melhorar indivíduos gerados aleatoriamente. Os resultados superam soluções híbridas disponíveis. Outro modelo de algoritmos multiobjetivo é descrito (OLIVEIRA; OLIVEIRA; TINÓS, 2015) para adaptabilidade de código genético, para identificar as regiões com provável melhor qualidade de código. Os trabalhos descritos demonstram a variabilidade de alternativas para redução de complexidade e aumento da precisão em PSP. A literatura atual dispõe de algoritmos mono-objetivo em sua maioria, que apresentam menor grau de adaptação.

Têm destaque na Bioinformática as pesquisas focadas em saúde, como estudo de doenças, descoberta de medicamentos e melhoria em sistemas *e-health*. Em (KAVITHA; SARITHA; VINOD, 2013) é apresentado um modelo de mineração de dados, baseado em *Random Forest* destinado a prever a presença e localização dos pontos de ligação do anticorpo ao antígeno no sistema imunológico humano, com o objetivo de desenvolver vacinas. Em (VIJAYAKUMAR; LAKSHMI, 2013) é descrito um sistema de lógica *fuzzy* para avaliar as estruturas de proteínas relacionadas à alergias, um dos crescentes problemas de saúde no mundo. Alguns trabalhos buscam desenvolver medicamentos, especialmente com atuação mais precisa e localizada no corpo humano. (KOOHI, 2013) propõe um método de previsão da interação droga-proteína utilizando filtragem colaborativa, com potencial para acelerar a descoberta de fármacos. O trabalho de (QASIM *et al.*, 2013) destina-se a identificação de alvos dos medicamentos por meio de aprendizado de máquina e mineração de dados. (KAZEMIAN; YUSUF, 2014) trabalha com análise de previsão transmembrana, já que a maior parte da concepção de novas drogas requer o uso de proteínas de membrana. O sistema *Adaptative Neural Fuzzy Inference System* (ANFIS) é capaz de prever e analisar hélices de membrana em sequências de aminoácidos com alta precisão. Por último, (LIU *et al.*, 2014) trata de um *framework* para previsão da estruturas 3D de complexos proteína-proteína, essencial para a compreensão de sistemas biológicos e concepção de medicamentos. O impacto que pesquisas semelhantes a essas podem gerar na área da saúde em diagnósticos, terapias e prevenção mostram a importância dessas propostas. A variabilidade das técnicas aponta para a necessidade de disponibilização de métodos e ferramentas de integração.

Uma das preocupações em PSP são o *overfitting* e o *underfitting* dos classificadores envolvidos no processo de *folding*, além de erros de predição. Os trabalhos a seguir lidam com

tais problemas. (PHUOC; KIM, 2011) propõe um método que analisa a distância entre os pontos de treinamento para agrupá-los em espaços onde há maior densidade de dados, de forma a evitar os comportamentos irregulares e aproveitar corretamente os recursos disponíveis. Os autores em (WAN *et al.*, 2013) propõem um método com aplicação de *Random Projection* (RP) para a construção de um conjunto classificador, com uso de SVM para redução de dimensão no espaço de informações, que melhora a predição e reduz *overfitting*. Para minimizar erros de predição (LUCENA *et al.*, 2013) desenvolve um modelo de regressão linear para previsão da concentração de proteína de trigo, medindo centenas de variáveis correlacionadas com as propriedades físico-químicas e que pode ser utilizada para estimar a concentração de proteína. Uma das estruturas cruciais para as proteínas apresentarem suas características bioquímicas, são as cadeias laterais. O empacotamento de tais cadeias é uma tarefa necessária para PSP (FACCIOLI, 2012). O trabalho de (QUAN *et al.*, 2013) apresenta uma abordagem paralela de empacotamento de cadeias laterais com otimização de colônia de formigas, que cooperam para encontrar as melhores cadeias, alcançando boa performance. Os trabalhos listados demonstram diversos desafios para a melhoria de qualidade em PSP e indicam que esforços colaborativos podem ser necessários em muitos casos.

A produção e obtenção de metadados de proteínas envolve diversos esforços. (YEH; CHEN; OUHYOUNG, 2005) usa contraste de estruturas para recuperação de informações em diversos servidores, reduzindo redundância. (PONMARY; RAJ; SHARMILA, 2007) fornece *softwares* para conversão e pesquisa, para melhoria no acesso aos dados. Em (CHENG; CHEN; KAO, 2010) é desenvolvido um sistema para pesquisar proteínas com estrutura semelhante, mesmo que de famílias diferentes. Tais proteínas realizam diferentes funções e seu mapeamento auxilia as demais pesquisas. O trabalho de (MANDAL; DAS; BHATTACHARJEE, 2012) extrai informações do campo anotação de arquivos PDB para geração de arquivos no formato FASTA, facilitando o trabalho de outros pesquisadores. (TAHA; YOO; ZAABI, 2014) propõe um sistema classificador chamado *Improving Protein Function Prediction Iteratively* (iFPFi) que prevê funções de proteínas não anotadas, além de permitir a compreensão dos relacionamentos entre os genes. Outros dois sistemas de recuperação de informações de proteínas são apresentados em (WANG; LIU, 2013) e em (LU *et al.*, 2014), onde ambos apontam a dificuldade de coleta devido à estrutura complexa e ao tamanho dos arquivos envolvidos. Uso de metadados nos servidores de armazenamento e buscas por similaridade são propostas de melhoria nesse ponto de limitação, além de considerar integração de bases de dados de forma mais aberta e acessível.

Devido ao elevado volume de processamento requerido, diversas abordagens em PSP utilizam técnicas de paralelização. O trabalho de (THOMAS *et al.*, 2005) apresenta um experimento de paralelização utilizando *Standard Template Adaptive Parallel Library* (STAPL) comparando com algoritmos clássicos que alcança resultados expressivos. Os autores em (SOVIC *et al.*, 2010) apresentam um *software* denominado *Parallel Protein Docking Tool*, uma implementação *Message Passing Interface* (MPI) de técnicas sequenciais e uma ferramenta de visualização, em um esforço para redução de tempo de resposta. O emprego de *soft computing*

em PSP é discutido em (ROUT; MISHRA; MISHRA, 2017) com uma extensa revisão. São listadas diversas tecnologias em Redes Neurais Artificiais e suas aplicações, além dos diversos desafios em aberto. Na mesma linha um *survey* é apresentado em (MARQUEZ-CHAMORRO *et al.*, 2015), que discute 90 técnicas diferentes e sua aplicação em PSP. O foco dessa revisão é comparar as técnicas de predição, entretanto dela também se extraem informações de ambiente de processamento e necessidades em geral, devido à complexidade e ao volume de dados. Em (BAI; YANG, 2017) busca-se um modelo unificado de *deep learning* para melhorar a qualidade da predição com análise de similaridade de estruturas sobre diversos aspectos. O desempenho dessa abordagem é superior a outras modelagens disponíveis. O uso de *docking* em PSP é tratado em (BARRADAS-BAUTISTA *et al.*, 2017). Essa estratégia é interessante para o compartilhamento de dados e de análises, podendo complementar o modelo de *workflows*. Nesse aspecto, as soluções propostas são em sua maioria intimamente dependentes de ambientes distribuídos, de forma que soluções de infraestrutura podem ser um complemento necessário.

Uma arquitetura de co-processadores para *folding* de proteínas é discutida em (KHOJA *et al.*, 2006), focado em *clusters* de computadores. Essa abordagem consegue melhorar o desempenho com o uso de *pipelines*, porém limitada pelos processadores disponíveis à época. Mesmo ambientes mais restritos como FPGA podem processar PSP, como discutido em (CHRYSSOS *et al.*, 2012), porém sua capacidade é reduzida, devido aos recursos mais modestos. Na pesquisa (SHARMA; PAPANIKOLAOU; MANOLAKOS, 2013) é apresentada uma implementação para trabalhar com o *Single-Chip Cloud Computer* (SCC), incluindo *framework* e bibliotecas específicas, que consegue bom *speed up*, porém apresenta limitações na alocação de memória. Algoritmos avançados também sofrem com tamanho do espaço de busca das estruturas de proteínas. (LI; LIU, 2013) detalha o emprego de Hadoop em PSP, demonstrando a vantagem dessa utilização sobre um nó individual. (YANG; ZHANG, 2015) descreve o *I-TASSER*, um servidor voltado para processamento de PSP, que apresenta boa capacidade de hospedagem de projetos, entretanto com limitações diversas desde métodos científicos até a capacidade do ambiente. Em (KUNHI; RAJU; CHIPLUNKAR, 2016) usa-se *deep learning* em ambientes paralelos com *Compute Unified Device Architecture* (CUDA) para obtenção de melhora de desempenho. O trabalho (DUBEY *et al.*, 2016) descreve um algoritmo paralelo para GPU que obtém melhora desempenho. Ambas as soluções apresentam alta complexidade e são restritas a um problema.

Essa seção considera diversas abordagens computacionais em PSP, alguns dos desafios encontrados e limitações gerais do campo, especialmente com relação à capacidade computacional. Para efeito de comparação, a Tabela 5 apresenta uma amostra dos trabalhos aqui citados, suas abordagens e as necessidades de infraestrutura dos mesmos.

Tabela 5 – Amostra de trabalhos em PSP e suas necessidades principais.

Trabalho	Problema	Abordagem	Necessidade
(FABRE; PULIDO; TELLO, 2013)	Precisão em PSP	Multiobjetivo	Recursos
(KRISHNAJI; RAO, 2013)	Precisão em PSP	Híbrida	Recursos
(WAN <i>et al.</i> , 2013)	<i>Misfolding</i>	SVM	Recursos
(QUAN <i>et al.</i> , 2013)	<i>Misfolding</i>	Ant Colony	Recursos
(KAVITHA; SARITHA; VINOD, 2013)	<i>e-health</i>	R. Forest	Recursos
(QASIM <i>et al.</i> , 2013)	<i>e-health</i>	Híbrida	Recursos
(CHENG; CHEN; KAO, 2010)	Dados de PSP	Pesquisa	Integração
(TAHA; YOO; ZAABI, 2014)	Dados de PSP	Classificação	Integração
(LU <i>et al.</i> , 2014)	Dados de PSP	Anotação	Integração
(LI; LIU, 2013)	Desempenho	HADOOP	HPC
(MARQUEZ-CHAMORRO <i>et al.</i> , 2015)	Desempenho	Diversas	HPC
(YANG; ZHANG, 2015)	Desempenho	Diversas	HPC
(KUNHI; RAJU; CHIPLUNKAR, 2016)	Desempenho	CUDA	HPC

Fonte: Dados da pesquisa.

Nota – Na coluna “Necessidade” a legenda “Recursos” refere-se à limitações de processamento, “Integração” à unificação de sistemas e bases e “HPC” ao acesso à DCI.

3.3 Estado da Arte de *gateways* científicos

As pesquisas em *e-science* combinam esforços na troca de informações, integração de plataformas e interdisciplinaridade. As inovações passam pelos SWMS, *gateways* científicos, *frameworks* e demais plataformas de convergência. Os *gateways* científicos estão avançando nessa área, onde as comunidades de usuários são fundamentais (ARMBRUST *et al.*, 2009), existindo exemplos em domínios tão diversos como meteorologia (SANSOM, 2010), agricultura (DRIEMEIER *et al.*, 2014) e biodiversidade (MATSUNAGA *et al.*, 2013). A pesquisa de (GESING *et al.*, 2017) apresenta uma visão geral sobre os *gateways*, sua construção e utilização. Segundo os autores, a falta de expertise pode ser um obstáculo importante para acesso a esses recursos, já que nem todo pesquisador tem entendimento de procedimentos computacionais avançados. Otimizações e estratégias de provisionamento de recursos são fundamentais para o bom funcionamento desses ambientes complexos, que devem atender necessidades de múltiplos grupos e adequar-se a uma gama de opções (SCHADT *et al.*, 2010).

Workflows científicos são parte fundamental da divisão de tarefas, organização de dados e reprodução de experimentos, sendo gerenciados principalmente por meio de SWMS. Com pesquisas voltadas para dados intensivos nos mais diversos contextos, os SWMS têm ampliado sua participação na análise, compartilhamento, uso de *Graphical User Interface* (GUI) e reprodutibilidade de experimentos (LIU *et al.*, 2015). A Tabela 6 apresenta um levantamento dos principais SWMS, seus mecanismos, modelos de dados e principais aplicações.

Diversas tecnologias podem ser integradas a esses ambientes para melhorar o funcionamento e o desempenho. A redução de discrepâncias entre o *workflow* descrito e sua estrutura

Tabela 6 – Tabela comparativa dos SWMS mais difundidos.

SWMS	Estrutura de Workflow	Escalonamento Estático	Partilha de dados	Interface do usuário	Detalhes
WSPGRADE/gUSE	DAG	Sim	Sim	GUI	Permite execução em infraestruturas distribuídas
Pegasus	DAG	Sim	Não	Texto	Alta performance e escalável
Kepler	DAG e DCG	Sim	Não	GUI	GUI baseado em desktop
Chiron	DAG e DCG	Não	Não	Texto	Paralelização com abordagem algébrica
Askalon	DAG e DCG	Não	Sim	GUI	Execução em um ambiente de nuvem híbrido
Galaxy	DAG e DCG	Não	Sim	GUI	Sistema web para pesquisa genômica
SciCumulus	DAG	Não	Sim	Texto	Baseado em nuvem Escalável
Swift	DAG e DCG	Não	Não	Texto	Alta performance e escalável
Taverna	DAG	Não	Sim	GUI	GUI baseado em desktop
Triana	DAG e DCG	Não	Não	GUI	Serviços P2P inclusos

Fonte: Adaptada de [Khan, Shakil e Alam \(2017\)](#).

é avaliada em ([GROTH; GIL, 2009](#)), com propostas de melhorias na descrição de linguagem natural. ([SONNTAG; KARASTOYANOVA; DEELMAN, 2010](#)) propõe uma tecnologia de meio entre *workflows* de negócio e científicos. ([KACSUK, 2011](#)) descreve o *framework* P-grade, para integração de *workflows* em grade, cujo serviço encontra-se indisponível atualmente. ([MORENO; TIRADO; SALAZAR, 2013](#)) identifica gargalos em aquisição e interpretação de dados de *work-flow* e apresenta uma biblioteca para o processamento de longas sequências de proteínas. O artigo ([OLABARRIAGA et al., 2014](#)) discute o crescimento dos SWMS, com altas demandas, a manutenção complexa e os recursos insuficientes, propondo que mais níveis de abstração podem auxiliar na criação de *gateways* robustos. ([BIANCHI et al., 2016](#)) propõe um SWMS para *Next Generation Sequencing* (NGS), com módulos de monitoramento e análise de rastreamento. Esses trabalhos demonstram os muitos aspectos a serem aperfeiçoados em *e-science* e o alcance que as ferramentas têm tido nos procedimentos científicos modernos.

O impacto dos *gateways* científicos tem sido alvo de diversas pesquisas. Em ([WILKINS-DIEHR et al., 2008](#)) é analisado o TeraGrid, onde se sugere a adoção de opções diversificadas para HPC. Os *gateways* têm aplicações para agrupamento multidisciplinar de pesquisas ([SHEN;](#)

FAN; PAMIDIGHANTAM, 2014), análise de grandes volumes de dados (AGUILERA *et al.*, 2016) e simulações pesadas como as de dinâmica dos fluidos (SMITH; ABEYSINGHE, 2017). No *survey* (LIU *et al.*, 2015) é apresentada uma revisão completa das necessidades e dos aspectos de computação sob demanda em *gateways* científicos. O levantamento aponta lacunas na interoperabilidade, tendência ao provisionamento estático, o desafio do escalonamento multi-nuvem e as tecnologias para encapsulamento de complexidade. Os autores consideram que na atualidade, a maioria dos *gateways* científicos apenas oferece extensões de modelos já usados em grades e alguns *middlewares* de interação. Poucos trabalhos avaliam requisitos dos usuários para tomada de decisão e revela necessidades atuais como variedade no provisionamento de recursos, balanceamento de carga e estratégias de escalonamento.

Como o escalonamento de *workflows* em HPC é um desafio em aberto, diversas abordagens são propostas. O trabalho (FARD; DELDARI, 2008) propõe o escalonamento de tarefas dependentes em grades, mas seu modelo de economia não chegou a ser adotado de forma significativa. Em (PRODAN; WIECZOREK; FARD, 2011) uma abordagem de otimização de oferta de grades e nuvens apresenta melhoria marginal de desempenho. Os autores em (FARD *et al.*, 2011) buscam a redução de *makespan* em nuvens pagas com a pesquisa de melhores ofertas e desenvolvem na sequência da pesquisa um escalonador estático multiobjetivo no *gateway* Askalon, melhorando soluções bi-critério. A pesquisa em (DURILLO; FARD; PRODAN, 2012) usa escalonamento de lista para *workflows* baseado no modelo de Pareto, alcançando redução de *makespan* e custos em nuvem, apesar de não testar em nuvens comerciais. Em (FARD; PRODAN; FAHRINGER, 2013) é analisado o escalonamento dinâmico de *workflows* em nuvens comerciais, reduzindo tempo de resposta, mas indica que a consideração de custos seja impraticável. Desenvolvem a seguir um modelo de provisionamento em nuvem com orçamento restrito, porém o sistema multiobjetivo apresenta falhas. Os mesmos autores desenvolvem um escalonador em lista multiobjetivo para Askalon, distribuindo para ambientes heterogêneos (grade e nuvem), que indica bons resultados. Na pesquisa (FARD; RISTOV; PRODAN, 2016) busca-se a redução de incerteza de provisão em nuvem, aumentando os recursos dinamicamente. O foco na redução de *makespan* afeta o custo e a solução multiobjetivo tem desvios significativos. Em (ARABNEJAD; BARBOSA, 2017) é proposto um algoritmo de escalonamento com QoS para *clusters* heterogêneos, que apresenta bons resultados, apesar de não considerar requisitos como consumo e tolerância a falhas.

A proveniência de dados é fundamental para a compreensão dos ambientes e para reprodução de experimentos. Nesse contexto, o trabalho (MARINHO *et al.*, 2012) traz o ProvManager, uma abordagem de proveniência e análise de dados, capaz de se adaptar a vários SWMS. (BOURHIS; DEUTCH; MOSKOVITCH, 2016) propõe um modelo baseado em proveniência para desenvolvimento de algoritmos eficientes para aplicações centradas em dados. (ALPER; BELHAJJAME; GOBLE, 2016) trata das diferenças entre proveniência manual e automática, e a qualidade dos dados obtidos. Os mesmos autores analisam *workflows* do Taverna para predição de padrões de proveniência, capaz de identificar erros e faltas nos dados disponíveis.

Em (MARINHO *et al.*, 2017) é proposto um modelo algébrico de extração de *workflows* que prevê a reprodutibilidade e rastreamento de experimentos. Em (SILVA *et al.*, 2017) é oferecida uma ferramenta de auxílio à análise de dados brutos, facilitando identificação de pontos de interesse.

A reutilização de *workflows* científicos é facilitada pelos SWMS. O reajuste de parâmetros pode ser difícil, necessitando de mecanismos automatizados. Os *Research Object* (RO) são usados no gerenciamento de *workflows*, sendo definidos como um recurso que agrega outros, tais como conjuntos de dados e *softwares*. (ROURE *et al.*, 2011) propõe um *framework* para a preservação de dados com RO para evitar a desintegração do *workflow*, quando ocorrem mudanças externas. Em (HOLL *et al.*, 2013) é proposta a captura de metadados de otimização, estendendo o modelo RO, com estudo de caso real em *e-science*. Em (HETTNE *et al.*, 2014) é explorada a descrição semântica de ROs, com melhoria na disponibilização de informações. Em (COHEN-BOULAKIA *et al.*, 2014) são propostos métodos de redução da complexidade estrutural para recuperação de dados. Os autores em (MATTOSO *et al.*, 2015) apresentam um *survey* sobre redirecionamento dinâmico de *workflows* em HPC, discutindo a necessidade de ferramentas de suporte e comparando técnicas. A maioria delas disponibiliza a análise apenas no fim da execução do *workflow* e não permite alterações durante o experimento, as quais poderiam gerar redução de passos e de repetições, criando ajustes mais precisos.

Em *e-science* a estrutura de hospedagem de recursos é variável. O consórcio EU Cancer-Grid de (KOVÁCS; KACSUK; LOMAKA, 2011) permite a criação de grade de *workstations* privados de instituições parceiras, usando WS-PGRADE, com supervisão humana mínima. Esse *gateway* científico é um exemplo da busca por alternativas nos modelos de provisão de recursos. O trabalho de (OLIVEIRA *et al.*, 2012) apresenta um modelo adaptativo de execução de *workflows* em nuvem, que avalia disponibilidade e ajusta parâmetros dinamicamente. Em (OLIVEIRA *et al.*, 2013) é feita uma análise de desempenho de estratégias paralelas em nuvem. O modelo é comparado com técnicas em Hadoop, apresentando resultados superiores. O trabalho de (KOZLOVSZKY *et al.*, 2014) propõe o *DCI-Bridge* um *middleware* para atuação entre *gateways* científicos, entretanto essa proposta tem limitações na integração com os *datacenters* e falhas de escalonamento. O portal MoSGrid (KRÜGER *et al.*, 2014) oferece simulações moleculares de alta qualidade, porém requer melhorias no provisionamento de recursos, na conversão de formatos de dados e no mapeamento dos metadados. Em (BALIS, 2016) é descrito um modelo computacional para *workflows* complexos em ambiente distribuído, oferecendo ferramentas de alto nível para desenvolvedores, facilitando a criação de fluxos descentralizados. (HE, 2016) trabalha com reserva de recursos para máquinas virtuais em nuvem, porém sem avaliações do processo de decisão. (ZHAO; PASCHKE; ZHANG, 2016) considera ambientes com múltiplos participantes propondo agentes de recomendação com regras de contexto, cuja complexidade acaba por limitar o modelo. SEAGrid é um *gateway* científico que permite a instalação em diferentes infraestruturas de computação (NAKANDALA *et al.*, 2016). Promissor, ainda tem limitações na transferência de arquivos, armazenamento e extração de dados. (PIERANTONI *et al.*, 2017) discute o uso de infraestruturas distribuídas no ambiente de *gateways* científicos, buscando

alternativas aos sistemas de grupos específicos. Em (ZHENG *et al.*, 2017) se empregam vários algoritmos de escalonamento em nuvem de *workflow*. A solução é limitada por considerar apenas soluções pagas. Todas essas considerações são exemplos da dificuldade de provisionamento para ambientes científicos de grande volume de dados e da amplitude de possibilidades.

As alternativas para modelos de sistema que simplifiquem o trabalho dos usuários são várias. (BALASKO; FARKAS; KACSUK, 2013) implementa um *gateway* científico genérico com caixas pretas, que permite a desenvolvedores oferecer novos portais, adaptando APIs e interfaces do gUSE. Em (HUMPHREY *et al.*, 2013) são propostos modelos de segurança para soluções Amazon e Azure, com prós e contras de cada servidor e as vantagens deste sistema. Em (SILVA *et al.*, 2014) os autores discutem a informação semântica no estudo biomédico e propõem uma padronização por meio de ontologias. Os autores em (CHARD *et al.*, 2014) mostram as dificuldades de manter aplicações de *e-science* devido ao número de usuários, tecnologias e protocolos. *Workflows*, escalabilidade, confiabilidade e interface amigável são destacados. Os autores em (HIDEN; WOODMAN; WATSON, 2016), defendem uso da nuvem para pesquisas relacionadas à saúde, fornecendo estimativa de preços com previsão de tempo de processamento. O aplicativo Galaxy Portal é apresentado em (BORNICH *et al.*, 2016), uma tentativa de fornecer acesso via dispositivos móveis. (PATTERSON *et al.*, 2017) propõe um repositório aberto para pesquisas em esclerose múltipla e uso de *deep learning* para extração de *workflows* de alto nível, com informações sobre fluxo, métodos estatísticos e dependências. Também nesse aspecto, diversas abordagens têm sido propostas, nas quais a possibilidade de prover conteúdo para múltiplos usuários é um objetivo principal, sem solução única.

A escalabilidade de *workflows* é discutida em diversas pesquisas. O trabalho (LI *et al.*, 2013) considera a distribuição de tarefas de SWMS em *clusters* que apresenta bons resultados. (KELSEY; MCCAFFERY; KOTTHOFF, 2014) apresenta uma estrutura de *e-science*, com uso de inteligência artificial para processamento em infraestruturas de menor capacidade. Em (CHARD *et al.*, 2015) é apresentado um provedor de recursos elástico para melhorar custos e tempo de processamento. Essa solução apresenta custo-benefício ruim entre recursos e valores. (STITZ *et al.*, 2015) critica a falta de padronização de nós virtuais e propõe um sistema de auxílio a administração. O trabalho de (ZHAO *et al.*, 2015), integra SWMS em ambientes em nuvem para aumento de recursos, porém a solução contempla exclusivamente soluções comerciais. (GUGNANI *et al.*, 2016) explora as vantagens dos *gateways* científicos e da computação em nuvem, com um *framework* de integração para *big data*, aumentando a capacidade de processamento. Em (GARZA; AICHELER; KOHLBACHER, 2016) e (GARZA *et al.*, 2016) é explorada a execução em diversos ambientes, de *workstations* a grades. Essa abordagem não considera a influência da carga de trabalho e sua relação com a capacidade de computação. Em (MALIK; SAINI; RANI, 2017) se busca otimizar a alocação de recursos visando economia de energia e mantendo QoS, mas a complexidade dessa resolução ainda é alta.

Sobre o comportamento da carga de trabalho destacam-se algumas pesquisas. (AHMAD;

LEITNER; KRIEGER, 2015) apresenta um modelo de monitoramento de estados, com baixo *overhead* de utilização em redes de sensores e dados de grande volume, entretanto apresenta falhas na agregação de dados. Em (PODZIMEK *et al.*, 2016) se discute as dificuldades de prever comportamento de carga de trabalho e utiliza cargas parciais ou reexecução para balanceamento. A solução porém requer testes com cargas maiores e diversificadas. Em (GUPTA; AZHARUD-DIN; JANA, 2016) o algoritmo escolhe as maiores tarefas e os ambientes com execução mais rápida para reduzir o *makespan*, porém não considera custos ou balanceamento, sendo viável para estudos de hipóteses. (MISHRA; RAO, 2016) descreve o uso de diversos algoritmos evolutivos em *Job Shop Scheduling*. As soluções são testadas em múltiplos *benchmarks* e os autores propõe algoritmos híbridos para melhoria de desempenho. O trabalho de (GUSEW; SCHEUERMANN, 2016) apresenta uma estratégia para o início precoce de tarefas não conectadas, transformando o DAG e mapeando etapas executáveis, capaz de reduzir o tempo de processamento. (BASSEM; BESTAVROS, 2017) empacota detalhes de virtualização e de aplicações para um *broker* em nuvem que alcança eficiência, mas que precisa evoluir nas estratégias de alocação de recursos. Essas pesquisas são relativamente recentes e contam com diversos pontos de contribuição em aberto.

O armazenamento de *workflows* e seus resultados é um problema que pode ser abordado por diferentes ângulos. O trabalho de (LIU *et al.*, 2014) integra Galaxy com o Globus Transfer¹ para movimentação confiável de dados. A solução tem limitações de segurança e manutenção de *Service Level Agreement* (SLA). Em (HAZEKAMP *et al.*, 2015) discute-se a capacidade de SWMS, quanto ao paralelismo e número alto de execuções. A integração do *Work Queue* fornece o despacho de tarefas e armazenamento de arquivos em *cache*, útil para o crescente nível de requisitos. (HAZEKAMP *et al.*, 2017) propõe um modelo combinado de armazenamento dos resultados e informações, com soluções estáticas e dinâmicas. Os autores em (GRÜNING *et al.*, 2017) discutem as limitações do Galaxy em processos posteriores à produção dos dados brutos e usam uma plataforma híbrida para análise de dados, extração de informações e reprodutibilidade.

A crescente demanda por alta capacidade e as necessidade de integração de tecnologias geram oportunidades de pesquisa em *e-science*. O levantamento descrito mostra algumas das limitações e as soluções propostas. Uma coleta de alguns dos trabalhos discutidos nessa seção é apresentada na Tabela 7, onde são descritos o foco principal, a abordagem e a limitação dos mesmos.

3.4 Trabalhos Relacionados

Nessa seção são discutidos os trabalhos que abordam problemas em *e-science* e buscam soluções em com métodos de sistemas distribuídos. Os *gateways* científicos têm evoluído e se diversificado, porém destacam-se vários desafios ainda sem solução, especialmente em agregação

¹ <https://www.globus.org/data-transfer>

Tabela 7 – Amostra de trabalhos em *gateways* científicos e suas necessidades principais.

Trabalho	Foco	Abordagem	Limitação
(OLIVEIRA <i>et al.</i> , 2013)	Escalabilidade	Nuvem	Integração
(ZHAO <i>et al.</i> , 2015)	Escalabilidade	Nuvem	Custos
(MATTOSE <i>et al.</i> , 2015)	Otimização	Híbrida	Complexidade
(HIDEN; WOODMAN; WATSON, 2016)	Usabilidade	Recomendação	Integração
(GUPTA; AZHARUDDIN; JANA, 2016)	Carga de trabalho	Particionamento	Balanceamento
(KOZLOVSZKY <i>et al.</i> , 2014)	Provisionamento	Heterogênea	Escalonamento
(CHARD <i>et al.</i> , 2015)	Provisionamento	Nuvem	Transferência
(ZHAO; PASCHKE; ZHANG, 2016)	Provisionamento	Recomendação	Complexidade
(GARZA; AICHELER; KOHLBACHER, 2016)	Provisionamento	Heterogêneo	Escalonamento
(HAZEKAMP <i>et al.</i> , 2017)	Armazenamento	Estática	Integração
(FARD; PRODAN; FAHRINGER, 2013)	Escalonamento	Dinâmico	Custos
(FARD; RISTOV; PRODAN, 2016)	Escalonamento	Multiobjetivo	Custos
(ZHENG <i>et al.</i> , 2017)	Escalonamento	Híbrido	Custos
(ARABNEJAD; BARBOSA, 2017)	Escalonamento	Heterogênea	Consumo

Fonte: Dados da pesquisa.

de pesquisas e modelos de provisionamento. Plataformas SOA se desenvolveram nos últimos anos, sendo adaptadas para necessidades atuais. Elas têm sido adotadas em áreas diversas como agricultura de precisão (SRINIVASULU *et al.*, 2017), *e-health* (CHAIM; OLIVEIRA; ARAUJO, 2017), IoT (GUO; ZHU; YANG, 2016), *Smart Cities* (CLEMENT; MCKEE; XU, 2017) e *Deep Learning* (WANG *et al.*, 2017), demonstrando sua flexibilidade. A adoção de SOA é efetiva devido à simplicidade de integração de serviços e reduções de custo devido à reutilização. Enfrenta ainda desafios relacionados ao desempenho e à QoS.

Há dez anos, quando o conceito de *e-science* era menos difundido, os autores em (LIU; YANG; LI, 2008) discutiam as limitações da computação em grade com relação à integração, propondo um modelo de SOA como plataforma de serviços padronizados. Desde então a tecnologia permitiu novas abordagens. (BOSIN *et al.*, 2010) descreve o papel de SOA em *e-science*, explorando a integração de grades e *workstations*. Esforço semelhante é discutido em (XU *et al.*, 2010) que propõe uma SOA para construção de infraestrutura de *e-science* para estudo de biodiversidade. Esses trabalhos propiciam a expansão do modelo para outros domínios.

A organização de informação e de recursos conta com várias abordagens baseadas em SOA. O BioCatalogue² é um repositório de *Web services* focados em Bioinformática. O sistema segue uma abordagem em que os serviços podem ser registrados, pesquisados e documentados por qualquer membro da comunidade, via módulos de descoberta, registro, anotação e monitoramento (BHAGAT *et al.*, 2010). Em (LIU; GORTON; WYNNE, 2013) é apresentado um *framework* para compor *workflows* adaptativos, com mecanismos de encapsulamento, que entretanto foi testado em um modelo limitado de *workflows*. O trabalho (CRICHTON *et al.*, 2014) traz o *Planetary Data System*, uma arquitetura multidisciplinar com SOA e banco de dados escaláveis. A arquitetura trouxe flexibilidade e escalabilidade para as pesquisas, com potencial de crescimento.

² www.biocatalogue.org

O trabalho de (PRASETYO; BANDUNG, 2015) propõe uma estrutura baseada em *Service-Oriented Architecture Decision* (SOAD) para modelagem de decisão em engenharia de requisitos, dadas as limitações dos modelos para integração de *software* e manutenção do conhecimento. Em (LU; KOGA, 2015) é discutida uma SOA para controle de tráfego aéreo global, com recursos escaláveis e estruturados, ainda em desenvolvimento. Em (VILLANUEVA-ROSALES *et al.*, 2015) é descrito o *framework* ELSEWeb, que visa a integração de dados ambientais em um esforço multidisciplinar. Tais trabalhos são amostras dos pontos em que SOA pode atuar para trazer soluções de qualidade.

Pesquisas da última década mostram a busca de eficiência em aplicações SOA em integração de sistemas, plataformas e tecnologias. No setor industrial são encontrados os exemplos de (HUIYANG; MEINA; JUNDE, 2007), (KUMARI; KANDAN; MISHRA, 2008), (QIFENG; ZHANGJIAN, 2011), (ELIASSON *et al.*, 2013), (MALARVIZHI; KALYANI, 2013) que envolvem troca de informações entre setores, manutenção de sistemas legados, monitoramento em tempo real e redução de custos. Em (FELJAN *et al.*, 2015) é apresentado um sistema de execução em sistemas *cyber*-físicos de larga escala, no qual a avaliação de custo precisa ser estendida. (OZDOGAN; AKGUN; KAYA, 2017) apresenta uma SOA para descoberta de serviços, com interface de busca metadados, gerenciamento do desenvolvimento e análise de dependências. Uma estimativa de custos (financeiro, tempo, manutenção, etc.) em SOA é vista em (ALNAHDI; LIU, 2017), considerando todas as partes (usuário, provedor, mediador) e os papéis dos componentes. Esses artigos fornecem padrões e conceitos para pesquisas correlatas.

Desafios de utilização se devem à complexidade e às características dos serviços distribuídos. (IBRAHIM; MOHAMMED, 2012) mostra um levantamento de arquiteturas de integração de *software*. Alguns dos maiores desafios encontrados pelos pesquisadores são as constantes atualizações de tecnologia, sistemas legados, volume de atividades e movimentação de dados. A integração de sistemas requer grandes esforços de programação e refatoração. Exemplos de trabalhos que buscam esse empreendimento são os de (BENEVENTANO *et al.*, 2009) que usa ontologias para integração de bases de dados, (LI; SONG, 2010) com *Web Services* aplicados em mineração de dados, (BENHADDI; BAINA; HASSAN, 2013) com linguagem de integração *Enterprise Integration Patterns* (EIPS) e SOA centrada no usuário, (YANG, 2013) com uma SOA de transações inter-departamentos, (EBENHOFER *et al.*, 2013) que promove ferramentas para integração e (BANHALMI *et al.*, 2013) que usa a camada de *Data Access Object* (DAO), gerada de acordo com ontologias. Via de regra, os autores desses trabalhos optam por SOA porque permite escalabilidade, reutilização e facilidade de manutenção. Em (WANG *et al.*, 2009) é apresentado um modelo dinâmico para integração de bases de dados baseado em SOA. (ANEES; ZEILINGER, 2014) propõem uma SOA com análise do impacto de monitoramento sobre o desempenho de serviços *Web*. Em (BROWN; GROLINGER; CAPRETZ, 2011) é apresentado um *framework* para a integração de bases de dados médicos. Os conceitos desses trabalhos são bastante promissores, ainda que sejam necessários testes mais abrangentes e a continuidade da pesquisa, em diversos casos.

A computação orientada a serviços permite a publicação de dados e recursos computacionais, promovendo a descoberta de ferramentas. A pesquisa de (BAHRAMI; SINGHAL, 2015) discute a falta de padronização e baixa interoperabilidade dos modelos de nuvem e o crescimento das ofertas em *aaS*, que complicam o modelo ao aumentar o número de camadas. Propõem uma camada de *template* para ordenar a criação das demais, padronização multi-nuvem e uma arquitetura personalizável em SOA. Em (HUANG, 2016) é apresentada uma modelagem de SOA para suporte à decisão no contexto de *e-health*. Os autores em (KUO *et al.*, 2016) discutem as limitações de SOA e *Web Services*, propondo uma plataforma que integra serviços e *workflows*. Esses trabalhos ainda estão em desenvolvimento e têm potencial de evolução nos respectivos campos. Têm limitações ao omitirem o modelo de dados e de provisionamento necessário.

A integração de diferentes sistemas operacionais, o compartilhamento de dados e aplicações heterogêneas pode ser alcançado com o uso de metadados. Para ambientes de tamanho reduzido, modelos baseados em *Enterprise Service Bus* (ESB) podem ser os mais indicados. Os exemplos de utilização dessa tecnologia são descritos a seguir, sendo o primeiro deles (ZHANG; LI; YU, 2006) com um projeto para integração de informação independente de arquitetura. (QING-RUI; HAI-TAO, 2010) implementa integração de aplicativos orientada a serviços, sem alterar o sistema original. Em (CHEN *et al.*, 2010) é proposto o *Army Command and Control System* (ACCS) um *software* de integração para compartilhamento de informações entre sistemas militares. (VRBA; FUKSA; KLIMA, 2014) apresenta uma plataforma de integração multi-agente voltada para empresas de construção de aeronaves. (BAGHERMOUSAVI; RASHIDI; HAGHIGHI, 2014) apresenta uma abordagem para reduzir a sobrecarga da medição do QoS em SOA, com foco em *Small and Medium Business* (SMB). (SHI *et al.*, 2014) fornece um *framework* para consulta e troca de dados entre sistemas. (CHAUDHARI; BHADORIA; PRASAD, 2016) defende a utilização do ESB para monitorar parâmetros de desempenho como tempo de resposta e uso de memória. Em (FERNANDEZ; TEDESCHI; PRIOL, 2016) é proposta uma abordagem para execução descentralizada de serviços que forneça um modelo de alto nível de composição de serviços, com controle de fluxo coordenação de nós. Os trabalhos descritos indicam a necessidade de tecnologias de intermediação e de recursos para troca de informações em ambientes heterogêneos, o que é comum em meios científicos. As lacunas nesse contexto são várias.

Os modelos de provisionamento de recursos que integram SOA e podem ser adotados para *e-science* são diversos. A computação em nuvem tem potencial para provisionamento sustentável e energeticamente eficiente, otimizando o uso de máquinas físicas e a gestão dos recursos com virtualização. Em (MENON *et al.*, 2012) é apresentado um sistema de processamento em nuvem destinado a pesquisadores da genética, visando armazenamento eficiente de dados em grande escala. Autores em (BORGETTO *et al.*, 2012) sugerem uma abordagem de migração e reconfiguração de VMs para economia de energia. Em (ABDUL-WAHID *et al.*, 2012) é proposta a distribuição de processamento entre nuvens públicas, *clusters* e arquiteturas com GPU, considerando escalabilidade e minimização da comunicação. A avaliação do sistema é

pouco abrangente e as sobrecargas são severas em alguns dos casos.

O trabalho de (FARAHNAKIAN; LILJEBERG; PLOSILA, 2014) usa aprendizado de máquina para adaptar o número de nós ativos, com resultados limitados a *datacenters* de pequena dimensão. (PENG *et al.*, 2016) descreve necessidade de monitorar requisitos dos usuários e a alocação de recursos simultâneos, propondo um modelo de pouca amplitude. (YING; LEI, 2014) apresenta escalonamento de tarefas estático e baseado em pré-condições, com bons resultados, porém testes em sistemas mais amplos são necessários. O trabalho de (GAN; ZHENG, 2016) define técnicas baseadas em SVM para escalonamento em nuvem, com foco em custos e tempo de execução, entretanto a descrição do algoritmo, avaliação e aplicação são limitadas. Em (CAMILLO *et al.*, 2016) são discutidas técnicas para restauração de serviços em sistemas distribuídos, com um algoritmo de busca exaustiva e heurística multiobjetivo para encontrar a solução otimizada para eventuais falhas. A eficiência desse modelo é superior a de outras propostas.

Os tipos de plataforma podem variar conforme os objetivos ou o público alvo. Grades descentralizadas são estudadas em (OSTBERG; ELMROTH, 2013), para escalonamento justo baseado em prioridades. O sistema de gerenciamento proposto alcança bom custo-benefício. O artigo (WALLOM *et al.*, 2015a) avalia o uso de *workstations* junto com soluções de contêiner como alternativa para a provisão de recursos e em (WALLOM *et al.*, 2015b) é proposta uma plataforma de padrão aberto para nuvem comunitária que sirva institutos de pesquisa. Um sistema escalável para PSP é mostrado em (MROZEK; GOSK; MAŁYSIAK-MROZEK, 2015), entretanto a solução é de código fechado, voltada exclusivamente para Azure Cloud. Esse trabalho apresenta boa fundamentação, porém os resultados são limitados. (DELDARI *et al.*, 2016) propõe escalonamento estático que agrupa o *workflow* em máquinas conforme a capacidade, voltado para nuvem. (PADMAVATHI *et al.*, 2016) usa agentes autônomos para negociação entre provedores visando reduzir custos, limitado porém na integração de provedores. Em (INDHUMATHI; NASIRA, 2016) é desenvolvida uma SOA com balanço de carga e tolerância a falhas para grade visando reduzir a comunicação e as tarefas em uma única grade. Ainda está em desenvolvimento e focada exclusivamente em grade.

O trabalho em (THAMAN; SINGH, 2016) apresenta uma revisão de escalonamento em nuvem e propõe um escalonador orientado a metas, para diferenciação de ambientes alvo. Também em (BHAGWAN; KUMAR, 2016) é apresentada uma revisão em escalonamento, comparação de técnicas e aponta as mais adequadas para cada objetivo, ressaltando os diversos desafios. (KRIEGER *et al.*, 2017) apresenta um ambiente multi-nuvem com auto-escalonamento para *workflows*. O trabalho de (LYASHOV *et al.*, 2016) discute a criação de ambientes heterogêneos com o uso de SOA fornecendo flexibilidade e facilidade de uso. Essas pesquisas corroboram para a necessidade de técnicas avançadas de provisionamento em *e-science*.

As alternativas para provisão de recursos para Bioinformática vêm sendo estudadas há tempos. Em (MACEDO *et al.*, 2013) é proposta uma política de alocação em *cluster* para

algoritmos paralelos, em que problemas de decisão ressaltam. A execução de múltiplas tarefas no Galaxy foi avaliada em (MAHESHWARI *et al.*, 2013), que integra com o Swift para escalabilidade, porém dificuldades de execução interromperam o empreendimento. Em (GUIMARAES *et al.*, 2014) é tratada a tolerância a falhas em grade para *workflows*, considerando problemas técnicos e falhas na descrição, porém o sistema impõe sobrecarga considerável. Em (SANDES; RALHA; MELO, 2014) é apresentada uma proposta de balanceamento para ambientes multi-nós, baseada em agentes de redistribuição. A solução é sofisticada, mas tem as desvantagens do aumento no volume de mensagens e de dados transferidos. Em (SANDES *et al.*, 2016b) é descrita a arquitetura MASA, uma arquitetura de processamento em plataformas como FPGAs e GPUs. Os bons resultados apresentados demonstram o potencial de processamento de ambientes heterogêneos. A MASA é voltada para aspectos de pesquisa diferentes da proposta dessa tese. (SANDES; BOUKERCHE; MELO, 2016) faz uma revisão de técnicas de HPC em Bioinformática e argumenta que não há soluções simples nesse campo, sendo o mais indicado soluções diversas, heterogêneas e incrementais, o que pode ser encontrado nos ambientes de SD.

As propostas de integração de ambientes computacionais para provisionamento de recursos são várias. O trabalho (KAROCZKAI; KERTESZ; KACSUK, 2016) propõe um *Broker* para DCI, baseado na prioridade das tarefas. O foco principal é medir o tempo de espera para execução, o que afeta o *makespan*, porém essa solução não contempla demais aspectos do *workflow*. Em (MAHESHWARI *et al.*, 2016) é avaliada a execução de *workflows* em *clusters* e em nuvem, que consegue reduzir o *makespan* para alguns casos, entretanto fixa limites para os *workflows* e sofre com cargas mais elevadas. O projeto EUBrazil Cloud Connect (FIORE *et al.*, 2017) fornece em uma infraestrutura de nuvem federada internacional, para pesquisa com dados intensivos. (CABIDDU; ATTENE, 2017) propõe transferências de partes de dados para processamento local e ferramentas *stand-alone*, com bons resultados, porém a infraestrutura tem limitações com reprodutibilidade. (WEI; YEH, 2017) discute economia computacional e confiabilidade em SOA, com alocação de recursos resiliente e custo flexível, aplicado em um *grid* virtual, onde ressaltaram deficiências em requisitos de QoS. (S; RAJENDRAN; RP, 2017) usa programação Linear Inteira com violações globais e penalidade associada, para descoberta de *workflow* e composição de serviços em ambientes multi-nuvem, com bons resultados. O trabalho de (COLLADO, 2017) propõe uma solução para provisionamento de recursos em servidores heterogêneos, considerando também a adição e *cluster* e grade, entretanto a avaliação desse trabalho é pouco detalhada e o modelo não prevê a integração de soluções. Nuvens híbridas são tratadas em (TOOSI; SINNOTT; BUYYA, 2017), com adição de nuvens públicas quando necessário. Essa solução é testada apenas em nuvens comerciais, entretanto. Em (GOHER *et al.*, 2018) é proposto um sistema de permuta de recursos agrupados em IaaS inter-nuvem, focado em alavancar provedores modestos, capazes de atender altas demandas de forma adaptativa. O trabalho de (PANADERO *et al.*, 2018) propõe o método *Multi-Criteria Biased Randomized* (MCBR), para seleção confiável de nós em sistemas de grande escala. A seleção é feita rapidamente, exigindo pouca potência computacional.

Tabela 8 – Amostra do levantamento de trabalhos relacionados

Trabalho	Abordagem	Avaliação
(LIU; YANG; LI, 2008)	Uso de SOA para integração de grades	Limitado a grade. Ultrapassado
(BOSIN <i>et al.</i> , 2010)	Vantagens SOA em e-science	Avaliação inicial. Pouco abrangente.
(INDHUMATHI; NASIRA, 2016)	Balanceamento de carga e tolerância a falhas	Restrito a grade.
(FELJAN <i>et al.</i> , 2015)	Execução em sistemas de larga escala	Focado em dispositivos de IoT
(OZDOGAN; AKGUN; KAYA, 2017)	Interface de metadados em SOA para ambientes empresariais	Não testado em ambientes científicos. Modelo adaptável.
(FERNANDEZ; TEDESCHI; PRIOL, 2016)	Execução descentralizada	Modelo de custos ausente
(BAHRAMI; SINGHAL, 2015)	Camada de integração *aaS	Modelo ainda incipiente
(LYASHOV <i>et al.</i> , 2016)	Criação de ambientes heterogêneos com SOA	Restrito a grade.
(BROWN; GROLINGER; CAPRETZ, 2011)	Integração de bases de dados	Faltam testes mais abrangentes
(WANG <i>et al.</i> , 2009)	Integração de bases de dados	Modelo geral. Adaptável a vários contextos
(MENON <i>et al.</i> , 2012)	Provisionamento de recursos	Focado em nuvem
(LU; KOGA, 2015)	Provisionamento de recursos	Aplicação fechada
(KRIEGER <i>et al.</i> , 2017)	Provisionamento de recursos	Focado em nuvem
(SANDES <i>et al.</i> , 2016b)	Provisionamento heterogêneo	Recursos de menor capacidade
(KAROCZKAI; KERTESZ; KACSUK, 2016)	Provisionamento heterogêneo	Faltam testes mais abrangentes
(COLLADO, 2017)	Provisionamento de recursos	Avaliação deficiente. Modelo restrito
(FIORE <i>et al.</i> , 2017)	Provisionamento de recursos	Focado em grade
(TOOSI; SINNOTT; BUYYA, 2017)	Provisionamento de recursos	Focado em nuvem
(PANADERO <i>et al.</i> , 2018)	Provisionamento de recursos	Recursos de menor capacidade
(GOHER <i>et al.</i> , 2018)	Provisionamento de recursos inter-nuvem	Focado em nuvem
MDAPSP	Padrão SOA para <i>gateway</i> científico e escalonamento para ambiente heterogêneo	Redução de <i>makespan</i> , integrável, provisão eficiente

Fonte: Dados da pesquisa.

A investigação da literatura mostra que o interesse em *e-science* tem crescido nos últimos anos, assim como os desafios e os diversos pontos de análise de soluções. As alternativas proporcionadas por SOA são diversas, assim como as necessidades que envolvem os experimentos de Bioinformática e a utilização de ferramentas dos *gateways* científicos. Percebe-se as diversas lacunas a serem preenchidas, dentre as quais tem urgência um modelo geral de estruturação desses *gateways* especialmente no provisionamento de recursos eficiente. A MDAPSP surge para atender essa necessidade, como uma arquitetura SOA com módulos integráveis, que considera um ambiente heterogêneo de recursos, possibilidade associação de módulos e fraco acoplamento. A Tabela 8 mostra um conjunto dos trabalhos relacionados, diferenciando-os de acordo com o cumprimento dos requisitos desejados para a arquitetura MDAPSP e mostrando os avanços que a arquitetura promove.

3.5 Considerações finais

Este Capítulo tratou dos trabalhos relacionados, sua organização no projeto e do estado da arte dos mesmos. A investigação da literatura mostra as necessidades que esse projeto busca suprir e como contribuir para o avanço das pesquisas. A revisão aponta para requisitos em

comum dos *gateways* científicos, como problemas no provisionamento de recursos, situação que é potencializada pela falta de opções desenvolvidas por especialistas em sistemas distribuídos.

Como citado anteriormente o Galaxy tem destaque em Bioinformática, pela sua flexibilidade e atuação em diversas pesquisas de ponta. Por esse motivo, ele foi o escolhido para foco do análise de desempenho dos *gateways* científicos. Idealmente, as soluções propostas pelo modelo de arquitetura pode ser generalizada para demais sistemas de *e-science*. O próximo capítulo descreve um estudo de caso usando o Galaxy, para medições de capacidade, estruturação em módulos e análise de gargalos.

ESTUDO EXPLORATÓRIO DO KOALA

4.1 Considerações iniciais

Este Capítulo apresenta a definição do Galaxy como foco de estudo de capacidade e destaca sua importância em *e-science*. Uma versão personalizada desse *gateway*, especializada em PSP é definida como estudo de caso, na qual são realizados testes diversos em ambiente controlado para a compreensão das necessidades recorrentes desses experimentos e dos modelos de provisionamento computacional possíveis. É descrito ainda o *Decision Maker*, um módulo de auxílio à escolha de ambiente computacional.

4.2 O Galaxy Koala

Em *e-science*, usuários com níveis diferentes de conhecimento buscam acesso a ferramentas computacionais avançadas. Nesse aspecto, destaca-se a importância do Galaxy e a sua aplicabilidade em diversas pesquisas de Bioinformática (GOECKS *et al.*, 2010). A possibilidade de instalar o *gateway* localmente é umas de suas principais vantagens, porque permite um controle maior das máquinas que o hospedam e contornar as limitações de largura de banda, por exemplo. Entretanto, máquinas disponíveis localmente em geral são menos potentes do que as de ambientes especializados, como supercomputadores ou servidores em nuvem (GESING *et al.*, 2017). De qualquer forma, a possibilidade de personalização e adição de mecanismos próprios faz do Galaxy um dos *gateways* mais flexíveis e capazes de oferecer um bom custo-benefício (AFGAN *et al.*, 2016). A Figura 15 mostra um exemplo de uma instância Galaxy, desenvolvida no Instituto de Ciências Matemáticas e de Computação (ICMC)¹. Denominada Koala², essa versão é especializada nas pesquisas de PSP, tendo produzido resultados para a área nos últimos anos (FACCIOLI; BORTOT; DELBEM, 2014). Pode-se modelar *workflows* na interface, adicio-

¹ www.icmc.usp.br/

² koala.lasdpc.icmc.usp.br

nar arquivos de proteínas, modificar os dados de entrada e os intermediários, encontrar preditores para PSP, realizar a execução e analisar resultados (DEFELICIBUS, 2016). Uma instância do Koala está disponível nos servidores do Laboratório de Sistemas Distribuídos e Programação Concorrente (LASDPC)³.

Figura 15 – Interface do Koala, implementação baseada no Galaxy

Fonte: Defelicibus (2016).

O Koala é um sistema *Web* para predição e análise de estruturas de proteínas, integração de algoritmos de PSP e análises de resultados. Assim como o *framework* base Galaxy, o Koala é de código aberto, conta com interface amigável e está preparado para ambientes distribuídos. O Koala Server é composto pelos seguintes componentes (DEFELICIBUS, 2016):

- **Servidor Web nginx:** servidor *proxy* em *Hypertext Transfer Protocol* (HTTP), o qual garante o acesso via *internet* às ferramentas;
- **PostgreSQL:** banco de dados relacional do Koala. Nele são armazenados detalhes do servidor, históricos de execução e informações dos usuários;
- **Pymol:** biblioteca que realiza a criação de imagens moleculares complexas, com base no arquivo PDB da proteína;
- **Highcharts:** uma API desenvolvida em JavaScript para geração de gráficos em tempo real;
- **JSmol:** softwares HTML5 para visualização de estruturas moleculares 3D;
- **Pulsar:** aplicação distribuída que cria serviços em máquinas-cliente para execução de tarefas em vários nós de computação.

³ lasdpc.icmc.usp.br

Para entendimento geral da estrutura, inicialmente é feito um mapeamento geral das ferramentas disponíveis. Para a fase inicial de experimentos, há recursos de inserção e coleta de dados, transformação desses arquivos em formatos específicos e preparação dos dados. A Figura 16 mostra as opções do usuário ao iniciar um procedimento de predição. O arquivo base das proteínas pode ser obtido do PDB ou informado via *upload*. Se necessário, é possível fazer modificações nos dados, tais como renomear átomos e unir diversos arquivos em um. A partir dessa fase o usuário toma decisões relativas ao tipo de processamento desejado para PSP.

Figura 16 – Diagrama do Koala - Primeiros passos de um experimento.

Fonte: Elaborada pelo autor.

No segundo passo, o usuário deve informar o tipo de preparação a ser realizada nos dados de entrada. Se o algoritmo de predição for do tipo evolutivo, devem ser criadas populações de indivíduos. Há duas ferramentas disponíveis para esse procedimento, como visto na Figura 17. Se o preditor não possuir essa formação como pré-requisito, o usuário pode seguir com os procedimentos.

Figura 17 – Diagrama do Koala - Preparação de dados.

Fonte: Elaborada pelo autor.

No passo seguinte, deve ser feita a escolha do preditor. O Koala possui atualmente onze algoritmos evolutivos dedicados aos testes de PSP, dentre os quais destaca-se o *Prot-Pred-Gromacs* (2PG), utilizado para os testes de capacidade do Koala. Variações desses recursos e

novas ferramentas podem ser integradas ao sistema, conforme forem desenvolvidas. A Figura 18 ilustra o processo de escolha do preditor.

Figura 18 – Diagrama do Koala - Predição.

Fonte: Elaborada pelo autor.

Ao iniciar o procedimento de PSP o usuário pode optar pelo uso do *GROningem MACHine for Chemical Simulations* (GROMACS) um *framework* de cálculo de minimização das forças envolvidas na estrutura da proteína. Esse procedimento influencia no *makespan* do *workflow* e também no resultado da predição. A Figura 19 mostra o fluxo decisório que conduz ao uso do *framework*.

Figura 19 – Diagrama do Koala - Minimização.

Fonte: Elaborada pelo autor.

Finalmente, atinge-se a última fase dos procedimentos. O Koala disponibiliza opções para o usuário tratar os dados e tomar decisões de mantê-los ou excluí-los. Também possibilita a disponibilização do *workflow* de execução, como visto na Figura 20. Há outras fases proporcionadas por ferramentas que não são consideradas nesse estudo de caso. Elas não interferem no resultado da predição.

Figura 20 – Diagrama do Koala - Disponibilização de dados.

Fonte: Elaborada pelo autor.

A Figura 21 descreve o processo de utilização da ferramenta em um diagrama de sequência. Esse *gateway* é utilizado para os testes de capacidade e as coletas de informações preliminares. Para um detalhamento completo, alguns experimentos de usuário foram realizados no ambiente.

Em versões anteriores, o Koala foi disponibilizado em instâncias de VM disponíveis em provedores de nuvem da Universidade de São Paulo (USP). Nesse caso, os usuários finais não têm acesso aos nós ou mesmo quaisquer informações sobre as máquinas que executam os procedimentos. Atrasos em experimentos e falta de capacidade foram recorrentes, problemas que foram agravados pela falta de suporte e posterior suspensão desse serviço em nuvem. A flexibilidade de instalação do Koala permite a utilização em máquinas variadas. Ele também permite a inclusão de novos módulos, tanto dos algoritmos dos preditores quanto para análises dos resultados. Os módulos disponíveis permitem integração e criação de *workflows*, além da importação de métodos prontos. Experimentos realizados no Koala ficam disponíveis publicamente para acesso, visualização e reprodução. Os históricos de execução, de entrada de dados e os *workflows* realizados são acessíveis por meio da mesma plataforma.

A experiência prévia de provisionamento em nuvem e a necessidade de uma coleta de dados mais precisa levou à opção por instalação do Koala nos *clusters* disponíveis no LASDPC. O laboratório conta três *clusters* denominados Cosmos, Andromeda e Halley. Desses foram selecionados o Andromeda e o Halley, para a instalação dos módulos do Koala, em uma distribuição para avaliar as capacidades de diferentes máquinas ao hospedar os serviços oferecidos pelo *framework*. As configurações e detalhes dos *clusters* são detalhadas nas Tabelas 9 e 10.

4.2.1 Estudo exploratório do Galaxy

Para um embasamento técnico e teórico que corrobore o desenvolvimento da arquitetura, é necessária uma avaliação do Galaxy que aponte suas capacidades e limitações, além

Figura 21 – Diagrama de sequência do planejamento de um experimento no Koala.

Fonte: Elaborada pelo autor.

de espelhar as necessidades apontadas na literatura. Usuários e administradores precisam de informações detalhadas dos testes e das máquinas, facilitando a reprodutibilidade, a visualização, a portabilidade e a flexibilidade. Nos experimentos desenvolvidos em (BRASIL; DELBEM; SILVA, 2013) por exemplo, descreve-se a execução manual e sequencial dos algoritmos. A automatização do processo de execução é uma evolução necessária, o que pode ser conseguido com modelos de *workflow*. Entretanto, sem recursos computacionais adequados a distribuição de tarefas fica comprometida, assim como nas informações de um procedimento em execução ou nas verificações do ambiente, o que compromete a *Quality of Experience* (QoE) e pode descumprir o SLA. Apesar da ampla utilização, o Galaxy apresenta muitas limitações.

A estrutura funcional do Koala passa pela organização dos módulos e distribuição e mensagens para coordenação das ações. Na instalação local, os nós foram divididos em três

Tabela 9 – Especificações dos nós do *Cluster Andromeda*

Cluster Andromeda	
Número de nós	13
Processador	8 núcleos AMD Vishera 4.2 Ghz
RAM	32 GB RAM DDR3 Corsair Vegeance
Hard Drives	HD 2TB Seagate SATA III 7200RPM
GPU	Video Nvidia GTX 650 – 1GB
Placa-mãe	Gigabyte 970A-D3
Fonte de Alimentação	ATX 650W Real
Sistema Operacional	Linux Ubuntu 14.04.4 LTS

Fonte: Dados da pesquisa.

Tabela 10 – Especificações dos nós do *Cluster Halley*

Cluster Halley	
Número de nós	13
Processador	Intel Core I7 Processor LGA -1150 – 4790 3.60GHZ DMI 5GT/S 8MB
RAM	32 GB RAM DDR3 Corsair Vegeance
Hard Drives	HD 2TB Seagate SATA III 7200RPM
GPU	Video Nvidia GTX 650 – 1GB
Motherboard	Gigabyte GA-Z97X-SLI ATX
Fonte de Alimentação	ATX 650W Real
Sistema Operacional	Linux Ubuntu 14.04.4 LTS

Fonte: Dados da pesquisa.

Tabela 11 – Capacidade das Máquinas do Koala

Nome	Nós	Núcleos	Memória RAM	HD
Koala Server	1	4	8 GB	50 GB
Koala Client	20	8	16 GB	50 GB
Koala Pred	4	8	16 GB	50 GB

Fonte: Elaborada pelo autor.

categorias gerais: *Koala Server*, *Koala Client* e *Koala Pred*. No *Koala Server* ficam hospedadas as configurações gerais do ambiente, a distribuição das tarefas e os esquemas de fluxo de dados. O acesso ao sistema é feito por qualquer terminal conectado, por meio da interface *Web*. No *Koala Client* ficam os módulos responsáveis pela criação de dados intermediários e por prover a informação para as fases posteriores. Os nós dessa categoria também são utilizados por procedimentos do GROMACS, que gera tarefas paralelas e distribui para os nós para cálculos de minimização de energia. Finalmente o *Koala Pred* abriga os diversos preditores, que calculam a estrutura 3D das proteínas. Tal organização pode ser vista na Figura 22. Na Tabela 11 estão descritas as capacidades das máquinas do Koala, conforme as categorias.

Figura 22 – Organização da estrutura do Koala em instalação local.

Fonte: Elaborada pelo autor.

No Koala está disponível o 2PG, um *framework* de computação evolutiva, capaz de realizar processamentos tanto mono quanto multiobjetivo, por meio de seus diversos preditores de PSP (FACCIOLI *et al.*, 2011), que também utiliza o GROMACS para o cálculo de *fitness* dos indivíduos (SPOEL *et al.*, 2013). O uso do 2PG na plataforma não exige formação técnica computacional (DEFELICIBUS, 2016). Esses algoritmos de predição foram utilizados como caso de teste para avaliação de capacidade do Koala.

4.2.1.1 Avaliação funcional

Em um primeiro experimento, foram escolhidas três proteínas de tamanhos diferentes (definido pelo número de resíduos), para serem utilizadas como carga no sistema. Todas são processadas pelo mesmo preditor do 2PG, o qual utiliza um algoritmo *Non-dominated Sorting Genetic Algorithm-II* (NSGA-II). Como esse algoritmo é multiobjetivo, foram definidos duas quantidades de objetivos de predição para cada cenário. Para os cenários com dois objetivos, os mesmos foram *Potential* e *GBSA Solvation*. Para os cenários com quatro objetivos são adicionados *Hydropobic* e *Hydrophilic* (DEFELICIBUS, 2016). O desenho de experimentos é exibido na Tabela 12. Essa bateria de testes visa a observação do comportamento do Koala e medições gerais no ambiente.

Tabela 12 – Parâmetros para o experimento inicial.

Algoritmo	Proteína	Resíduos	População	Gerações	Objetivos
NSGA-II	1VII	36	100/200	50/100	2/4
NSGA-II	1EOD	100	100/200	50/100	2/4
NSGA-II	1AGY	200	100/200	50/100	2/4

Fonte: Elaborada pelo autor.

O tempo de execução de um experimento pode variar de alguns minutos a várias semanas, conforme os parâmetros da execução. Para os preditores que contam com algoritmos genéticos, o tamanho da população e o número de gerações definidas afetam tanto o tempo resposta quanto a qualidade da predição (LUCENA *et al.*, 2013). O tamanho da proteína pode variar de dezenas de aminoácidos até a casa dos milhares, influenciando consideravelmente no tempo final. Caso uma tarefa seja abortada, o processamento tem que ser reiniciado. Por ser sequencial, com acesso simultâneo, é criada uma fila de execução. Mesmo o usuário administrador não tem acesso a informações sobre a existência de outros processos, tempo de fila ou do saturamento dos recursos. O sistema não verifica a validade dos parâmetros de entrada (como um arquivo FASTA inválido) e informa que a tarefa foi concluída com sucesso. Isso pode gerar dificuldades em testes com muitas variações e baterias.

Os experimentos mostram que, do ponto de vista de carga de trabalho, o Koala sofre com problemas de distribuição, sobrecarga e incapacidade de execução de procedimentos de maior escala, conforme diversos erros emergiram durante as medições. O crescimento do tempo de resposta é imprevisível, bastando algumas variações nos parâmetros de configuração para uma escalada no tempo e nível de recursos requeridos. Quanto ao volume de dados envolvidos, tanto dados intermediários quanto os de saída crescem vertiginosamente, sendo que em alguns casos o ambiente computacional foi incapaz de lidar com as informações. Falhas pontuais no servidor foram identificadas, como por exemplo na execução de proteínas maiores que 200 aminoácidos com população de 200 indivíduos e 100 gerações. Nessa configuração, o sistema não foi capaz de empacotar os dados, impedindo a conclusão do experimento. Foi preciso acessar o servidor

para verificação. Não é uma falha recorrente, contudo. Quando ocorre o preenchimento total do disco rígido, nenhuma notificação é disparada, sendo preciso acessar os servidores para mover ou apagar arquivos e então realizar novos experimentos. Apenas administradores são capazes de realizar tal verificação. A transferência do volume de dados em uma rede interna é um procedimento factível, entretanto pode ser problemática via *Internet*.

O sistema não fornece uma previsão de conclusão do experimento, nem informa o estado atual da execução, o que pode ser implementado como serviço. Uma vez submetida uma instância para execução por meio do *gateway* é preciso aguardar uma notificação de término. Em caso de falha em um *host*, o usuário pode ficar sem qualquer resposta. Outra limitação é a impossibilidade de acompanhamento dos procedimentos de *folding*. Contrariamente a serviços de SWMS (MATTOSO *et al.*, 2015), o Galaxy não permite *steering* nos *workflows*.

O Koala oferece a importação de proteínas do PDB, que entretanto tem algumas limitações e em alguns casos os dados de entrada são informados manualmente. Há alguns projetos (MANDAL; DAS; BHATTACHARJEE, 2012), (WANG; LIU, 2013) e (LU *et al.*, 2014) para resolver essa questão de recuperação de dados de proteína. Diversas partes desse *framework* são passíveis de paralelização, o que aumentaria o desempenho do sistema. Atualmente o *Galaxy Pulsar* faz a distribuição de tarefas para cálculo de minimização, pois essas têm baixa dependência. Na Tabela 13 são listadas as principais falhas identificadas no Galaxy Koala e qual a gravidade das mesmas. Esse estudo aponta para os diversos desafios na montagem e disponibilização do Koala, as quais se repetem em outros *gateways* científicos (ZIMMERMANN; GRUNZKE; KRÜGER, 2017), além de guiar as decisões de projeto para a construção da arquitetura. O problema de provisionamento computacional destaca-se como um dos primeiros a ser equacionado.

Tabela 13 – Principais problemas encontrados no Galaxy Koala

Componente	Falha	Domínio	Consequências	Detectável	Frequência
Preditor	Parada	Conteúdo	Médio	Sim	Ocasional
Preditor	Entrada	Conteúdo	Menor	Sim	Comum
Memória	Insuficiente	Tempo	Grave	Sim	Comum
HD	Insuficiente	Conteúdo	Fatal	Não	Raro
<i>Feedback</i>	Parada	Tempo	Menor	Não	Comum
Monitoramento	Inexistente	Conteúdo/Tempo	Grave	Não	Comum
Provedor	Parada	Conteúdo	Grave	Sim	Ocasional

Fonte: Elaborada pelo autor.

4.2.2 Avaliação da geração de dados intermediários

Em uma segunda análise, foram coletadas informações do comportamento dos dados intermediários de experimentos de PSP. Foi avaliado tempo de processamento para construir os arquivos de população utilizados pelos preditores. Essas execuções podem ter parâmetros muito variados, influenciando em outras tarefas da execução, afetando a capacidade do sistema e o

Tabela 14 – Fatores de avaliação do procedimento *Build Population*.

Proteína	Tamanho da Cadeia	Tamanho da População
2N0I	12	10-200
1VII	36	10-200
5K2L	49	10-200
1B1G	75	10-200
1EOD	100	10-200
1BR0	120	10-200
10Z9	150	10-200
1IM5	180	10-200
1AGY	200	10-200
2EEK	220	10-200
1PX4	250	10-200
2R3A	300	10-200

Fonte: Elaborada pelo autor.

makespan. Como parâmetros de entrada, o programa "*protpred-Gromacs-pop-initial*" recebe o arquivo FASTA (que contém a sequência de aminoácidos) e o tamanho da população, que definirá o número de modelos gerados, além de alguns parâmetros para o GROMACS e o campo de força utilizado. Como saída, a ferramenta gera um arquivo PDB (DEFELICIBUS, 2016). A construção da conformação fornece a informação necessária aos preditores, que calculam as forças de energia da proteína e produzem as estruturas em 3D.

Para avaliar o comportamento dessas tarefas, um conjunto de 12 proteínas diferentes foi selecionado, variando o tamanho da cadeia. Os resultados mostram o comportamento dos dados e o crescimento no tempo de processamento para cada proteína. Também é importante notar a diferença no tempo coletado de uma proteína para outra, e como ela pode variar de acordo com os dados de entrada. Na Tabela 14 são descritos o nome das proteínas, seu tamanho de cadeia e a variação no número de indivíduos, variando de 10 a 200. A configuração base do Koala é a mesma, disponibilizada na Tabela 11.

O objetivo é quantificar e buscar compreender o padrão de comportamento de tal processamento com vistas a auxiliar a tomada de decisão com relação à capacidade computacional requerida. Três técnicas de regressão são realizadas com os dados, para definir a relação entre o tamanho da cadeia de aminoácidos e o tempo para a criação das populações, para um dos preditores do Koala. O LibreOffice Calc⁴, oferece cálculos variados do conjuntos de dados, dos quais foram usados três modelos: Regressão Linear, Regressão Logarítmica e Regressão Exponencial (SUN; OUYANG; YUE, 2017). Os três modelos são comparados conforme suas variáveis de resposta como apresentado na Tabela 15. O melhor modelo será executado em todos os cenários, para a definição do padrão de crescimento.

Segundo essa padronização, observa-se que a regressão exponencial alcança os melhores

⁴ www.libreoffice.org

Tabela 15 – Variáveis de resposta do cálculo de regressão.

Modelo de Regressão	Linear	Logarítmica	Exponencial
R^2	Quanto mais próximo de 1 melhor		
Erro padrão	Quanto menor, melhor		
Slope	Informa a direção da linha de tendência		
Intercept	Intersecção com a linha de valores Y		

Fonte: Dados da pesquisa.

Tabela 16 – Amostra das variáveis de saída das regressões lineares.

Modelo de Regressão	Linear	Logarítmica	Exponencial
R^2	0,9561782269	0,7269354637	0,933477307
Erro padrão	5,5869713627	13,9464643979	0,1520720274
Slope	0,3365048177	26,1406581458	0,6544152696
Intercept	7,4664651426	-70,3608146712	2,1453665018

Fonte: Elaborada pelo autor.

resultados. A Tabela 16 apresenta uma amostra dos resultados obtidos para a proteína 1VII. O comportamento da regressão é semelhante para as demais proteínas testadas, indicando uma não-linearidade na correlação dos dados.

A primeira proteína é a 2N0L, composta por 12 aminoácidos (ou resíduos). Os resultados mostram a variação do tempo de processamento inicial de apenas 10 segundos para cerca de 80 segundos nas últimas variações, exibidos na Figura 23a. Para a proteína 1VII (36 aminoácidos), os resultados mostram o crescimento de 180 segundos para quase 1000 segundos como se vê na Figura 23b. A proteína 5K2L (49 aminoácidos) apresenta resultados acima de 1200 segundos para processar as populações maiores, conforme a Figura 23c. A proteína 1B1G (75 aminoácidos) requer 3600 segundos (ou 6 minutos) para populações de 200 indivíduos, como visto na Figura 23d. A linha de regressão para este casos mostra comportamento próximo ao dos dados coletados no Koala.

A regressão da proteína 1EOD (100 aminoácidos), foi bastante precisa especialmente para as populações menores, conforme mostra a Figura 24a. Esta proteína leva mais de 10 minutos para produzir maiores populações. A proteína 1BR0 (120 aminoácidos) tem um tempo de execução próximo ao da anterior como se vê na Figura 24b. A regressão feita com a proteína 1OZ9 (150 aminoácidos) mostra uma boa taxa de acerto, como visto na Figura 24c. A proteína 1IM5 (180 aminoácidos), tem comportamento similar, exibida na Figura 24d. A proteínas a partir dessa faixa de tamanho começam a gastar várias horas apenas para a criação dos arquivos de população com maior número de indivíduos.

A última amostra traz as quatro maiores proteínas selecionadas nessa avaliação. Para a proteína 1AGY (200 aminoácidos), a regressão apresenta um comportamento bem próximo aos dados coletados, como visto na Figura 25a. No caso da proteína 2EEF (220 aminoácidos),

Figura 23 – Gráficos com o resultado da regressão exponencial. Primeira amostra.

Fonte: Elaborada pelo autor.

Nota – A linha azul representa os dados coletados e a vermelha o comportamento da regressão. O tamanho da população varia de 10 a 200 indivíduos.

a linha de regressão apresenta leve desvio, porém respeita o comportamento de crescimento, como observado na Figura 25b. O mesmo pode ser dito do resultado da proteína 1PX4 (250 aminoácidos), apresentado na Figura 25c. A proteína 2R3A (300 aminoácidos) foi a de maior cadeia selecionada. A regressão apresenta um padrão regular, como visto na Figura 25d. Para as proteínas desse último cenário, o tempo de geração da população pode ser de até 8 horas de processamento. Considerando-se que este é um passo intermediário de PSP, percebe-se a necessidade de planejamento de capacidade adequado. Ressalta-se a possibilidade de escolher parâmetros de tamanho de proteína e de populações maiores do que os aqui relacionados, exigindo ainda mais do ambiente computacional.

Os dados intermediários são gerados no processo prévio à predição. Pode-se considerar armazenar e reutilizar as informações geradas, avaliando os custos de movimentação ou criá-los novamente quando necessário, avaliando o custo benefício entre conjunto de dados e capacidades da máquina (YUAN *et al.*, 2013). É possível definir formas de reutilização dos dados intermediários, conforme suas características.

Estes resultados são importantes de dois pontos de vista. Primeiro é possível compará-los diretamente e ter uma ideia de como a variação nas entradas afeta o tempo necessário para a

Figura 24 – Gráficos com o resultado da regressão exponencial. Segunda amostra.

Fonte: Elaborada pelo autor.

Nota – A linha azul representa os dados coletados e a vermelha o comportamento da regressão. O tamanho da população varia de 10 a 200 indivíduos.

construção das populações. Em segundo lugar, as informações de regressão permitem extrapolar os dados coletados e prever o comportamento da carga de trabalho. Neste momento, tal estudo serve como uma avaliação da sobrecarga imposta pelos dados intermediários e avaliar a abordagem de regressão nos dados. Isso é útil para identificar os requisitos dos experimentos e ajudar a decidir se é útil reaproveitar dados, transferir ou produzi-los novamente. Esse tipo de heurística deve aparecer em versões mais avançadas da MDAPSP. Ao se considerar sistemas distribuídos para o provisionamento de dados e antevendo o desacoplamento dos módulos dos *gateways* científicos, avaliações mais avançadas devem ser realizadas para identificar corretamente qual a melhor abordagem em cada caso.

4.2.3 Mapeamento da carga de trabalho

Para o detalhamento da carga de trabalho imposta por um dos preditores do Koala, foi realizada uma terceira avaliação. Nesse caso, o objetivo é identificar o peso das funções internas em um processo de predição, de forma a identificar pontos de sobrecarga, gargalos e dependências. Para isso foi utilizada a ferramenta *Gnu Profiler* (Gprof)⁵, um mecanismo capaz

⁵ http://www.network-theory.co.uk/docs/gccintro/gccintro_80.html

Figura 25 – Gráficos com o resultado da regressão exponencial. Terceira amostra.

Fonte: Elaborada pelo autor.

Nota – A linha azul representa os dados coletados e a vermelha o comportamento da regressão. O tamanho da população varia de 10 a 200 indivíduos.

de medir o desempenho de um programa e detalhar suas funções internas. São registrados o número de chamadas para cada função e a quantidade de tempo dedicado a cada uma delas. Aquelas que consomem as maiores frações de tempo são identificadas na saída do programa. Isso permite direcionar os pontos de melhoria em um programa, focando nas funções que dominam o tempo de execução. Para esse experimento foi utilizado o preditor 2PG-mono, disponível no Koala. Esse preditor foi o selecionado por ser um dos principais utilizados nos experimentos de PSP e por ser um modelo base para os demais preditores. Como o nome indica, esse preditor só permite o cálculo de um objetivo de predição. Um exemplo da saída desse programa de teste é mostrado na Tabela 17.

Esse experimento é importante para a modelagem de comportamento de um dos preditores do Koala e a partir dele, mapear as funções e sua hierarquia. Com essas informações é modelada de forma sintética essa carga, em padrão de arquivos DAX, configurando um *workflow* científico do processo de predição. A Figura 26 mostra o primeiro dos *workflows* gerados, nesse caso para a proteína 1VII, executada com o preditor 2PG-Mono. Foram realizadas medições com diversas proteínas e diferentes parâmetros de teste, as quais serão disponibilizadas e utilizadas

Tabela 17 – Exemplo de saída do Gprof com as funções de um preditor.

Tempo (%)	Tempo Total	Tempo por chamada	Nome da função
53.10	0.60	0.60	search_pdb_atom_from_resnum_atom_name
24.78	0.88	0.28	rotation_by_angle_dih
7.96	0.97	0.09	writeATOM
2.65	1.00	0.03	run_program
2.65	1.03	0.03	copy_pdb_atom
1.77	1.05	0.02	rotate_all_atoms
1.77	1.07	0.02	get_res_name_from_res_num
0.88	1.08	0.01	load_pdb_atoms_split
0.88	1.09	0.01	rtrim
0.88	1.10	0.01	is_equal
0.88	1.11	0.01	write_top_residue_atom_info
0.88	1.12	0.01	write_top_residue_atom_info_side_chain
0.88	1.13	0.01	write_top_residue_atom_info_withoutmoved

Fonte: Elaborada pelo autor.

para avaliações futuras.

Um segundo *workflow* de proteína foi definido por esse mesmo procedimento, para a cadeia 1AGY que é composta de 200 aminoácidos, como visto na Figura 27. Além do esperado aumento do tempo de processamento e maior sobre carga imposta ao sistema, as variações de tamanho das proteínas envolvem mudanças nas chamadas das funções durante o procedimento de PSP. Tais modelagens são fundamentais para a identificação da variabilidade de carga entre os muitos cálculos envolvidos em um experimento e ainda mais para o modelo de simulação no qual os *workflows* são informados como parâmetros de carga de trabalho.

O terceiro *workflow* gerado corresponde à proteína 1EOD, com 100 aminoácidos em sua cadeia, como visto na Figura 28. Novamente, o número de funções mais invocadas varia e nesse caso foram encontrados alguns pontos de maior potencial para distribuição de tarefas.

Os círculos são representações das funções mais invocadas pelos procedimentos do algoritmo 2PG que indicam pontos críticos do programa, os quais podem ser alvo de melhorias e podem indicar possibilidade de paralelização. É importante notar que atualmente os procedimentos do 2PG são exclusivamente sequenciais, entretanto a modelagem dos procedimentos indicam os pontos em que as funções podem ser paralelizadas. Além disso, o cálculo de minimização de energias, realizado pelo GROMACS não é considerado nessa modelagem atualmente. Esse cálculo já existe de forma paralelizada, realizado pela ferramenta Pulsar (DEFELICIBUS, 2016). Essa ferramenta distribui tarefas entre os nós conectados à infraestrutura do Koala para aumentar o desempenho do processamento. Nesse caso, o uso de ambientes distribuídos e a indicação de quais nós estejam disponíveis para esse procedimento pode melhorar a qualidade e o desempenho dos experimentos. Inclusive internamente às funções são encontrados pontos em que a dependência de dados e o número de sub-tarefas permite técnicas de processamento distribuído capazes de promover *speed up*. Essa investigação não é realizada nesse trabalho, nem

Figura 26 – *Workflow* do 2PG-Mono, para a proteína 1VII.

Fonte: Elaborada pelo autor.

em ambiente real, nem no simulado. Ambas as abordagens podem ser úteis para essa avaliação, apresentando resultados precisos e inovadores.

Outra condição a se considerar é a capacidade de execuções completas de PSP em diversas máquinas, ampliando a capacidade de atendimento do Koala. Dessa forma, mesmo que não haja paralelismo de tarefas, a maior disponibilidade de máquinas com provedores de serviço capazes de atender às múltiplas solicitações já melhora o nível de serviço considerando o acesso de múltiplos usuários e reduzindo potenciais filas de atendimento. Esse é um dos pontos fundamentais de motivação para a arquitetura MDAPSP, considerando a distribuição dos módulos dos *gateways* científicos em máquinas heterogêneas para melhor capacidade de atuação.

4.2.4 Monitoramento da Arquitetura

O monitoramento é um conceito bem suportado nos SWMS e execuções de *workflow* científicos em ambientes de alto desempenho requerem esse apoio. O monitoramento possibilita

Figura 27 – *Workflow* do 2PG-Mono, para a proteína 1AGY.

Fonte: Elaborada pelo autor.

informações atualizadas do estado de um experimento, verificar a qualidade dos dados ou a ocorrência de falhas. Um *gateway* deve fornecer informações que permitam redirecionar os *workflows* ou reexecutar partes do mesmo (MATTOSO *et al.*, 2015). Apesar dos modelos de monitoramento disponíveis (BUTNARU; BUSE; PFLÜGER, 2012) (KAIL; KACSUK; KOZLOVSZKY, 2015) (SIMONET *et al.*, 2015) para *workflows*, o aspecto de supervisionar os componentes físicos (*hosts*) de um *gateway* científico não está bem explorado (KOZLOVSZKY *et al.*, 2014).

O objetivo é monitorar os componentes incluindo as estruturas de provisionamento, para

Figura 28 – *Workflow* do 2PG-Mono, para a proteína 1EOD.

Fonte: Elaborada pelo autor.

evitar seleção de algum elemento com falhas, longas esperas e degradação de qualidade. As métricas a serem exploradas em ambientes de larga escala incluem coleta de desempenho das estruturas de HPC (POUCHARD *et al.*, 2017). O tamanho, a complexidade e o dinamismo das estruturas de *e-science* (PERERA; MARRU; GANNON, 2008) e a geração de grandes volumes de dados (SIMONET *et al.*, 2015). dificultam aspectos de gerenciamento e manutenção.

Por essas razões, projeta-se que a MDAPSP empregará serviços de monitoração dos nós que compoñham a infraestrutura. Usado em outros trabalhos com bons resultados e baixo *overhead* (ESTRELLA, 2010) (OLIVEIRA *et al.*, 2014), o monitor Ganglia é uma ferramenta útil para coletar informações do sistema sobre provedores de serviço, armazenamento e escalonadores. O monitor é maleável e oferece uma coleção de instrumentos, permitindo uma

Tabela 18 – Desenho de experimentos de teste do Ganglia Monitor

Proteína	Tamanho	População	Geração	Objetivos	Minimização
1VII	36	20 ou 40	50, 80 ou 100	2 ou 4	Falso ou Verdadeiro
1BLQ	90	20 ou 40	50, 80 ou 100	2 ou 4	Falso ou Verdadeiro
1EOD	100	20 ou 40	50, 80 ou 100	2 ou 4	Falso ou Verdadeiro
1OZ9	150	20 ou 40	50, 80 ou 100	2 ou 4	Falso ou Verdadeiro

Fonte: Elaborada pelo autor.

configuração detalhada de índices, nos quais os parâmetros de QoS são selecionados conforme as necessidades. Esse monitoramento de componentes difere do monitoramento do *workflow*, destinado à acompanhamento dos experimentos em execução. Ganglia é um sistema de monitoramento para sistemas de alto desempenho, com características de escalabilidade e capacidade de distribuição. Funciona em hierarquia organizacional, composto por um nó mestre e diversos escravos, com escuta e anúncio de estado baseado em *multicast*. A aplicação é responsável por coletar índices de desempenho em intervalos predefinidos e pode ser adaptado para evitar sobrecarga, coletando informações como uso de memória, tráfego de rede e ocupação de CPU (BHATNAGAR; PATE, 2013).

4.2.4.1 Estudo de Monitoramento

O Ganglia foi instalado nos provedores que hospedam os preditores do Koala, para um mapeamento funcional. O experimento nesta seção foi projetado para mostrar interação de cargas de trabalho com o sistema, de forma a detectar mudanças nos índices dos componentes coletados pelo Ganglia. Neste caso, foi usado um preditor com um algoritmo NSGA-II que oferece opções multiobjetivo para a predição da proteína. O desenho de experimentos é descrito na Tabela 18. Foram escolhidas quatro proteínas, de acordo com os seus tamanhos de cadeia. O tamanho da população é de 20 ou 40 indivíduos e as gerações são de tamanho 50, 80 ou 100. Os objetivos de predição foram definidos em quantidades de 2 ou 4, sendo no primeiro caso *GBSA Solvation* e *Van der Waals*, e para a segunda instância são adicionados *Hydrophobic* e *Hydrophilic*. Finalmente, foi definido o processo de minimização de energias como Verdadeiro ou Falso, uma tarefa demorada que acrescenta complexidade significativa aos cálculos e aumenta o tráfego de mensagens (FACCIOLI, 2012). Nesse caso, todos os experimentos foram executados no ambiente do *cluster* Andromeda, descrito na Tabela 9. As quatro proteínas foram testadas em todas as combinações de cenário, para coletar a variabilidade de carga que os experimentos de PSP podem impor ao ambiente. Os tempos de resposta variam de poucos minutos a até mais de 3 horas de execução.

Durante as execuções, as informação coletadas com Ganglia demonstram os momentos de maior carga de trabalho e os componentes computacionais mais sobrecarregados. Na Figura 29a a carga agregada nos provedores é demonstrada. PSP é uma tarefa exigente, exigindo alta capacidade de processamento e memória. Os picos são representados pelos momentos em que as

proteínas de maior cadeia foram processadas. Na Figura 29b, outra visão das máquinas demonstra que a carga de trabalho sobrecarregou a capacidade instalada. Isso aponta para a necessidade de melhor distribuição de tarefas e balanceamento de carga para atender a demanda. A Figura 29c mostra o uso da memória durante os experimentos. Em alguns pontos o limite disponível do sistema (32 GB de RAM) esteve próximo de ser alcançado. A Figura 29d mostra o tráfego na rede interna do laboratório. A troca de informações foi elevada durante os experimentos e os picos nos parâmetros de saída são explicados pela distribuição do Koala e as comunicações entre os nós.

Figura 29 – Gráficos do monitoramento do Koala com uso do Ganglia

Fonte: Elaborada pelo autor.

Essa é uma avaliação geral da distribuição atualmente configurada do Koala em laboratório. Os resultados indicam que proteínas de tamanho recorrentemente usados em experimentos podem sobrecarregar ambientes computacionais com boa capacidade instalada. Testes mais abrangentes devem ser realizados no Koala e outros *gateways* para a definição de padrões de uso e boas práticas de monitoramento.

4.3 Decision Maker

O provisionamento de recursos é um problema sensível, seja estático ou sob demanda (SENTURK *et al.*, 2016). Há uma tendência atual de enviar os recursos para provedores de nuvem, que atrai pela escalabilidade e o gerenciamento por conta da empresa provedora. As

soluções *pay-as-you-go* oferecem diversos níveis de serviço, considerando várias capacidades de investimento (MADDURI *et al.*, 2015).

Contudo, é necessário considerar outras opções além da nuvem, tendo em conta a largura de banda, os preços envolvidos, segurança dos dados e liberdade de configuração (COCK *et al.*, 2013). Uma opção são os *clusters* de computadores, que são comumente disponíveis em universidades e institutos de pesquisa. Na MDAPSP considera-se a provisão de recursos diversificados para *gateways* científicos e a análise de soluções otimizadas, usando o *Galaxy* como estudo de caso.

A análise de vários *gateways* científicos disponíveis atualmente, permite identificar alguns dos módulos gerais dessas ferramentas, e identificar os componentes internos, como visto na Figura 30. Em geral, os *gateways* são acessíveis por meio de uma interface, geralmente acessível em uma página *Web*, onde ações como autenticação e níveis de controle são realizadas. A interação do usuário com o sistema também é realizada por esse módulo, sendo fundamentais a facilidade de uso e o suporte. O módulo seguinte abriga as ferramentas de SWMS, onde são definidas sequências de tarefas a serem executadas, além de importar e exportar essas informações, ponto em que a reprodutibilidade tem destaque. Na sequência da estrutura aparecem ferramentas de gerenciamento como escalonadores, monitores e balanceamento de carga. Finalmente aparece o módulo de execução, responsável por encaminhar as requisições para as máquinas e repositórios que tratam da execução, armazenamento e recuperação de informações dos procedimentos.

Nesta distribuição é proposto um novo módulo, não encontrado nos *gateways* avaliados. O *Decision Maker*, é um componente responsável por fazer uma análise prévia das necessidades do usuário, calculando o melhor custo-benefício de configuração computacional para a execução de um experimento. Sua função é identificar o melhor tipo ou conjuntos de máquinas para realizar os procedimentos, de forma a promover economia de recursos computacionais, reduzir tráfego de rede desnecessário e evitar despesas com instâncias de computação em nuvem. A Figura 31 apresenta um conceito abstrato de três ambientes de computação, incluindo máquinas *desktop*, *cluster* e máquinas em *nuvem* que compõe um ambiente heterogêneo de capacidade crescente, no qual também é considerado alternativamente um banco de dados integrado, acessados por todas as plataformas de computação.

O mecanismo do *Decision Maker* envolve o uso de aprendizado de máquina para analisar as execuções passadas e descobrir padrões de comportamento do sistema, para então definir a melhor configuração conforme a entrada de dados. Dependendo dos recursos necessários e dos tamanhos dos dados envolvidos, a decisão pode ser tomada entre fazer tudo localmente, em máquinas *desktop*, fazer uso de máquinas instaladas em *cluster* ou acessar instâncias de nuvem.

Figura 30 – Organograma dos módulos gerais de *gateways* científicos.

Fonte: Elaborada pelo autor.

Nota – O módulo *Decision Maker* é uma lacuna nessas ferramentas e se propõe a otimizar o uso dos recursos computacionais. Considerando-se um modelo heterogêneo de provisionamento, o módulo oferece informações para auxílio à escolha do tipo de máquina, baseadas no histórico de execuções.

4.3.1 Algoritmo do Decision Maker

Esta seção apresenta os algoritmos de aprendizado de máquina usados para aprender com os dados obtidos em experimentos de PSP executados previamente na plataforma Koala. Inicialmente apresenta-se as notações. Cada ponto de dados é referido como o vetor $(\mathbf{x}, y)$, onde $\mathbf{x} = (x_1, x_2, \dots, x_d)$ é um vetor d -dimensional de *variáveis de entrada* (ou características), e y é uma única variável de saída, por exemplo, a *variável alvo*. O objetivo de um algoritmo de aprendizagem é produzir um *preditor* $\phi()$, que é treinado usando o conjunto de aprendizagem $\mathcal{L} = (\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)$. Se o preditor $\phi()$ tem bons resultados ele realmente aprendeu com dados, de forma que poderá prever dados não vistos com erro baixo, desde que estes dados tenham sido amostrados na mesma distribuição de dados $\mathcal{L}$.

Como a variável alvo y_i dos experimentos é um valor numérico (o tempo de processamento), foram escolhidos algoritmos referidos como *modelagem de previsão*. Para validar o

Figura 31 – Ilustração do ambiente heterogêneo de recursos da infraestrutura.

Fonte: Elaborada pelo autor.

Decision Maker são avaliados três algoritmos diferentes de regressão linear e um baseado em SVM. Esses algoritmos foram selecionados para esta tarefa devido aos dados serem contínuos e porque diferentes algoritmos de aprendizagem de máquinas aprendem de maneira diferente com os dados. Isso significa que há variações na qualidade da aprendizagem conforme o conjunto de dados.

O método de *regressão linear* aproxima uma fórmula como a mostrada na Equação 4.1, na qual $W = w_0, w_1, w_2, \dots, w_d$ é o vetor de coeficientes. Assim, o preditor obtido é um ponto produzido entre as variáveis de entrada $X = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$ e o coeficiente W . Esses coeficientes são estimados pela minimização do erro quadrático, mostrado na Equação 4.2.

$$y_i \approx w_0 + w_1 \cdot x_{i,1} + w_2 \cdot x_{i,2} + \dots + w_d \cdot x_{i,d}, \text{ em que } i \in 1, 2, \dots, n \quad (4.1)$$

$$O = \sum_{i=1}^n (W \cdot X_i - y_i)^2, \text{ em que } i \in 1, 2, \dots, n \quad (4.2)$$

Coefficientes menores W são desejáveis, porque eles reduzem *overfitting* (AGGARWAL, 2015). Para fazer isso, duas abordagens para regressão linear são mais efetivas: *Ridge Regression* e *Lasso*. No primeiro, uma penalidade L_2 é somada na função objetivo O da Equação 4.2. Essa penalidade, $\lambda ||W||^2$, onde $\lambda > 0$, penaliza as regressões lineares com coeficientes maiores. Por outro lado, o algoritmo *Lasso* usa a penalidade L_1 somada para O , $\lambda \sum_{i=1}^d |w_i|$, obtendo soluções esparsas, ou seja, com poucos componentes diferentes de zero. Essa característica é especialmente eficaz ao lidar com características irrelevantes.

O algoritmo *Epsilon-Support Vector Regression* (SVR) foi inicialmente desenvolvido

para a classificação binária de dados numéricos. Concentra-se na otimização do hiper-plano ao maximizar a margem entre duas classes consideradas. Essa abordagem demonstrou boa generalização para dados não vistos, pois é fundamentada no *framework* da teoria da aprendizagem estatística (AGGARWAL, 2015).

Um SVR pode ser usado como regressão usando o método ε - SV regression (SMOLA; SCHÖLKOPF, 2004). Seu objetivo é encontrar uma função linear $f(x) = W \cdot X_i + b$, com $w \in \mathbb{R}$ e $b \in \mathbb{R}$, que tenha no máximo ε desvio dos alvos realmente obtidos y_i para todos os dados de treinamento. Ou seja, se o erro de previsão for inferior a ε , ele não é considerado. Ao mesmo tempo, busca w , uma propriedade chamada *flatness*. A vantagem de usar SVM é que eles podem ser acoplados a *kernels*, transformando os dados de entrada em outro espaço de maior dimensão. Essa transformação permite separar dados não-lineares.

4.3.2 Avaliação dos algoritmos de regressão

O objetivo dos experimentos nessa Seção é obter um modelo que possa prever o tempo de processamento dos experimentos de um preditor de PSP com base em histórico de testes, executados em três ambientes diferentes (computador local, *cluster* e nuvem). O objetivo é estimar qual a melhor infraestrutura de máquinas para execução de um experimento, com base nos parâmetros de entrada.

Primeiramente foi realizada uma longa bateria de experimentos, com um conjunto diversificado de parâmetros, analisando o tempo de processamento em cada um deles. Para esses experimentos é utilizado o preditor 2PG-Mono (FACCIOLI, 2012), que conta com um algoritmo genético usado para determinar a forma de 3D da proteína, tendo como parâmetros de entrada o tamanho da cadeia de aminoácidos, o tamanho da população e o número de gerações. A Tabela 19 apresenta os respectivos valores.

As proteínas para essa avaliação foram selecionadas com base no *Critical Assessment of protein Structure Prediction (CASP)*⁶, um dos eventos mais importantes focados em PSP, e as proteínas utilizadas pelas equipes na competição estão entre os principais objetivos do campo. Um histograma dos tamanhos de proteína (em relação ao número de aminoácidos) é apresentado na Figura 32.

Para compor um ambiente computacional de máquinas variadas, foi definida uma configuração de máquinas de trabalho em três classes diferentes, o tipo “desktop”, o tipo “cluster” e o tipo “cloud”. As diferentes configurações para essas máquinas são apresentadas na tabela 20. Todas essas máquinas são hospedadas no *cluster* Andromeda do LASDPC. As denominações das mesmas e suas configurações buscam emular as configurações de máquina condizentes com cada um dos tipos e também para formar uma variação de pontos de análise. Nesse experimento é considerado o tempo de resposta como variável a ser analisada.

⁶ <http://predictioncenter.org/>

Tabela 19 – Parâmetros para os experimentos preliminares.

Proteína	Tamanho da Cadeia	População	Gerações
2N0L	12	De 20 até 30	De 10 até 150
1BCV	20	De 20 até 30	De 10 até 150
1AI0	21	De 20 até 500	De 20 até 1000
2ETI	28	De 20 até 200	De 20 até 400
1C94	38	De 20 até 200	De 20 até 1000
5K2L	49	De 20 até 200	De 40 até 800
5HVZ	50	De 20 até 100	De 10 até 400
1B1G	75	De 20 até 50	De 40 até 400
1EOD	100	De 100 até 300	De 200 até 1200
1CM7	100	De 30 até 100	De 30 até 100
1OZ9	150	De 20 até 50	De 30 até 200
4IEU	200	De 50	De 2 até 30
1AGY	200	De 20 até 50	De 30 até 300
2EEK	220	De 20 até 30	De 20 até 200
2R3A	300	De 20	De 60 até 200
2BX6	350	De 20 até 50	De 40 até 150

Fonte: Elaborada pelo autor.

Nota – Experimentos usados na criação da base de dados para o aprendizado de máquina.

Figura 32 – Levantamento dos tamanhos das cadeias processadas no último CASP, em 2016.

Fonte: Elaborada pelo autor.

Foi coletado o tempo de processamento de 753 experimentos. Para os modelos de regressão, são variáveis de entrada: o tamanho da cadeia proteica, o tamanho da população, o número de gerações e qual o tipo de máquina utilizado (nuvem, *cluster* ou local). A variável de

Tabela 20 – Capacidades de máquina de cada classe

Ambiente	HD	CPU	RAM	SO
Desktop	20 GB	2 Núcleos	8 GB	Linux Ubuntu 14.04.4 LTS
Cluster	50 GB	4 Núcleos	16 GB	
Cloud	200 GB	8 Núcleos	32 GB	

Tabela 21 – Amostra dos resultados obtidos.

Cadeia	População	Gerações	Tipo de Máquina	Tempo de processamento (s)
12	100	300	Cluster	41.563
36	100	400	Desktop	173.7608
38	200	350	Cluster	276.0833
100	100	300	Desktop	677.2027
100	200	400	Nuvem	979.0801

Fonte: Elaborada pelo autor.

saída é o tempo de processamento. A Tabela 21 traz uma amostra de tais dados, em que se pode notar as variáveis de entrada X , tamanho de cadeia, população, gerações são números discretos e o tipo de máquina é uma *string*. Para a saída Y , o tempo de processamento é uma variável contínua. Como o objetivo do modelo é prever o tempo de processamento, foram treinados modelos de regressão com esses dados.

Os algoritmos avaliados são quatro no total, sendo três de regressão linear: a regressão linear típica (mínimos quadrados comuns), a regressão Lasso (com penalidade L1) e a regressão Ridge (com penalidade L2). O quarto é uma regressão não-linear, do tipo SVR, explicados na Seção 4.3.1. São algoritmos comuns na área de aprendizado de máquina, que fornecem fortes fundamentos teóricos e são comprovadamente capazes de aprender (ABU-MOSTAFA; MAGDON-ISMAIL; LIN, 2012).

Antes do treinamento de tais algoritmos, os dados de entrada são divididos em conjuntos de treinamento e teste, uma técnica chamada *holdout* (ABU-MOSTAFA; MAGDON-ISMAIL; LIN, 2012), para realmente avaliar os resultados de previsão oferecidos pelos algoritmos. São treinados os algoritmos com 75% dos dados (o conjunto de treinamento) e os avaliados com o restante de 25% de dados não vistos (o conjunto de teste). Para medir os resultados do modelo, são usadas duas métricas: R^2 score e *Root Mean Squared Error* (RMSE). Quanto melhor a previsão, o R^2 é mais próximo de 1 e menor é o RMSE. Também foram realizadas algumas transformações nos dados. Antes do treinamento, para aproveitar ao máximo os algoritmos de aprendizagem, são criados três conjuntos de dados adicionais além dos dados brutos: uma versão padronizada (que respeita uma média de 0 e variância unitária), um *log* transformado e um *log* de dados padronizados. No final, todos os algoritmos são avaliados com os quatro conjuntos de dados diferentes, verificando as duas métricas. A biblioteca Python Scikit-learn (PEDREGOSA et al., 2011) fornece todo o ambiente para particionar dados, treinar algoritmos de aprendizado e

Tabela 22 – Resultados da regressão.

Regressão	Dados Brutos	Padronizado	Log	Log Padronizado
Linear	0.478 / 932.721	0.478 / 932.721	0.534 / 881.054	0.534 / 881.054
Lasso	0.480 / 930.741	0.492 / 920.413	0.534 / 880.912	0.535 / 880.317
Ridge	0.479 / 931.531	0.487 / 924.743	0.540 / 875.631	0.540 / 875.442
SVR	0.981 / 174.638	0.997 / 70.505	0.941 / 312.445	0.948 / 293.050

Fonte: Elaborada pelo autor.

Nota – Considerados quatro algoritmos: Regressão linear, Lasso, Ridge e SVR, com quatro conjuntos de dados: bruto, padronizado, log e log padronizado. Os resultados são mostrados usando duas métricas: R^2 / RMSE. Quanto mais perto R^2 de 1 e quanto menor o RMSE, melhor a predição.

avaliá-los e a ferramenta SciPy ([JONES et al., 2001](#)) usada para transformar os dados e exibir os resultados. Os resultados dessa avaliação são mostrados na Tabela 22.

Como se pode notar, os modelos de regressão linear proporcionaram resultados insatisfatórios e se comportaram de forma muito similar, mesmo para diferentes transformações nos dados. Isso indica que os dados seguem alguma não-linearidade, já que o SVR obteve resultados muito melhores. Nos dados padronizados, o SVR obteve o melhor R^2 , de 0.997 (lembrando, quanto mais perto de 1, melhor) e o RMSE de 70.505 (o mais próximo de 0 melhor). Na Figura 33, são exibidos os resultados obtidos com as regressões, ambos considerando no eixo X o valor previsto para uma linha de dados não vistos e o eixo Y como o valor esperado (isto é, o valor real). Se a previsão segue os valores esperados (o que deve ocorrer se definido um bom modelo), deve-se obter um gráfico no qual o valor previsto seja igual ao valor esperado, formando uma linha diagonal. Na Figura 33a, o gráfico mostra as previsões obtidas a partir do modelo linear onde se pode notar que os valores não respeitam a linha diagonal. O mesmo vale para as Figuras 33b e 33c para as regressões Lasso e Ridge, respectivamente. Obteve-se comportamento semelhante para todas as transformações de dados. Finalmente, a Figura 33d exibe o resultado obtido com o melhor modelo (SVR em dados padronizados), o qual segue quase perfeitamente a linha diagonal.

Tendo esse modelo em mãos, pode-se estimar o melhor ambiente para o processamento de PSP, dentro dos parâmetros considerados. Planejando um novo experimento com as seguintes variáveis: o tamanho da cadeia de proteína de 72, o tamanho da população de 80 e o número de gerações de 200. Pode-se prever o tempo de processamento desse experimento em cada ambiente enviando como entrada para o modelo três linhas diferentes, em que os mesmos parâmetros são informados para os três tipos de máquina: $X_1 = \langle 72, 80, 200, \text{"local"} \rangle$, $X_2 = \langle 72, 80, 200, \text{"cluster"} \rangle$, e $X_3 = \langle 72, 80, 200, \text{"cloud"} \rangle$. A previsão do modelo para essas variáveis de entrada é $y_1 = 1673.344$, $y_2 = 784.890$ e $y_3 = 806.269$. Portanto, por essas estimativas,

Figura 33 – Gráficos com os resultados dos modelos de regressões.

Fonte: Elaborada pelo autor.

Nota – Dados de treinamento e teste obtidos em experimentos no Koala. Resultados dos quatro algoritmos sobre o conjunto de dados *log* padronizado.

sabe-se que o *cluster* é o ambiente mais adequado para esses parâmetros. Para cada novo experimento, é possível fazer o mesmo procedimento para escolher corretamente o tipo de máquina. Importante ressaltar que a previsão usando SVM não é computacionalmente intensiva (SMOLA; SCHÖLKOPF, 2004), sendo um bom candidato para mecanismo principal do *Decision Maker*. O conjunto de variáveis discutidos no último exemplo não aparece em nenhum dos conjuntos de treinamento, e mesmo assim o algoritmo é capaz de fazer uma estimativa adequada. Portanto, o modelo treinado não está apenas memorizando dados anteriores, mas realmente aprendendo a partir dos dados.

4.3.3 Avaliação do modelo SVR

Utilizando o modelo obtido, foram realizadas algumas execuções para indicar o melhor ambiente de experimentos, tendo como parâmetros 4 tamanhos de proteína: 10, 50, 100 e 200 aminoácidos. O tamanho da população varia em 10, 50, 100 e 200 indivíduos e o número de

gerações foi fixado em 200. Nas figuras a seguir o melhor ambiente para cada cenário é mostrado. As configurações de máquina com que o modelo foi treinado foram apresentadas na Tabela 20, portanto refletem as mesmas capacidades computacionais. O SVR analisou esses três tipos denominados como “*desktop*”, “*cluster*” e “*nuvem*”, representando uma capacidade crescente de recursos, porém não são feitas análises quanto à transferências de dados ou particularidades de servidor *online*, por exemplo. Tampouco são diferenciados os *clusters*, sendo considerado um único tipo de máquina.

Os valores menores nos gráficos indicam os melhores, representando o tipo de máquina mais adequado para os parâmetros do experimento de PSP. Os resultados demonstram que a nuvem não é a melhor escolha para todas as ocasiões. As análises são feitas pelo *Decision Maker* segundo o modelo de treinamento do SVR.

No primeiro cenário com a proteína de tamanho 10, o ambiente que apresentou o melhor comportamento foi a máquina local, como visto na Figura 34. À medida que os parâmetros de entrada crescem, a diferença tornou-se menor, mas o padrão é preservado. Isso indica o bom desempenho de computadores *desktop* para proteínas menores, que requerem menos capacidade.

Figura 34 – Gráfico dos resultados gerados pelo *Decision Maker*. Primeiro cenário.

Fonte: Elaborada pelo autor.

Nota – Nesse cenário, as máquinas mais indicadas para o processamento são as do tipo *desktop*. Tempo de resposta em minutos.

No segundo cenário, com o tamanho da proteína base de 50 aminoácidos, o melhor ambiente em qualquer das variações foi para as máquinas do tipo *cluster*. As informações extraídas do SVR indicam que a partir desta proteína, a máquina do tipo *desktop* começa a apresentar tempos de respostas piorados, limitando sua capacidade de atuação. Os resultados são apresentados na Figura 35.

No terceiro cenário, os melhores resultados foram alcançados pelas máquinas da nuvem.

Figura 35 – Gráfico dos resultados gerados pelo *Decision Maker*. Segundo cenário.

Fonte: Elaborada pelo autor.

Nota – Nesse cenário, as máquinas mais indicadas para o processamento são as do tipo *cluster*. Tempo de resposta em minutos.

Os resultados das máquinas *cluster* ainda são competitivos, apresentando valores próximos aos da concorrente. As máquinas *desktop* por sua vez apresentam tempos de respostas muito elevados, sendo considerados impeditivos nesse caso, como visto na Figura 36.

Figura 36 – Gráfico dos resultados gerados pelo *Decision Maker*. Terceiro cenário.

Fonte: Elaborada pelo autor.

Nota – Nesse cenário, as máquinas mais indicadas para o processamento são as do tipo *nuvem*. Tempo de resposta em minutos.

No quarto cenário, a distância entre os resultados alcançados pelo *cluster* e as máquinas

da nuvem aumentou, como exibido na Figura 37. Para os parâmetros dessa rodada, máquinas de menor capacidade já são consideradas inadequadas para o processamento. O *cluster* pode ser considerado uma alternativa para usuários de baixo orçamento ou que não tenham requisitos de tempo de resposta como imprescindíveis. A saída do SVR indica que proteínas maiores do que essa devem ser processadas em máquinas de capacidade superior, mais comumente encontradas em servidores de nuvem.

Figura 37 – Gráfico dos resultados gerados pelo *Decision Maker*. Quarto cenário.

Fonte: Elaborada pelo autor.

Nota – Nesse cenário, as máquinas mais indicadas para o processamento são as do tipo nuvem. Tempo de resposta em minutos.

Usando SVR em dados obtidos de experiências anteriores, foi possível definir um modelo para prever corretamente a melhor configuração de ambiente de experimentos de acordo com os requisitos preliminares. O modelo é capaz de selecionar o ambiente usando as variáveis tamanho da cadeia de proteína, tamanho da população e o número de gerações para o algoritmo genético. Os resultados mostram que nem sempre o ambiente da nuvem é a melhor opção, já que outros ambientes, como computador local ou um *cluster* conseguiram resultados melhores, conforme o cenário. O SVR tem a capacidade de melhorar suas previsões, conforme o conjunto de dados for atualizado e aprofundado. Isso indica que seu mecanismo estará em constante atualização, fornecendo informações importantes ao sistema e aos usuários. Este modelo é o núcleo do *Decision Maker*, um módulo de identificação de ambiente de processamento, a ser integrado ao *Broker* da MDAPSP. Esse módulo é o embrião da arquitetura geral, por meio do qual importantes passos decisórios e otimizações podem ser ofertados.

4.4 Considerações Finais

Este Capítulo apresentou um estudo detalhado do Galaxy Koala, uma versão personalizada do *gateway* geral, demonstrando suas funcionalidades e módulos. O *framework* foi instalado localmente em *clusters* de computadores para permitir avaliações de capacidade e experimentos variados. Foram avaliadas diferentes cenários de experimentos, nos quais alguns dos gargalos do sistema foram identificados. Os experimentos permitiram também um estudo da carga de trabalho imposta pelas cadeias de proteínas nos preditores do Koala, gerando a criação de modelos sintéticos dessa carga, mapeados em arquivos de *workflow* científico. Foi feito ainda um estudo preliminar de monitoramento dos componentes do Koala, demonstrando a importância de tal avaliação em um DCI. Finalmente, após a exploração dos mecanismos e da revisão de soluções da literatura, é proposto o *Decision Maker*, um componente para ambientes distribuídos heterogêneos, capaz de indicar o melhor conjunto de máquinas baseado em histórico de execuções. Tal módulo está no cerne da arquitetura conceitual e deverá ter papel importante em um futuro protótipo. As avaliações e a complexidade dos mecanismos envolvidos na proposta da MDAPSP conduzem para a decisão de um sistema simulado inicial, no qual testes gerais de capacidade e compatibilidade possam ser realizados antes da construção da arquitetura em ambiente real.

MDAPSP

5.1 Considerações iniciais

Este Capítulo apresenta os módulos e a metodologia de funcionamento da MDAPSP. Um dos objetivos é tornar mais simples a realização de experimentos científicos computacionais de *gateways* científicos utilizando a modelagem proposta, auxiliando na utilização de ambientes de tecnologia avançada e facilitando a interação com padrões e métodos. A arquitetura é proposta de forma conceitual, pontuando a organização de seus módulos e a necessidade de cada um deles. Essa arquitetura é validada posteriormente em um modelo simulado.

5.2 A Arquitetura Conceitual

A MDAPSP é uma arquitetura base para oferecimento de recursos computacionais para pesquisadores que utilizem *workflows* científicos, mais especificamente em Bioinformática. Busca manter a facilidade de uso dos *gateways* científicos, permitindo o acesso de usuários com pouca experiência a ambientes computacionais avançados e otimizar o desempenho conforme disponibilidade de máquinas e requisitos de QoS. Essa arquitetura compreende uma estrutura modularizada, permitindo integração de tecnologias, atualizações e substituições de suas partes. Fornece abstração de complexidade de tecnologias e permite a personalização com a adição de aplicativos, ferramentas e módulos, conforme as necessidades dos usuários. Para isso, os modelos básicos de publicação e descoberta de recursos são reconhecidamente funcionais (LI; SONG, 2010). Modularidade e escalabilidade são importantes para agregar diferentes pesquisas e acolher novas abordagens. Diversos experimentos devem ser suportados simultaneamente, com fundamentação e objetivos distintos (GARZA; AICHELER; KOHLBACHER, 2016). O atendimento dos clientes pode ser categorizado em classes com diferenciação de acesso e dos requisitos, o que aumenta a complexidade da infraestrutura.

A oferta de recursos para o Galaxy (assim como outros *gateways*) podem ser pensados

de forma aprimorada, oferecendo avanço para ferramentas e experimentos em Bioinformática. A arquitetura MDAPSP oferece a possibilidade de testar vários métodos, aumentando a flexibilidade, considerando também o processamento e armazenamento em ambiente heterogêneo, aliado a aspectos de desempenho. A Figura 38 mostra uma visão de alto nível da arquitetura. Os clientes podem acessar os recursos via página *Web*. Feita a admissão, o usuário deve escolher o tipo de estrutura científica necessária para realizar os experimentos, uma vez que vários domínios podem registrar seus serviços. Do *gateway* principal, o usuário pode gerenciar procedimentos, *workflows* e dados de proveniência, coletados e armazenados em banco de dados. Além disso, o *gateway* pode dividir os usuários em classes. Uma combinação de parâmetros dos usuários e experimentos é capaz de guiar a combinação de recursos computacionais para uma plataforma otimizada.

Figura 38 – Ilustração abstrata dos módulos da MDAPSP e sua contribuição em *e-science*.

Fonte: Elaborada pelo autor.

Experimentos em PSP contam com diversos tipos de preditores (COCK *et al.*, 2013), os quais demandam muitos recursos. Esses preditores variam desde o modelo de processamento quanto ao volume de dados produzidos (SLEATOR, 2012). A hospedagem em sistemas isolados é um complicador (PHUOC; KIM, 2011), situação em que uma infraestrutura de integração pode contribuir. Semelhante a iniciativas como o *WeFold* e o *I-TASSER*, a MDAPSP pode compor um ambiente de *coopetition* (*cooperation* + *competition*) e de formação de ambientes integrados de experimentação. Tal sistema deve atender demandas variadas, mantendo QoS e fornecendo alternativas de colaboração (CRIVELLI *et al.*, 2013). O esquema organizacional proporcionado por *workflows* é uma das chaves para integração de soluções e compartilhamento de informações

de pesquisa, inclusive dados intermediários e de configuração (PLANKENSTEINER *et al.*, 2013). Os *workflows* científicos identificam passos e organizam um esforço colaborativo de tarefas gerenciáveis. Isso auxilia a identificação de gargalos de desempenho e dependências de execução (GARZA *et al.*, 2016). Para executar um *workflow*, um SWMS gera um Plano de Execução de *Workflow* (do inglês *Workflow Execution Planner* (WEP)), que captura decisões de otimização e diretivas de execução. Alguns dos pré-requisitos do WEP são seleção de parâmetros, análise do experimento e visualização do andamento.

A criação de SWMS específicos atende apenas à comunidade alvo da ferramenta. Essa é uma limitação porque a interoperabilidade de *workflows* é uma característica importante atualmente (GARZA *et al.*, 2016). Os *gateways* científicos se propõem a fazer a intermediação entre usuários e recursos computacionais e podem integrar os recursos de SWMS. A formação de comunidades de usuários, o uso de plataformas abertas e a opção por *gateways* que incentivam a integração e modificação de seus mecanismos, aliado a modelos acessíveis de computação é uma das formas de facilitar o acesso aos recursos computacionais avançados. Diversos dos *gateways* científicos disponíveis não são flexíveis, por serem de código fechado ou não admitirem extensões, o que limita essas ferramentas. Para compreender a estruturação em módulos da MDAPSP é preciso identificar a atuação da arquitetura no contexto de *e-science* e como ela pode atender a algumas das necessidades atuais desse campo. Os *gateways* podem ser descritos em uma estrutura de camadas, como a disposta na Figura 39. A MDAPSP atua nas camadas de Infraestrutura e Execução WEP, mais especificamente nos blocos de provisionamento e escalonamento, destacados na Figura 39.

Essas camadas podem ser detalhadas da seguinte forma:

- **Apresentação:** no topo da hierarquia, faz a interface com o usuário, para interação com os mecanismos do SWMS em todos os estágios do ciclo de vida de um *workflow*. A interface pode ser textual ou gráfica (DIAS *et al.*, 2013).
- **Serviços de Usuário:** fornece informações detalhadas com as quais é possível tomar decisões e direcionar os *workflows*. Monitoramento traz informações do estado dos experimentos. Redirecionamento de *workflows* é a interação entre o SWMS e o usuário para controlar o progresso da execução ou a configuração de parâmetros. Compartilhamento promove a partilha de informações, como desenho do *workflow*, dados de entrada e resultados. Proveniência disponibiliza dados fundamentais para reprodutibilidade, interpretação de resultados e diagnóstico de problemas (PLANKENSTEINER *et al.*, 2013).
- **Geração WEP:** gera um plano conforme o desenho do *workflow*. Refatoração inclui transformações, remoção de atividades e divisão de tarefas. A paralelização explora divisão de tarefas concretas para execução, encapsula os dados relacionados e uma atividade passa a corresponder a diversas tarefas paralelas. Otimização de *workflow* diz respeito a captura

Figura 39 – Organização em camadas dos SWMS

Fonte: Adaptada de [Liu et al. \(2015\)](#).

Nota – A MDAPSP atua entre as camadas de infraestrutura e WEP.

de informações dos processos de refatoração e paralelização e adiciona instruções para gerar um novo WEP ([VAHI et al., 2013](#)).

- **Execução WEP:** cuida do escalonamento, execução de tarefas e tolerância a falhas do *workflow*. Pode contar com um plano de escalonamento, conforme a estrutura das tarefas. A execução pode ser realizada nos *gateways* ou em mecanismos externos. Nessa fase são produzidos resultados e coletados dados de proveniência da execução. Mecanismos de tolerância a falhas lidam com erros na execução de tarefas, garantem disponibilidade e resiliência a uma execução ([LIU et al., 2015](#)).
- **Infraestrutura:** provisionamento dos meios de computação. A interface conecta os meios físicos às máquinas e mecanismos de armazenamento. Nessa camada são definidas as limitações do ambiente. Armazenamento pode conter *cache*, persistência, bases de dados ou discos compartilhados ([DEELMAN et al., 2015](#)).

A variabilidade de tecnologias que podem compor tal infraestrutura é alta, assim como os pontos de investigação e melhorias necessárias. Alguns exemplos podem ser destacados como estratégias de escalonamento que incluem fragmentos de *workflows*, tarefas individuais ou

bag of tasks, mecanismos de tolerância a falhas podem variar entre replicação, *check-points* e substituição de componentes (GANGA; KARTHIK, 2013) e a otimização pode incluir ajustes no planejamento da execução ou intervenção em tempo de execução (DIAS *et al.*, 2013).

Recursos avançados de computação são cada vez mais necessários para permitir o manejo de dados (STEWART *et al.*, 2011). Tirar o máximo desses recursos requer uma compreensão completa dos dados e do problema computacional envolvido, incluindo desafios como acesso, gerenciamento e transferência de informações (TORTEROLO *et al.*, 2009). A integração de ambientes computacionais pode oferecer alta velocidade e redução de custo (SCHADT *et al.*, 2010). A movimentação de grandes volumes pela rede é limitada pela largura de banda (YUAN *et al.*, 2013), assim como garantir acesso, se esses dados forem armazenados de forma centralizada. Uma alternativa é levar a HPC para onde estão os dados (ARMBRUST *et al.*, 2009). A MDAPSP pode fornecer acesso a esses sistemas computacionais e direcionamentos para escolher o ambiente ideal conforme os parâmetros do usuário. Com a construção de SOA é possível ter módulos que atuem tanto recebendo informações, quanto mantendo instâncias de máquina onde os dados estão hospedados.

Há uma tendência de ofertar os recursos dos *gateways* científicos em servidores na nuvem, devido às vantagens conhecidas (MENON *et al.*, 2012). Apesar da acessibilidade, a computação em nuvem tem complicadores como gerência, localidade e preço de aquisição. Os custos incluem sacrificar o controle do *hardware* subjacente e depender que grandes conjuntos de dados sejam transferidos para processamento (SCHADT *et al.*, 2010). *Clusters* de computadores podem ser otimizados para muitas classes diferentes de aplicações computacionalmente intensas e máquinas com GPU promovem ganho de desempenho notável para casos de grande paralelismo, sendo considerados aceleradores de baixo custo (LINDERMAN *et al.*, 2010). A computação heterogênea fornece um ambiente computacional ainda mais flexível e barato que melhora o desempenho e a eficiência expondo os recursos arquitetônicos aos programadores (SANDES *et al.*, 2016b). Entretanto, executar nesses ambientes requer conhecimentos de computação significativos (SCHADT *et al.*, 2010).

Os modelos de computação podem ser considerados de forma diversa, buscando uma variação nos níveis de recursos oferecidos. Diversas camadas de *hardware* e *software* podem compor o ambiente computacional, como mostrado na Figura 40. No primeiro nível fica a estrutura física que abriga o *hardware*, como infraestrutura de rede e servidores. A seguir aparecem a camada de virtualização de recursos, seguida pela camada de SO. Finalmente, existem as infraestruturas de *software* e as camadas de aplicação. Os diferentes tipos de computação (à direita na figura) podem ser diferenciados se estão sob o controle do usuário (linha contínua) ou por terceiros (linhas tracejadas). Os serviços de nuvem e grade são os mais adequados para aplicações com paralelismo livremente acoplado ou grosso. Tecnologias de cada um dos modelos como MPI e MapReduce podem ser importantes conforme os serviços oferecidos. Os sistemas heterogêneos incluem aceleradores de hardware especializados (SCHADT *et al.*, 2010). Tal

variabilidade é fundamental para a concepção tecnológica da MDAPSP, na qual ambientes heterogêneos de DCI devem integrar a infraestrutura de recursos.

Figura 40 – Organização das infraestruturas de HPC.

Fonte: Adaptada de [Schadt et al. \(2010\)](#).

5.3 Organização geral

MDAPSP busca garantir o bom desempenho de *gateways* científicos, fornecendo uma infraestrutura adequada de recursos, além de permitir o acesso de usuários com pouca experiência a ambientes computacionais avançados, mantendo o desempenho conforme requisitos de QoS e disponibilidade de máquinas. Em sua primeira versão, deve fornecer recursos computacionais em conformidade com as necessidades dos experimentos de PSP. A organização em módulos promove o fraco acoplamento, permite a integração de tecnologias e alterações dos componentes, tanto em casos de mau funcionamento como nas etapas de evolução da arquitetura. A Figura 41 mostra os principais módulos da MDAPSP e sua distribuição. O cliente acessa os recursos disponíveis em uma interface, que pode ser a mesma do *gateway*, sendo atualmente baseado no Galaxy Koala. A interface é a entrada para as ferramentas fornecidas e para a execução dos experimentos. A estrutura segue a seguinte organização:

- **Cliente** - os usuários interessados em soluções computacionais para suas pesquisas.
- **Interface SCINT (Science Interface)** - a arquitetura deve fornecer uma interface genérica, onde os clientes podem operar o sistema e gerenciar seus fluxos de trabalho e dados.

Figura 41 – Organização dos módulos da MDAPSP.

Fonte: Elaborada pelo autor.

Interfaces personalizadas podem integrar a arquitetura, tornando-a adaptável a muitos contextos;

- **Gerenciamento de dados** - disponível nos *gateways* científicos, permite ao usuário recuperar e fazer o *upload* de informações sobre os experimentos, *workflows* e dados de proveniência disponibilizados;
- **Seleção de gateway** - a primeira versão da arquitetura não oferece opções nesse contexto. As versões futuras podem ser independentes de um *gateway* específico e este módulo deve fornecer as diversas ferramentas de outros sistemas;
- **Core** - o módulo de orquestração da arquitetura. Nele os mecanismos são definidos conforme as opções e parâmetros dos usuários. Ele conta com algoritmos de otimização, auxílio à decisão e histórico de execuções. Um dos seus módulos principais tem a função de identificar a infraestrutura mais indicada para um dado experimento. O *core* também tem a função de comunicação com os demais módulos e coletar dados de QoS da arquitetura;
- **Repositório de Aplicações** - estruturas que mantêm informações atualizadas dos aplicativos, como localização, disponibilidade, requisitos básicos e versão. O usuário tem a opção de orientar a seleção, mas os mecanismos automáticos estão presentes.
- **Registro de workflow** - armazenamento de *workflows* e informações de proveniência, que permite uma compreensão completa das etapas de experiências, reprodutibilidade e composição de projetos novos e mais complexos;

- **Ambiente de processamento** - coleção de máquinas com capacidades variáveis, capaz de atender a uma grande e diversificada população de usuários. Os clientes são divididos de acordo com suas necessidades, investimentos ou afiliação.

Nessa organização, a arquitetura pode funcionar com a interação entre todos os módulos ou mesmo na ausência de alguns deles. O fraco acoplamento é vantajoso pois permite que sejam feitas trocas no componentes ou selecionados caminhos otimizados conforme as necessidades do usuário. Em sua versão completa, a arquitetura MDAPSP é organizada em módulos independentes e interligados, como visto na Figura 42. Essa distribuição permite uma compreensão da estrutura geral, do fluxo de informações, dos componentes principais e dos desafios envolvidos em sua disponibilização. Seguem a descrição dos módulos e as suas interações.

Figura 42 – Organização completa dos módulos da MDAPSP

Fonte: Elaborada pelo autor.

- **Protein Data Storage (PDS)** - é o local de armazenamento de dados de proteínas, a ser acessado para coleta das informações. Sendo de uso geral, pode armazenar informações de outros campos de pesquisa.
- **PDS - Registry** - registra informações sobre a localização e as características da proteína, agilizando a busca e permitindo a publicação de novidades em seu mecanismo.

- **Protein Data Pattern (PDPAT)** - é um módulo do *Broker* que armazena informações de padronização do PDS - *Registry*. Auxilia na definição de modelos de dados, de forma a auxiliar em procedimentos de integração ou conversão de informações.
- **Service Provider** - módulo geral que abriga os recursos computacionais e os provedores que processam os experimentos propriamente ditos. Conta com sub-módulos para organização e desempenho.
 - **Protein Selector (PS)** - responsável pela obtenção dos dados completos de proteína do PDS. Pode incluir *workflows* completos e dados de processamento de tais sequências.
 - **Admission Control** - responsável por avaliar se os provedores estão disponíveis e prontos para atender os pedidos dos clientes e controlar o acesso concorrente.
 - **Service Provider Interface (SPI)** - interface do provedor de serviços que recebe informações dos provedores, como localização e disponibilidade. É responsável pela troca de dados com o *Broker* e os provedores que atendem às requisições.
- **Broker** - módulo central de orquestração da arquitetura. É responsável por mediar as comunicações entre os componentes. Também abriga os módulos de decisão e de coleta de informação. Fica encarregado de encaminhar as requisições dos clientes e retornar as respostas aos mesmos.
 - **Decison Maker** - módulo projetado para auxiliar o processo de escolher onde um experimento deve ser executado. Faz análise de experimentos anteriores para definir o melhor ambiente computacional para um determinado experimento. O Algoritmo do *Decision Maker* é baseado na avaliação de dados históricos, como demonstrado na seção 4.3. Esse módulo é o primeiro da MDAPSP a ter uma versão inicial desenvolvida, sendo considerado o embrião da arquitetura geral.
 - **Service Provider Quality of Service (SPQoS)** - responsável por manter informações atualizadas sobre o status dos provedores. Uma vez descoberta a localização e disponibilidade do provedor, o módulo obtém informações de QoS como CPU, Memória e Disco, em unidades de taxa de utilização ou outro índice de estado. Deverá contar com mecanismos de monitoramento como discutido na seção 4.2.4.
 - **Service Provider Type (SPT)** - armazena informações sobre o tipo de provedor de serviços, como suas características e tecnologias internas. Com base nessas informações, é possível indicar o provedor mais indicado, se ele é local ou remoto e o tipo de *framework* de execução como MPI, OpenMP ou CUDA, por exemplo. Os usuários podem guiar a decisão.
 - **Cache Module** - destinado a armazenar as operações mais recentes ou as mais frequentes. Recurso de otimização de buscas e recuperações de dados. Para obter

um bom desempenho, o *cache module* depende de várias execuções, que criam um histórico. Com base nesse rastreamento, é possível acelerar o funcionamento do *Decision Maker* que vincula o tipo de proteína e tipo de provedor de serviços com o ambiente de provisionamento.

- **Search Manager** - informa ao PDS - *Registry* os dados inseridos pelo usuário, para que faça a busca dos dados de proteína. Também se comunica com o *Decision Maker* para que esse calcule os requisitos de máquina necessários.
- **Client Request Interface** - este módulo recebe os pedidos da Aplicação Cliente e negocia a conexão com o *Broker* da arquitetura.
- **Results Data Storage (RDS)** - armazena todas as informações relacionadas ao processamento da solicitação do cliente, desde o início da requisição, dados e parâmetros dos experimentos, provedor utilizado e etc. Tanto o PDS como o RDS podem ser um banco de dados local, um banco de dados distribuído ou um sistema de arquivos distribuído.
- **Aplicação Cliente** - o usuário interage com a arquitetura por meio de aplicações disponíveis na interface *Web*, acessível por dispositivos comuns (*desktop*, celular, *tablet*). Permite a visualização e gerenciamento de dados e aplicativos. A Figura 43 mostra um diagrama de sequência da execução de um experimento na MDAPSP.

Figura 43 – Diagrama de sequência de um experimento na MDAPSP.

Fonte: Elaborada pelo autor.

Algumas fases independentes e detalhes importantes também devem ser mencionados. A inserção das informações de proteína no banco de dados distribuído não é exclusiva de experi-

mentos realizados na arquitetura. Resultados de outros *gateways* científicos e *frameworks* podem agregar essa base. Quando uma operação for realizada, o PDS - *Registry* atualiza os metadados dos arquivos, tais como localização, tipo e tamanho. O acesso aos resultados armazenados no módulo RDS também deve estar aberto e disponível para quaisquer interessados. O acesso e análise de resultados pode ser feito na interface da MDAPSP. O RDS mantém registros completos dos experimentos. Este banco de dados precisa considerar informações gerais, tais como:

- usuário solicitante;
- tempo de resposta total a partir da solicitação;
- informações do provedor de serviços (IP) e tipo (com GPU, CPU ou ambos).

Uma das questões importantes em PSP refere-se aos arquivos de entrada e saída dos experimentos. A variação no tamanho dos mesmos é grande, o que pode levar a alterações nas estratégias de armazenamento e movimentação. Nesse sentido, a portabilidade de mecanismos da arquitetura permite a execução em localidades diferentes, evitando trânsito desnecessário de informações via rede. Um outro ponto fundamental é a disponibilização de modelos de predição diversificados, o que permite a comparação de resultados e a evolução das soluções. Essa variabilidade proporciona a criação de procedimentos otimizados e até mesmo a mitigação das falhas encontradas em projetos prévios. O ambiente integrativo também pode ser empregado para a reprodução dos experimentos e validação das técnicas.

É importante fazer a medição correta da carga máxima suportada pelo sistema para evitar problemas de utilização. A previsão de carga é um dos pontos-chave que será oferecido como serviço, para estimar o custo computacional exigido pelos experimentos. Essa deverá ser uma etapa do mecanismo de previsão de carga a ser disponibilizado, em que o objetivo é aproximar a carga ideal de utilização para cada tipo de preditor utilizado. Em sua última versão, o projeto *WeFold* trabalhou com aproximadamente 100 problemas simultaneamente no sistema de *coopetition*, o que pode ser um bom parâmetro de utilização do protótipo da MDAPSP ([WEFOLD, 2013](#)). Velocidade de processamento, transferência de dados, consumo de memória, espaço em disco e consumo de processamento serão alguns dos parâmetros avaliados na MDAPSP uma vez que esteja completamente funcional, para que valores otimizados estejam disponíveis conforme as necessidades. Informações desse tipo usualmente não são avaliadas por *gateways* científicos, apesar de serem informações imprescindíveis para um bom desempenho.

Devido à complexidade dos mecanismos envolvidos, das grandezas em volume de dados e configurações de máquinas necessárias, a MDAPSP é desenvolvida inicialmente em um modelo de simulação computacional que permite uma análise precisa de suas capacidades funcionais. Nesse modelo simulado são avaliadas as influências da carga de trabalho de *workflows* científicos, a eficiência de um modelo heterogêneo de máquinas para provisão de recursos e a capacidade de escalonamento de tarefas conforme as necessidades de cada experimento. Com essa modelagem

se estabelece a base para o futuro protótipo da arquitetura e alguns dos padrões necessários para seu bom funcionamento.

5.4 Considerações finais

Este Capítulo apresenta a base conceitual da MDAPSP e seus módulos gerais. O mecanismo do *Decision Maker* destaca-se como um dos módulos centrais, integrando o *broker* dessa arquitetura. Entende-se que essa SOA apresenta uma alternativa realística na modelagem de *gateways* científicos e contribua na evolução dos modelos atuais de *e-science*. Os desafios envolvidos na prototipação de tal estrutura são diversos, derivando em problemas de múltiplas áreas. A multidisciplinaridade do projeto é um complicador que envolve esforços concentrados e estudos variados. Tal complexidade guia a decisão de validar o modelo da arquitetura por meio de simulação computacional. Dessa forma, podem ser feitos estudos diversos de compatibilidade, variações de algoritmos de controle e extrapolação dos dados de carga de trabalho. O próximo capítulo aborda os módulos desenvolvidos como prova de conceito e funcionamento dos mesmos.

MODELO DE SIMULAÇÃO

6.1 Considerações iniciais

Este Capítulo detalha a implementação de módulos da MDAPSP em um modelo simulado, que permite a averiguação de suas capacidades e validação do conceito. Descreve os detalhes do programa de simulação principal para o ambiente da MDAPSP, dos algoritmos base e os experimentos visando modelo de escalonamento de tarefas. A estruturação tem por objetivo extrapolar os dados coletados e fornecer uma visão técnica da arquitetura, por meio de testes de capacidade e avaliação de algoritmos.

6.2 Ferramentas

O ambiente de simulação foi desenvolvido com auxílio do IDE NetBeans versão 8.2. O projeto criado utiliza-se da API do CloudSim Simulation Toolkit versão 4.0 (CALHEIROS *et al.*, 2011) e do *framework* de extensão WorkflowSim versão 1.0 (CHEN; DEELMAN, 2012) com as bibliotecas de funcionalidades de apoio Commons Math Sources versão 3.3.2 (MATH, 2016), Flanagan 1.7.0 (FLANAGAN, 2016) e JDOM 2.0.0 (HUNTER; LEAR, 2014). Também foram utilizadas algumas classes de apoio do *framework* de extensão BEQoS, sendo elas *DatacenterConfiguration*, *Utilities* e *VMConfiguration* (PARDO *et al.*, 2016).

6.3 Descrição dos algoritmos

A versão 1.0 do *WorkflowSim* limita a dois o número de *datacenters* mapeáveis em uma arquitetura. Não há especificações no simulador que indiquem a razão para essa abordagem, nem outros trabalhos na literatura atual que utilizem quantidades superiores a dois *datacenters*. Como a MDAPSP considera um mínimo de três ambientes de provisionamento de recursos, foi preciso alterar o funcionamento base do algoritmo de alocação de recursos do simulador, reparo este

que constitui uma contribuição direta ao *WorkflowSim* a ser disponibilizada para o *framework*, denominada *VM Allocation Policy for Multi Datacenters* (APMD). Esse algoritmo permite o escalonamento das tarefas de acordo com as capacidades de máquina, capacitando a arquitetura a oferecer as otimizações conforme a técnica proposta com o *Decision Maker*. Eventualmente, essa decisão pode ser deixada a cargo do usuário, sendo oferecidas opções automatizadas ou personalizadas conforme a preferência do usuário. O foco atual é mostrar o funcionamento do modelo automático, seguindo os procedimentos dos *gateways* científicos que devem atender usuários de muitos campos do conhecimento.

Considerando a arquitetura conceitual da MDAPSP apresentada no Capítulo 5, foi elaborada a modelagem dos módulos principais da mesma em ambiente de simulação. Os módulos não considerados atualmente são importantes no contexto geral da arquitetura, entretanto eles requerem estudos aprofundados e variações tecnológicas não contempladas na atual pesquisa. De qualquer forma, a estrutura fundamental da MDAPSP foi implementada e avaliada, conforme é possível verificar na Figura 44, destacando os módulos envolvidos.

Figura 44 – Ilustração dos módulos da MDAPSP considerados no modelo simulado.

Fonte: Elaborada pelo autor.

O programa de simulação geral foi implementado com o objetivo de modelar o ambiente de infraestrutura real descrito conforme as capacidades de máquina de cinco *datacenters* diferentes, sendo um computador do tipo *workstation*, três segmentos de *clusters* de computadores e uma unidade de computação em ambiente de nuvem. Ressalte-se que alguns elementos da

infraestrutura real são abstraídos para adaptá-los às necessidades e padrões exigidos pelo *WorkflowSim*, descritas em código no programa principal. Inicialmente são descritas as caracterizações de infraestrutura dos *datacenters* por meio de instâncias da classe de objeto *DatacenterConfiguration* (objeto da API BEQoS). Nesses objetos são configuradas todas as informações pertinentes à descrição dos *datacenters*, tais como número de nós, quantidade de memória RAM por nó, quantidade de núcleos de processamento, capacidade de armazenamento em disco, taxa de processamento em MIPS por nó, dentre muitas outras informações. O Código 1 a seguir apresenta, como exemplo, a configuração criada para modelar a infraestrutura do *cluster* Cosmos. Apenas alguns parâmetros são demonstrados abaixo, uma vez que há diversas outras opções de fatores configuráveis, por exemplo custo de alocação por VM e fuso horário.

Código-fonte 1 – Configuração do *cluster* Cosmos

```

1: DatacenterConfiguration confCosmos = new
    DatacenterConfiguration();
2: confCosmos.hostId = 1000; // Identificação do datacenter
3: confCosmos.mips = 4000; // MIPS
4: confCosmos.num_cores = 4; // Número de núcleos
5: confCosmos.num_hosts = 10; // Total de nós
6: confCosmos.ram = 8000; //MBytes
7: confCosmos.bw = 10000; // Largura de banda em Mbps
8: confCosmos.storage = 500000; //MBytes
9: confCosmos.arch = "x64"; // Arquitetura de Computador
10: confCosmos.os = "Linux"; // SO
11: confCosmos.vmm = "Xen"; // Virtualizador
  
```

Na sequência do programa de simulação principal, é preciso informar as configurações equivalentes às máquinas virtuais que serão posteriormente instanciadas no ambiente de cada um dos *datacenters*. Na modelagem considerada nesse trabalho, é definida uma (VM) por nó físico (*Host*) em cada *datacenter*, com objetivo de efetuar a avaliação da capacidade total desses nós. Variações nessa política podem ser implementadas de forma a ampliar as análises em cenários de simulação futuros. As VMs são configuradas por meio do objeto *VMConfiguration*, da (API BEQoS), ocupando todos os recursos dos *hosts* como CPU, memória, disco, MIPS, entre outros. O Código 2 apresenta um exemplo de como é criado o objeto de configuração de VM referente ao *cluster* Cosmos.

Código-fonte 2 – Configuração de Máquina Virtual no *cluster* Cosmos

```

1: int vmNumA2Cosmos = 10; // Número de VMs
2: VMConfiguration vmConfCosmos = new VMConfiguration();
3: vmConfCosmos.bw = 1000; // Largura de banda
4: vmConfCosmos.pesNumber = 4;
  
```

```

5: vmConfCosmos.mips = 4000; // MIPS
6: vmConfCosmos.vmm = "Xen"; // Virtualizador
7: vmConfCosmos.size = 250000; //MBytes
8: vmConfCosmos.ram = 4000; //MBytes

```

Para simular o ambiente real considerado, foram criadas cinco configurações de *data-center* e cinco configurações para as VMs. Os objetos que emulam os *datacenters* são da classe de objeto *WorkflowDatacenter* (da API *WorkflowSim*). Os *datacenters* são denominados: *DC_Workstation*, *DC_Cosmos*, *DC_Andromeda*, *DC_Halley* e *DC_SoftLayer*. O *DC_Workstation* representa a infraestrutura na qual se aloca o computador do tipo *workstation* da marca Dell, encontrado no mercado atual. Os *datacenters* *DC_Cosmos*, *DC_Andromeda* e *DC_Halley* emulam os *clusters* de computadores disponíveis no LASDPC do ICMC - USP. E o *DC_SoftLayer* emula a contratação de recursos de nuvem da empresa SoftLayer (vinculada à IBM) na modalidade de serviço Bare Metal. O Código 3 apresenta a instanciação dos objetos *WorkflowDatacenter* para cada um dos ambientes considerados.

Código-fonte 3 – Instanciação dos *datacenters*

```

1: WorkflowDatacenter dcA1 = Utilidades.createCustomDatacenter (
    DC_WorkStation, confA1);
2: WorkflowDatacenter dcA2Cosmos = Utilidades.
    createCustomDatacenter (DC_Cosmos, confCosmos);
3: WorkflowDatacenter dcA2Andromeda = Utilidades.
    createCustomDatacenter (DC_Andromeda, confAndromeda);
4: WorkflowDatacenter dcA2Halley = Utilidades.
    createCustomDatacenter (DC_Halley, confHalley);
5: WorkflowDatacenter dcA3 = Utilidades.createCustomDatacenter (
    DC_SoftLayer, confSoftLayer);

```

O ambiente de provimento de recursos computacionais modelado nessa descrição dos *datacenters* representa o módulo de provisionamento de serviços da MDAPSP. No modelo conceitual são previstos detalhamentos tecnológicos como tipo de programação paralela disponível e módulos de controle de admissão, os quais são abstraídos nessa simulação. A provisão no *WorkflowSim* contempla a definição dos diferentes tipos e capacidades de máquina como apresentado na Figura 45.

A seguir são declaradas no programa de simulação as informações referentes aos algoritmos de planejamento da execução do *workflow*, do escalonamento das tarefas e o padrão de operação para o sistema de arquivos distribuído usado no ambiente simulado, por meio dos objetos *Parameters* e *ReplicaCatalog* (pertencentes à API *WorkflowSim*). O Código 4 demonstra como esses valores foram definidos no programa. Esse ponto, na definição do simulador, é

Figura 45 – Ilustração do módulo de provisionamento da MDAPSP no modelo simulado.

Fonte: Elaborada pelo autor.

fundamental para os propósitos de construção do ambiente simulado e de definição dos aspectos a serem avaliados experimentalmente. No caso, a variação entre os algoritmos de escalonamento padrão e os desenvolvidos para a MDAPSP é definida nesse momento. Os algoritmos de planejamento e de sistema de arquivos são obtidos do ambiente padrão do *WorkflowSim*.

Código-fonte 4 – Seleção dos algoritmos de Escalonamento e Planejamento

```
1: Parameters.SchedulingAlgorithm sch_method = Parameters.
    SchedulingAlgorithm.MDSASCHED;
2: Parameters.PlanningAlgorithm pln_method = Parameters.
    PlanningAlgorithm.PLN1;
3: ReplicaCatalog.FileSystem file_system = ReplicaCatalog.
    FileSystem.SHARED;
```

Outros parâmetros de inicialização importantes que devem ser declarados na sequência são os de sobrecarga de processamento imposta no ambiente e os parâmetros de agrupamento do *workflow* que será processado durante a execução. Essas configurações são representadas pelas classes de objeto *OverheadParameters* e *ClusteringParameters* (ambas da API *WorkflowSim*). O presente trabalho não considera os atrasos gerados dinamicamente pelo simulador, nem o agrupamento de tarefas. Dessa forma as configurações que constam de tais objetos são definidas como nulas. É importante destacar essa informação porque ela pode influenciar na concepção geral do ambiente, na influência dos fatores e no rendimento geral da arquitetura. Em avaliações futuras, tais parâmetros podem ser considerados.

A partir desse ponto, o programa de simulação instancia os objetos referentes aos *datacenters*. São definidos alguns objetos das classes *WorkflowPlanner* e *WorkflowEngine* que inicializam o *framework* do *WorkflowSim*. O primeiro efetua o planejamento estático ou dinâmico do *workflow*, organizando a ordem de execução das tarefas e armazenando-as numa lista. Parte desse procedimento passa pela leitura do arquivo DAX feita com o *WorkflowParser*. Já o objeto *WorkflowEngine* é conectado ao *WorkflowPlanner* e sua função é servir como

camada de ocultação para tarefas administrativas de execução, tais como gerenciamento de VMs (criação, destruição) e gerenciamento das tarefas no formato de *cloudlets* (envio e recepção das *cloudlets* para as VMs instanciadas). O componente *WorkflowEngine* conecta-se, por sua vez, com o *WorkflowScheduler*, responsável por representar o escalonador principal do sistema, efetuando a distribuição da carga de trabalho para a infraestrutura disponível, conforme as regras de negócio definidas. Para o ambiente de simulação projetado no presente trabalho, foi utilizada uma instância de *WorkflowPlanner* e uma instância de *WorkflowEngine*. A seguir é feita a declaração das listas de VMs a serem criadas para cada *datacenter* e sua posterior submissão para o *WorkflowEngine*, o qual distribui as VMs adequadamente para os nós (*Hosts*), conforme as configurações anteriormente apresentadas.

Os *datacenters* são vinculados ao escalonador padrão (*WorkflowScheduler*) do ambiente de simulação. Esse processo se dá por meio do ID (número identificador) do escalonador, o qual é informado em conjunto com o ID do Datacenter no método *bindSchedulerDatacenter()* do objeto *WorkflowEngine* já instanciado. O Código 5 a seguir demonstra tal operação. O *WorkflowSim* permite utilizar mais de um escalonador no ambiente, porém, pelos requisitos atuais do projeto em relação ao ambiente físico real foi instanciado um único componente comum de gerência dos recursos e tarefas para os *datacenters*.

Código-fonte 5 – Vinculação dos *datacenters* ao escalonador.

```
1: wfEngine.bindSchedulerDatacenter(dcA1.getId(), 0);  
2: wfEngine.bindSchedulerDatacenter(dcA2Cosmos.getId(), 0);  
3: wfEngine.bindSchedulerDatacenter(dcA2Andromeda.getId(), 0);  
4: wfEngine.bindSchedulerDatacenter(dcA2Halley.getId(), 0);  
5: wfEngine.bindSchedulerDatacenter(dcA3.getId(), 0);
```

A seguir um ajuste de parametrização é feito para configurar a escala de valores considerada para a carga de trabalho do ambiente, principalmente para o tempo de execução (*runtime*) de cada tarefa. Isso é realizado por meio do método estático chamado *setRunScale* do objeto *Parameters*. Tal configuração permite obter variabilidade nos tempos de execução das tarefas, tornando os resultados de simulação fidedignos às cargas de trabalho executadas numa infraestrutura real. O valor padrão é uma escala na ordem de 1.0 (uma vez) o valor informado nos *workflows* de carga, especificamente no parâmetro de tempo de execução (*runtimes*) informados nos arquivos DAX. Essa variável pode ser alterada para representar diversos modelos de carga no ambiente. Um exemplo desse parâmetro de escala é visto no quadro baixo.

```
Parameters.setRuntimeScale(1.0);
```

Finalmente são incluídos comandos para o início da simulação, por meio do objeto *CloudSim* e do método *startSimulation()*. O simulador controla os eventos discretos ocorridos e

executa o controle do ambiente distribuído modelado por meio das regras de negócio programadas no bloco principal do programa de simulação.

6.3.1 Workflows de carga de trabalho

A carga de trabalho no simulador é implementada por meio de *workflows* científicos, nos quais são definidas as tarefas, os pesos individuais de cada *Task* e as transferências de arquivos entre as mesmas. Os referidos *workflows* científicos são descritos no formato de arquivos DAX, que conforme já explanado anteriormente, são uma representação descritiva com base no uso de *tags* XML. O *framework WorkflowSim* conta com diversos modelos prontos de *workflows* de várias áreas de pesquisa, também disponíveis no *gateway* Pegasus (DEELMAN *et al.*, 2015). Para realizar os experimentos de validação da arquitetura, esses *workflows* sintéticos podem ser utilizados como carga de trabalho e também para efeitos de comparação, dentre os quais foi escolhido o Montage, devido à sua estrutura de tarefas ser semelhante aos *workflows* de proteínas definidos nessa tese. Além desses, foi decidido empregar também cargas de trabalho baseadas nos experimentos de PSP, que puderam ser descritos em *workflows* conforme as informações das proteínas obtidas do PDB e as avaliações de experimentos realizados no Koala e mapeados para arquivos DAX como descrito na Seção 4.2.3, representando as tarefas, o tempo e o fluxo de execução, dentre outros parâmetros.

A diferença entre os arquivos DAX criados está na organização e quantidade de suas tarefas internas. Também é considerada a variação dos valores dos parâmetros “*Amin*”, “*Pop*” e “*Ger*”, os quais definem os critérios usados nos experimentos do preditor 2PG do Koala e representam a complexidade do tamanho da carga para o escalonador de tarefas da simulação. Nos arquivos DAX há três tipos de *tags* principais: $\langle job \rangle$, $\langle uses \rangle$ e $\langle child/parent \rangle$. Elas são utilizadas para descrever cada tarefa do *workflow*, os arquivos de entrada ou saída, a ordem da execução das tarefas e suas dependências. Por meio dos arquivos DAX é possível representar a execução dos *workflows* de proteínas de forma fidedigna, baseado nos estudos preliminares de execuções reais na infraestrutura do Koala. Logo, considera-se que tal medida dê maior crédito aos resultados experimentais obtidos, para avaliação do modelo simulado e para basear futuros planejamentos de experimentos. Essa modelagem dos experimentos de PSP em *workflows* científicos é uma abordagem inovadora, principalmente pela lacuna de estudos que utilizem tais procedimentos como modelo de carga. A dificuldade nesse mapeamento foi alta, motivo pelo qual abrange inicialmente apenas um dos preditores disponíveis no Koala, contemplando três proteínas de tamanhos de cadeias diferentes. Novos *workflows* de proteína podem ser definidos no modelo de carga sintético, podendo também ser avaliados no ambiente de simulação de forma a indicar seu comportamento em execuções reais, auxiliando em tomadas de decisão e configuração de capacidades de máquina. O mesmo é válido para os demais preditores do Koala e inclusive para aplicações de outros *gateways* científicos, generalizando a abrangência da MDAPSP.

Os *workflows* representam o módulo cliente da MDAPSP, contendo as requisições feitas ao sistema, informando os parâmetros de entrada, a ordem de processamento e a dependência entre as tarefas. Os arquivos DAX também representam o PDPAT, módulo interno do *Broker* da MDAPSP, responsável por descrever o padrão dos dados de proteínas e dos experimentos a serem executados. A Figura 46 demonstra o mapeamento de tais módulos na simulação. Na condição desse ambiente simulado, os módulos PDS, PDS-Registry e RDS projetados na arquitetura são abstraídos e devem ser estudados futuramente. Como descrito no Capítulo 5, o PDS compõe um armazém de informações de proteínas, em que os usuários podem coletar e depositar arquivos para construir uma base de dados pública. O PDS-Registry é um módulo com informações de propriedades desses arquivos, destinado a informar sobre o estado dos mesmos aos módulos da arquitetura e o RDS é a base onde se armazena resultados e dados de proveniência. Esses módulos serão importantes no contexto completo da arquitetura, promovendo dinamicidade e reprodutibilidade.

Figura 46 – Ilustração dos módulos cliente e PDPAT modelados no *WorkflowSim*.

Fonte: Elaborada pelo autor.

Nota – O PDPAT faz parte do *Broker*, porém o módulo principal é modelado em outra parte da simulação.

6.3.2 Delays de Rede

Configurações de latência de rede foram adicionadas ao ambiente de simulação por meio de uma variável de nome *delay*. Os *delays* de rede são incorporados na simulação por meio do acionamento do método *addLink* do objeto *NetworkTopology*, em que se pode configurar a latência entre duas entidades de simulação com a informação de seus IDs, da largura de banda do *link* (em Mb/s) e o tempo padrão da latência para cada comunicação realizada em segundos (ou instantes de *clock* do simulador).

A configuração dos *delays* de rede é fundamental em ambientes de simulação que busquem representar um ambiente real de forma fidedigna. Os atrasos de tempo devido à latência de rede ocorrem com muita frequência em aplicações do tipo *workflow*, pois há grande quantidade de comunicações durante a execução do programa distribuído e há várias ocorrências

Tabela 23 – Configuração de *links* de rede entre componentes da arquitetura.

ID Datacenter	Destino Datacenter	Largura de Banda (mbits/s)	Latência (segundos)
WorkflowScheduler (Broker)	DC_WorkStation	1000	0.10
WorkflowScheduler (Broker)	DC_Cosmos	1000	0.10
WorkflowScheduler (Broker)	DC_Andromeda	1000	0.10
WorkflowScheduler (Broker)	DC_Halley	1000	0.10
WorkflowScheduler (Broker)	DC_SoftLayer	1000	0.50
DC_WorkStation	DC_Cosmos	1000	0.25
DC_WorkStation	DC_Andromeda	1000	0.25
DC_WorkStation	DC_Halley	1000	0.25
DC_WorkStation	DC_SoftLayer	1000	0.50
DC_Cosmos	DC_Andromeda	1000	0.25
DC_Cosmos	DC_Halley	1000	0.25
DC_Cosmos	DC_SoftLayer	1000	0.50
DC_Andromeda	DC_Halley	1000	0.25
DC_Andromeda	DC_SoftLayer	1000	0.50
DC_Halley	DC_SoftLayer	1000	0.50

Fonte: Elaborada pelo autor.

de comunicação entre tarefas distintas, de modo a sincronizar dados e informações para que o programa finalize com seus resultados corretos. É oportuno descrever o critério utilizado para o estabelecimento das latências de comunicações de rede, possibilitando a reprodutibilidade dos experimentos e compreensão das especificações. A Tabela 23 apresenta a largura de banda de cada *link* virtual considerado, o valor da latência em cada *link* e os *datacenters* de origem e destino da comunicação de rede em tempo de simulação. Os valores são referentes às mesmas configurações definidas no código de instanciação dos *datacenters*. Com base nessas informações foi definida uma largura de banda padrão entre todos os *links* de rede como sendo no valor de 1Gbit/s. Já os tempos de latência foram padronizados com base na distância física e na taxa de colisão encontrada nos barramentos da rede do LASDPC bem como no *link* de conexão com a *internet* da USP. Assim sendo, as comunicações entre cada um dos *datacenters* e o *Workflow Scheduler* (entidade que representa o *Broker* da MDAPSP) foram definidas com latência de 100ms (milissegundos). As comunicações com a plataforma em nuvem computacional SoftLayer foram normalizadas com uma latência padrão de 500ms. As comunicações entre os *datacenters* que representam os *clusters* foram normalizados com uma latência padrão de 250ms.

6.3.3 Extensão do framework WorkflowSim

O ambiente simulado estende características do *WorkflowSim* para emular uma infraestrutura real e busca prover contribuições diretas na execução de *workflows* científicos. As contribuições da simulação são a possibilidade de efetuar os ensaios de experimentos com a carga de trabalho em ambiente compartilhado e efetuar uma distribuição otimizada do fluxo na

infraestrutura, por meio do ajuste da política de alocação dos recursos (tanto dos *hosts* quanto de VM). Conforme os requisitos dos *workflows*, permite ainda prover uma boa relação entre a alocação controlada de recursos, com desempenho adequado e a economia de gastos com locação de recursos computacionais de terceiros. A Figura 47 apresenta um diagrama de classes com uma visão geral da extensão das alterações efetuadas no *framework WorkflowSim* no intuito de atender a demanda de requisitos funcionais do ambiente real modelado, destacando as principais alterações, novas classes e heranças realizadas no contexto de programação.

Figura 47 – Diagrama de classes das extensões implementadas para o desenvolvimento da MDAPSP.

Fonte: Elaborada pelo autor.

Foram usadas classes utilitárias da API da arquitetura de Cloud BEQoS no programa principal de simulação descrito nesse capítulo. Já para as novas características implementadas pela arquitetura MDAPSP foram desenvolvidas outras 4 classes de objeto, sendo elas: APMD, *Simple Planning Algorithm*, e dois escalonadores descritos na próxima seção. A classe *Simple-PlanningAlgorithm* possui um algoritmo de planejamento simplificado para a carga de trabalho dos *workflows*. Tal algoritmo foi concebido para não efetuar nenhum tipo de ajuste ou agrupamento nas tarefas do *workflow*, permitindo avaliar isoladamente a influência dos algoritmos de escalonamento de tarefas incluídos no componente *WorkflowScheduler*.

6.4 Algoritmos de Escalonamento

Essa seção apresenta os dois algoritmos de escalonamento desenvolvidos para atuar no núcleo da MDAPSP, orquestrando o atendimento das requisições, o fluxo de dados e as análises de distribuição de acordo com políticas que busquem manter o funcionamento das transações de *workflows* científicos.

Os algoritmos descritos nessa seção contemplam o núcleo central da MDAPSP, como demonstrado na Figura 48, modelando o *Broker*, *Execution Module* e o *Decision Maker*, cujos detalhes são descritos no Capítulo 5. O *Broker* é definido na classe do *WorkflowSim* denominada

Workflow Scheduler, a qual foi adaptada e modificada conforme as padronizações e requisitos da arquitetura. Como o *Broker* é responsável pela orquestração do atendimento aos clientes, ele engloba entre outros mecanismos o módulo de execução, as políticas de balanceamento e escalonamento e as estratégias definidas pelo módulo de tomada de decisão. Esse módulo pode ter diversas políticas incluídas em seu funcionamento, entretanto foi determinada a construção de uma versão mais concisa, capaz de realizar as avaliações de distribuição de tarefas, capacidade de atendimento e funcionalidade geral da arquitetura, antes de propor mecanismos mais sofisticados para o núcleo da MDAPSP.

Figura 48 – Ilustração dos módulos centrais modelados em simulação.

Fonte: Elaborada pelo autor.

6.4.1 MDSA

Para o ambiente de simulação projetado foram desenvolvidos algoritmos de escalonamento específicos, visando atender à demanda de cargas de trabalho de *workflows*. O primeiro algoritmo é o MDSA. O referido algoritmo tem como objetivo avaliar parâmetros do *workflow* e então tomar a decisão de qual *datacenter* deve processar as tarefas usando uma sub-rotina de alternância cíclica do tipo *First Come First Served* (FCFS), a qual envia cada tarefa para uma VM do *datacenter* escolhido na referida ordem em que as tarefas são definidas no arquivo descrito DAX. Os referidos parâmetros considerados no processo decisório do escalonamento são “*Amin*” (quantidade de aminoácidos), “*Pop*” (tamanho de população) e “*Ger*” (número de gerações do preditor). Esses três valores são providos em uma *tag* XML adicionada ao arquivo DAX de nome *parameters*. O valor atribuído a cada um desses atributos é um número inteiro e as regras de escalonamento para verificar qual a melhor estrutura são apresentadas na Tabela 24. Essa comparação reflete o comportamento do *Decision Maker* e das avaliações realizadas previamente no Koala.

Em um caso realístico de experimentação, os valores são dinâmicos e podem variar de forma imprevisível, inclusive descrevendo parâmetros que não foram contemplados pelas regras do MDSA. Nesses casos, o algoritmo do escalonador seleciona as máquinas do tipo *cluster* para a execução do *workflow*, uma vez que elas representam a opção intermediária de recursos

Tabela 24 – Condições estáticas do escalonador MDSA.

Variação de Parâmetros	Datacenter
(amin <50 AND pop <50 AND ger <50)	DC_WorkStation
((amin >= 50 AND amin <150) AND pop <30 AND ger <20)	DC_WorkStation
(amin <50 AND (pop >= 50 AND pop <200) OR ger <150)	Cluster
((amin >= 50 AND amin <150) AND (pop >= 30 AND pop <80) AND ger <50)	Cluster
(amin >= 150 AND pop <40 AND ger <40)	Cluster
(amin <50 AND (pop >= 200 OR ger >= 150))	DC_SoftLayer
((amin >= 50 AND amin <150) AND (pop >= 80 OR ger >= 50))	DC_SoftLayer
(amin >= 150 AND pop >= 40 AND ger >= 40)	DC_SoftLayer

Fonte: Elaborada pelo autor.

Nota – O registro “Cluster” na coluna “Datacenter” abrange os três *clusters* do LASDPC.

disponíveis, sendo capazes de atender com qualidade a uma boa porcentagem dos eventos. Essa regra evita que ocorra alguma exceção no atendimento das requisições.

Assim, para se entender a execução de instruções de programa do algoritmo de escalonamento MDSA é importante observar a seguinte sequência de passos:

1. Ler os parâmetros do arquivo DAX (*Amin*, *Pop* e *Ger*);
2. Efetuar processo de escolha da infraestrutura;
3. Separar as VMs do *datacenter* escolhido para execução do *workflow*;
4. Atualizar a lista de VMs do simulador;
5. Atribuir cada tarefa do *workflow* conforme o número de VMs do *datacenter* escolhido usando uma sub-rotina do tipo FCFS;
6. Despachar as tarefas para execução nas VMs.

A proposta do MDSA visa superar os obstáculos comuns encontrados em algoritmos de escalonamento disponíveis em outros sistemas, como a falta de heurísticas e pré-avaliação. O MDSA promove a possibilidade de distribuição da carga de experimentos de PSP em capacidades de máquinas variadas, despachando o processamento para diferentes *datacenters* de acordo com a análise prévia dos requisitos do *workflow*, evitando a utilização de filas sequenciais. A utilização indiscriminada das máquinas disponíveis pode causar desperdício de recursos, longas esperas e resultados de predição ruins, como já discutido. A opção por regras estáticas visa uma previsibilidade no comportamento do escalonador e a preparação da infraestrutura para seu correto funcionamento perante as cargas de trabalho impostas. Ainda que conte com regras simples, projeta-se que tal mecanismo seja capaz de ordenar adequadamente a distribuição das tarefas de PSP no ambiente da arquitetura, identificando com boa precisão a necessidade de

recursos computacionais de determinado experimento, automatizando o processo de escolha do ambiente e indicando ao *Broker* o processo decisório.

6.4.2 WLSA

O segundo escalonador é o WLSA, também incluso no *Broker* da MDAPSP. O WLSA proporciona análises de capacidade diversas, estudos de cumprimento de prazo e a distribuição de *workflows* científicos em ambiente heterogêneo. Como o objetivo da MDAPSP é formar um ambiente viável para *gateways* científicos oferecidos em conjunto variado de máquinas, tais medições são fundamentais para a estruturação adequada dos mecanismos e sua validação técnica.

O WLSA calcula o tamanho da carga imposta por um determinado *workflow* e a converte para Milhões de Instruções (MI). Ao determinar esse parâmetro, o escalonador tem a capacidade de estimar qual o *datacenter* necessário para executar o *workflow*. Adicionalmente, é determinado um prazo (*deadline*) para a conclusão da execução do *workflow*, que o escalonador busca cumprir ao determinar o *datacenter*. O WLSA tem maior capacidade de análise da carga, escalando o número de máquinas e definindo o melhor ambiente disponível, de forma a respeitar o *deadline* e escolher o *datacenter* adequado, avaliando os conjuntos de recursos de forma crescente. As regras de alocação do WLSA permitem a combinação de mais de um *datacenter* na execução de *workflows* científicos, conforme as necessidades de cada caso. Dessa forma, quando o WLSA não encontra nenhum *datacenter* capaz de atender a requisição dentro do prazo estabelecido, ele aloca todas as infraestruturas em um único conjunto de recursos de forma a tentar cumprir o *deadline* e manter a qualidade de atendimento sem ocorrências de atrasos e sobrecargas. Uma vez que atualmente o modelo não contempla parâmetros de custo para as diferentes infraestruturas, a alocação é baseada exclusivamente em critérios de capacidade de processamento.

O WLSA calcula sobrecargas de tempo ao executar os *workflows*. Esse cálculo é fundamental para descrever de forma realística o comportamento das tarefas em ambiente computacional. Para isso, o escalonador avalia as *tags* no arquivo DAX que indicam procedimentos de leitura e escrita, os quais são causa de atrasos em experimentos reais. Dessa forma, tais valores são considerados e acrescentados ao tempo de processamento total do *workflow*. A somatória da sobrecarga é realizada antes da determinação do prazo de execução do *workflow*, momento em que o WLSA determina qual o ambiente computacional a ser selecionado.

Atualmente, o escalonador não contempla combinações das infraestruturas em pares ou mesmo grupos menores, porém isso pode ser desenvolvido posteriormente. Em avaliações futuras, podem ser realizadas análises mais detalhadas que contemplem critérios de custo, alterando as definições de escalabilidade de acordo com classes de atendimento. Assim será possível descrever cenários com diversas opções de custo-benefício, relacionando o tempo de resposta com o valor de investimento. Variações nas políticas de alocação são previstas, permitindo diversas práticas de acordo com decisões administrativas e de projeto. Esse escalonador é

também mais flexível que o MDSA ao fazer avaliações dinâmicas que correlacionam a carga de trabalho e os recursos computacionais disponíveis, enquanto o outro algoritmo tem regras fixadas baseadas em conhecimento prévio. É importante destacar que o WLSA também usa informações do *Decision Maker* para a construção da sua lógica de funcionamento e que ambos os escalonadores podem alcançar melhor desempenho conforme o modelo do SVR for alimentado e aprimorado. O MDSA entretanto é mais dependente da qualidade do modelo de predição do *Decision Maker* para descrever suas regras, que podem ter níveis de complexidade e de especificidade mais avançados conforme as informações sejam coletadas da infraestrutura.

6.5 Considerações finais

Este Capítulo apresentou o modelo de simulação criado no *WorkflowSim* para estruturação inicial da arquitetura MDAPSP. O *WorkflowSim* é um simulador especializado em *workflows* científicos, que fornece um ambiente completo para implementação e análises diversas. Os módulos principais da arquitetura conceitual são implementados nesse *framework*, incluindo dois escalonadores de tarefas especializados em *workflows* de PSP. O próximo capítulo apresenta a avaliação de desempenho dos módulos desenvolvidos.

AVALIAÇÃO EXPERIMENTAL

7.1 Considerações iniciais

Este Capítulo apresenta avaliação experimental dos módulos desenvolvidos em ambiente de simulação com o *framework WorkflowSim*. Os módulos são avaliados com carga sintética do *workflow* Montage disponível no *gateway* Pegasus e dois outros desenvolvidos para essa tese por meio dos experimentos realizados no *gateway* Koala, os quais representam experimentos de PSP.

7.2 Ambiente de Teste

A estrutura da MDAPSP descreve três tipos de ambiente de processamento, de acordo com as especificações requeridas: um do tipo *workstation*, um do tipo *cluster* e um do tipo *nuvem*. Para as máquinas do ambiente do *cluster* foram mapeadas as configurações de acordo com os *racks* presentes no LASDPC¹. Por esse motivo há uma subdivisão no ambiente de *cluster*, contando com três configurações distintas. No total, a arquitetura tem a possibilidade de realizar o escalonamento para cinco configurações computacionais diferentes. Ainda que as máquinas dos *clusters* sejam semelhantes, é importante detalhar suas diferenças para obter resultados realísticos, condizentes com o ambiente real, além de diferenciar os conjuntos de experimentos a ser processado em cada tipo de máquina, a fim de otimizar a utilização e auxiliar os usuários no processo de decisão.

Todas as configurações de máquina foram feitas com base em parâmetros disponíveis nas especificações dos fabricantes. Para a máquina do tipo *workstation* foi escolhida uma versão moderna, considerada um modelo base de entrada para o mapeamento dos recursos. A especificação pode ser vista na Tabela 25.

Para a configuração de *clusters* são mapeados conforme as três instalações do LASDPC,

¹ <http://infra.lasdpc.icmc.usp.br>

Tabela 25 – Especificações da máquina tipo *workstation*.

Dell Workstation	
Processador	Intel® Xeon® E3-1225 V5 (Quad Core, 3.3GHZ, 8MB Cache)
RAM	8GB, 2133 MHz, ECC UDIMM DDR4
HD	500 GB SATA 7.200 RPM 3,5 "
GPU	NVIDIA® QUADRO® K420 de 2GB
Preço (2018)	R\$ 6.497,00

Fonte: Dados da pesquisa.

Tabela 26 – Especificações dos nós do *Cluster Cosmos*

Cluster Cosmos	
Número de nós	19
Processador	Intel Core 2 Quad Processador Q9400 (6M Cache, 2.66 GHz, 1333 MHz FSB)
RAM	8 GB RAM DDR3 Kingston
HD	160GB Seagate Sata II 7200RPM
GPU	-
Placa-mãe	Gigabyte GA-Z97X-SLI ATX
Fonte de Alimentação	ATX 300W Real
Sistema Operacional	Linux Ubuntu 14.04.4 LTS

Fonte: Dados da pesquisa.

Tabela 27 – Especificações dos nós do *Cluster Andromeda*

Cluster Andromeda	
Número de nós	13
Processador	8 núcleos AMD Vishera 4.2 Ghz
RAM	32 GB RAM DDR3 Corsair Vegeance
Hard Drives	HD 2TB Seagate Sata III 7200RPM
GPU	Video Nvidia GTX 650 – 1GB
Placa-mãe	Gigabyte 970A-D3
Fonte de Alimentação	ATX 650W Real
Sistema Operacional	Linux Ubuntu 14.04.4 LTS

Fonte: Dados da pesquisa.

nomeadamente Cosmos, Andromeda e Halley, em ordem crescente de capacidade. As configurações do Cosmos são apresentadas nas Tabela 26. Os demais, já tratados na Seção 4, têm reproduzidas suas configurações nas Tabelas 27 e 28.

Finalmente, para a plataforma de computação em nuvem, há opções mais diversas, desde máquinas simples até estruturas de supercomputadores. Uma vez que a MDAPSP considera uma capacidade crescente de máquinas, essa infraestrutura é inspirada na solução Bare Metal, da empresa Soft Layer. Esta vende serviços sob demanda, bem como máquinas particulares com garantias de uso exclusivo. É possível definir desde os componentes básicos como discos rígidos,

Tabela 28 – Especificações dos nós do *Cluster Halley*

Cluster Halley	
Número de nós	13
Processador	Intel Core I7 Processor LGA -1150 – 4790 3.60GHZ DMI 5GT/S 8MB
RAM	32 GB RAM DDR3 Corsair Vegeance
Hard Drives	HD 2TB Seagate Sata III 7200RPM
GPU	Video Nvidia GTX 650 – 1GB
Motherboard	Gigabyte GA-Z97X-SLI ATX
Fonte de Alimentação	ATX 650W Real
Sistema Operacional	Linux Ubuntu 14.04.4 LTS

Fonte: Dados da pesquisa.

Tabela 29 – Especificações da Plataforma Bare Metal

Bare Metal		Preço (US\$)
Processador	4 processadores Quad Intel Xeon E7-4850 v2 (48 Cores, 2.30 GHz)	\$ 1.046,00
RAM	1 TB RAM	\$ 2.188,00
HD	24 HDs 8 TB Sata	\$ 2.136,00
Largura de Banda Pública	1000 GB	\$ 45,00
Porta Uplink	10 Gbps Público & Privado	\$ 100,00
Sistema Operacional	Ubuntu Linux 16.04 LTS Xenial Xerus (64 bit)	-
Opcionais	Firewalls, Monitoramento e Fonte de Alimentação	\$ 10,00
Armazenamento	EVault: 100GB EVault Disk to Disk Enterprise Backup	\$ 80,00
Total	Custo Mensal	\$ 5.605,00

Fonte: Dados da pesquisa.

memória e processadores, até ajustes finos como largura de banda, velocidade de *upload*, SO e *firewalls*, sendo que cada um desses recursos tem uma taxa proporcional. Na Tabela 29 as configurações escolhidas são apresentadas com os respectivos preços em dólares.

Todos os experimentos realizados utilizam as configurações de máquinas aqui descritas, representando instalações atualmente disponíveis do Koala e projeta a execução em ambientes variados, traçando o caminho para o estabelecimento de ofertas de recursos para *gateways* científicos diversos. Como descrito no Capítulo 6, a codificação da MDAPSP considera uma VM por nó, dessa forma é possível inferir a capacidade de cada uma conforme as informações descritas nas tabelas.

Conforme as práticas de simulação computacional, é preciso estimar variações nos parâmetros do sistema que representem a aleatoriedade do mundo real, de forma a evitar enviesamento no dados. Assim, foram definidos critérios de dados que variem dinamicamente no sistema da MDAPSP, tanto para o tamanho da carga imposta ao sistema quanto aos parâmetros

Tabela 30 – Parâmetros de cenários para carga de trabalho

Variabilidade de Carga					
Faixa	A	B	C	D	E
Escala	1	2	5	10	15

Fonte: Elaborada pelo autor.

Tabela 31 – Parâmetros dinâmicos do MDSA.

Parâmetro	Mínimo	Máximo
Aminoácidos	20	200
População	20	300
Gerações	15	200

de funcionamento dos dois escalonadores desenvolvidos. Para representar a variabilidade na carga de trabalho foram determinadas faixas de oscilação conforme a Tabela 30. O parâmetro “escala” é um fator multiplicador do parâmetro denominado “*runtime scale*” que determina o tamanho das tarefas em MI, utilizado para execução nas *cloudlets*. Esse fator é importante para tornar a representatividade do modelo de carga aproximada aos cenários encontrados em sistemas computacionais e será usado nos mesmos padrões para todos os algoritmos avaliados.

Para o MDSA foram escolhidos para variarem os parâmetros de PSP que esse escalonador analisa para tomar as decisões de distribuição das tarefas. Essa aleatoriedade é implementada em código pela função *Math.Random* que descreve uma distribuição uniforme dos valores ao fazer o sorteio. Para tais medidas foram definidas faixas de valores mínimos e máximos, de forma a representar com boa aproximação os ambientes físicos, também respeitando a distribuição do CASP 2016 e os experimentos realizados no Koala. Esses valores são informados na Tabela 31.

Para o escalonador WLSA o parâmetro variável é o *deadline* de execução. Esse valor é calculado conforme o tamanho da carga do *workflow* e comparado com a capacidade dos *data-centers* disponíveis, escalonando para aquele que consiga cumprir o prazo determinado. A faixa de variação do *deadline* calculado é de 50% a 100% do tempo estimado pelo escalonador, de forma a estressar o ambiente e exigir boa capacidade de máquina da arquitetura. Esse parâmetro também varia seguindo uma distribuição uniforme da função supracitada. O WLSA também calcula um parâmetro de sobrecarga nas tarefas, estimado pelo simulador, baseado nos comandos de leitura e escrita conforme os arquivos dos *workflows* científicos. Essa sobrecarga é adicionada ao *makespan* do *workflow* para descrever com maior grau de precisão o comportamento do experimento. Dessa forma, o modelo consegue representar adequadamente os *overheads* observados em ambiente real.

Tabela 32 – Desenho de experimentos da arquitetura

Fator	Nível 1	Nível 2	Nível 3
Escalonador	Round Robin	MDSA	WLSA
Workflow	1VII	1AGY	Montage 25
Infraestrutura	Apenas <i>Cluster</i>	Sem nuvem	Com nuvem

Fonte: Elaborada pelo autor.

7.3 Desenho de Experimentos

O desenho de experimentos para avaliação dos algoritmos desenvolvidos segue os padrões de avaliação de desempenho de sistemas computacionais (JAIN, 1991). Foram definidas combinações de três fatores e três níveis conforme descrito na Tabela 32. Os escalonadores MDSA e WLSA são comparados entre si e com um escalonador padrão de algoritmo *Round Robin*, já disponível no *WorkflowSim*, frequentemente utilizado como parâmetro de funcionamento de sistemas. Para a carga de trabalho são utilizados *workflows* sintéticos de três tipos: um denominado Montage 25, disponível no portal Pegasus², já bem estabelecido como *benchmark* e análise de desempenho. O Montage representa aplicações que compõe imagens complexas do universo a partir de quadros menores. Os demais *workflows* determinam as tarefas envolvidas em processamentos de PSP, utilizando informações de proteínas disponíveis no PDB, tendo sido escolhidas a 1VII de 36 aminoácidos e a 1AGY de 200 aminoácidos. Esses *workflows* foram definidos a partir de experimentos executados no Koala, como demonstrado na na Seção 4.2.3, de forma a refletir com precisão o ambiente avaliado. Com relação à infraestrutura de recursos, tendo todos os *datacenters* disponíveis, foi decidido variar em três cenários de avaliação: o primeiro descreve apenas os *clusters*, o segundo que soma a *workstation* aos *clusters* e um terceiro que inclui na arquitetura os recursos de nuvem. Dessa forma é possível estabelecer uma diversidade de capacidades de atendimento e identificar a possibilidade de execução local em máquinas individuais de alguns *workflows* científicos. A variável de saída é o tempo de resposta, para todos os casos. A combinação dos fatores para os níveis considerados é exibida na Tabela 33. São realizadas trinta repetições para cada um dos experimentos, com o objetivo de obter uma média na variável de resposta que tenha representatividade estatística.

Os cenários de avaliação são definidos de forma a comparar os três escalonadores considerados entre si, em todas as variações disponíveis. Dessa forma são alterados os *workflows* e variado o tipo de ambiente de processamento, conforme descrito na Tabela 34.

7.4 Resultados

Com o sistema funcional e os parâmetros definidos, foram executados os experimentos para analisar a capacidade da arquitetura de distribuir a carga de trabalho adequadamente e

² <https://confluence.pegasus.isi.edu/display/pegasus/WorkflowGenerator>

Tabela 33 – Combinação de Fatores e Níveis dos Experimentos

Experimento	Escalonador	Workflow	Recursos
1	Round Robin	MONTAGE	<i>Cluster</i>
2	Round Robin	MONTAGE	Sem Nuvem
3	Round Robin	MONTAGE	Com Nuvem
4	Round Robin	1VII	<i>Cluster</i>
5	Round Robin	1VII	Sem Nuvem
6	Round Robin	1VII	Com Nuvem
7	Round Robin	1AGY	<i>Cluster</i>
8	Round Robin	1AGY	Sem Nuvem
9	Round Robin	1AGY	Com Nuvem
10	MDSA	MONTAGE	<i>Cluster</i>
11	MDSA	MONTAGE	Sem Nuvem
12	MDSA	MONTAGE	Com Nuvem
13	MDSA	1VII	<i>Cluster</i>
14	MDSA	1VII	Sem Nuvem
15	MDSA	1VII	Com Nuvem
16	MDSA	1AGY	<i>Cluster</i>
17	MDSA	1AGY	Sem Nuvem
18	MDSA	1AGY	Com Nuvem
19	WLSA	MONTAGE	<i>Cluster</i>
20	WLSA	MONTAGE	Sem Nuvem
21	WLSA	MONTAGE	Com Nuvem
22	WLSA	1VII	<i>Cluster</i>
23	WLSA	1VII	Sem Nuvem
24	WLSA	1VII	Com Nuvem
25	WLSA	1AGY	<i>Cluster</i>
26	WLSA	1AGY	Sem Nuvem
27	WLSA	1AGY	Com Nuvem

Fonte: Elaborada pelo autor.

Tabela 34 – Cenários de avaliação.

Cenário	Escalonador	Workflow	Infraestrutura
1	Todos	1VII	Apenas <i>Cluster</i>
2	Todos	1VII	Sem nuvem
3	Todos	1VII	Com nuvem
4	Todos	1AGY	Apenas <i>Cluster</i>
5	Todos	1AGY	Sem nuvem
6	Todos	1AGY	Com nuvem
7	Todos	MON	Apenas <i>Cluster</i>
8	Todos	MON	Sem nuvem
9	Todos	MON	Com nuvem

Fonte: Elaborada pelo autor.

medir os tempos de resposta. A Figura 49 mostra uma comparação direta de todos os cenários considerados, em que podem ser observadas as diferenças em termos de carga que os *workflows* geram para o ambiente de processamento. Também fornece uma visão geral de como cada um dos escalonadores se comporta para as diversas cargas e configurações de ambiente. O *Round Robin* é utilizado como ponto de comparação por ser um algoritmo clássico e testado sob diversas condições. Como se pode notar, em geral o desempenho dele foi superado pelos escalonadores MDSA e WLSA, desenvolvidos para a arquitetura. Em parte das variações o MDSA apresenta um empate estatístico com o *Round Robin*, devido ao seu mecanismo interno semelhante, após a fase decisória. Entretanto, na maioria dos casos o MDSA conseguiu otimizar o desempenho, demonstrando boa capacidade. Já o algoritmo WLSA destaca-se ao apresentar o melhor comportamento em todos os casos. Esses algoritmos são fundamentais para o funcionamento da arquitetura MDAPSP, motivo pelo qual devem alcançar bons níveis de desempenho. Pode-se notar que um algoritmo como o *Round Robin* pode prejudicar a utilização do ambiente heterogêneo de máquinas, uma vez que nenhuma análise é realizada por parte desse escalonador, o que resulta tanto em sobrecargas de máquinas de menor capacidade quanto em subutilização de recursos de maior potência computacional. Os cenários são avaliados separadamente a seguir, sendo agrupados pelo tipo de *workflow*.

Figura 49 – Comparação de todos os cenários avaliados.

Fonte: Elaborada pelo autor.

Os três primeiros cenários contemplam a carga de trabalho do *workflow* da proteína 1VII, variando a infraestrutura de computação e são observados na Figura 50. O primeiro deles considera apenas os *clusters* e os resultados mostram que os escalonadores *Round Robin* e MDSA empataram estatisticamente nos tempos de resposta obtidos, ainda que as médias tenham uma diferença relativa. Isso ocorre porque o MDSA tem poucas opções para análise da carga conforme suas regras estáticas e dessa forma ele se comporta como algoritmo de alternância. O WLSA teve resultados melhores para esse caso, já que o parâmetro de *deadline* implementado melhora a utilização de recursos e força a seleção de melhores máquinas.

No cenário 2 o processamento é realizado na infraestrutura dos *clusters* e da *workstation*. Mais uma vez os escalonadores *Round Robin* e MDSA apresentam resultados semelhantes, configurando um empate estatístico. Ambos variam na mesma faixa do cenário anterior, indicando que as máquinas da *workstation* foram pouco acionadas. O WLSA por sua vez destaca-se ao reduzir consideravelmente o tempo de resposta, já que ele tem capacidade de utilizar as VMs proporcionalmente. Ele também manteve a variação semelhante ao cenário anterior, pelas mesmas razões.

No cenário 3 os escalonadores são avaliados com todos os *datacenters* da MDAPSP. Os resultados para esse caso são melhores dos que os cenários anteriores. O escalonador *Round Robin* não melhora em seu desempenho, se mantendo dentro da variação observada anteriormente. O MDSA apresenta tempos de resposta melhores nesse cenário, escolhendo máquinas de melhor capacidade, conforme as variações dos parâmetros indicaram necessidades de maior potência computacional. O WLSA novamente alcança o menor tempo de resposta, com boa diferença para os demais escalonadores, já que com as máquinas da nuvem disponíveis ele pode cumprir os *deadlines* estabelecidos com bastante folga. Esses cenários são um exemplo de caso no qual as máquinas do *cluster* são suficientes para o processamento em boa parte das repetições, sendo a nuvem acionada poucas vezes. Tal resultado justifica a opção pela variação do ambiente computacional e pode ser utilizado para justificar o investimento em máquinas locais. Sendo a 1VII uma proteína usada em diversos experimentos de PSP, os resultados desses cenários demonstram que a MDAPSP pode atuar com precisão em casos semelhantes.

Figura 50 – Resultados para o *workflow* 1VII, em três cenários de avaliação.

Fonte: Elaborada pelo autor.

Os três próximos cenários utilizam como carga de trabalho a proteína 1AGY, devidamente

mapeada em um *workflow* científico. Os cenários variam novamente conforme a infraestrutura disponível aos escalonadores de tarefas. Como pode ser visto na Figura 51, esse *workflow* impõe uma carga bem mais elevada aos sistemas, representando os tempos de execução observados em ambiente real e a variação de *makespan* que os experimentos de PSP podem apresentar conforme mudam os parâmetros. No 4º cenário, apenas os *clusters* são utilizados para processamento. Como se observa no primeiro terço da Figura, novamente os escalonadores *Round Robin* e MDSA têm comportamentos idênticos. Isso ocorre porque uma vez selecionadas as máquinas pelo MDSA, seu mecanismo interno se comporta como uma alternância cíclica, e, devido ao conjunto reduzido de opções, coincide com o algoritmo usado como *benchmark*. Também para esse caso, ainda que a média tenha diferença entre eles, o desvio padrão impede que se afirme qual é o superior. O WLSA consegue reduzir levemente o tempo de resposta, configurando uma diferença estatística clara para os demais. Como essa é uma proteína grande, conforme os parâmetros do usuário podem ser necessárias máquinas de maior potência, e esse escalonador escolhe os *clusters* mais potentes na maioria das vezes.

O cenário 5 utiliza o *workflow* da proteína 1AGY para teste da infraestrutura sem o *datacenter* em nuvem. Nesse caso, os escalonadores da arquitetura destacam-se dos resultados obtidos pelo *Round Robin* com boa margem. O MDSA mantém o resultado obtido no cenário anterior, demonstrando coerência de comportamento e apresentando vantagens com relação ao *Round Robin* que em sua alternância define máquinas de pior capacidade. Isso prova que mesmo que as regras do MDSA ofereçam pouca flexibilidade, esse mecanismo mantém regras de qualidade mínimas. O WLSA apresenta os melhores resultados, reduzindo significativamente o tempo total, também com desempenho comparável ao do cenário anterior, indicando não ter utilizado máquinas de menor capacidade.

No cenário 6, todos os *datacenters* são disponibilizados para execução dos experimentos. Nessa variação, o escalonador *Round Robin* não apresenta diferença para o cenário anterior, demonstrando limitação na distribuição de tarefas. O MDSA apresenta um tempo de resposta satisfatório, porém também não apresenta redução para a variação anterior, ficando dentro da margem estatística. As variações na escala de carga fizeram com que a seleção de VMs fosse semelhante e o algoritmo selecionou as máquinas da nuvem em poucas ocasiões. Já o WLSA demonstra mais uma vez sua capacidade de distribuição de tarefas, obtendo os melhores resultados nesse cenário, ao escolher adequadamente as VMs da nuvem para execução desse *workflow*, na maioria das variações. Mesmo com o cumprimento de *deadlines* e adicionando as sobrecargas de execução, a utilização de VMs adequadas é demonstrada pela redução no *makespan*.

No três últimos cenários, os escalonadores são avaliados com a carga de trabalho do *workflow* Montage e os resultados são observados na Figura 52. Para esses cenários, observa-se um tempo de resposta bem menor do que os dois *workflows* de proteínas. Ainda que o Montage 25 seja comparável a eles em número de tarefas, essas são bem mais simples e executadas

Figura 51 – Resultados para o *workflow* 1AGY, em três cenários de avaliação.

Fonte: Elaborada pelo autor.

em poucos segundos. Nesses três últimos cenários, o *datacenter* da *workstation* tem melhor desempenho e é capaz de atender bem a algumas das requisições. No cenário 7, apenas com *clusters* como plataforma, pode-se observar que os escalonadores *Round Robin* e MDSA têm comportamentos estatisticamente idênticos com relação ao tempo de resposta. Isso é explicado porque a menor disponibilidade de *datacenters* e as poucas tarefas do Montage oferecem menos pontos de otimização, apresentando um empate técnico. O WLSA consegue reduzir o tempo de resposta nesse caso, distribuindo adequadamente as tarefas entre as VMs disponibilizadas, ao definir o *deadline* e escalonar os melhores recursos.

No cenário 8 utiliza-se a infraestrutura de recursos desconsiderando-se apenas a nuvem. Nesse cenário o MDSA consegue melhorar o seu desempenho e utilizar adequadamente as VMs disponíveis, distanciando-se do *Round Robin*, que manteve comportamento semelhante ao do cenário anterior. O WLSA reduz o tempo de resposta da execução do *workflow*, distribuindo as tarefas adequadamente.

No cenário 9 o *workflow* Montage é executado com a infraestrutura de recursos completa. Nessa avaliação, os resultados para os três escalonadores são estatisticamente semelhantes aos do cenário 8. Isso ocorre porque em nenhum dos casos o *datacenter* da nuvem foi utilizado de forma significativa, mantendo-se as regras de negócio inalteradas. Basicamente, esse *workflow* impõe uma carga menor e apenas em combinações específicas as melhores VMs seriam necessárias, não apresentando uma influência considerável.

Os resultados comparados dos cenários com menos recursos para os cenários que abran-

Figura 52 – Resultados para o *workflow* Montage em três cenários de avaliação.

Fonte: Elaborada pelo autor.

gem todos os *datacenters* podem parecer triviais do ponto de vista de comparação dos tempos de resposta obtidos pelos escalonadores. Entretanto, o objetivo dessa variação é analisar o comportamento da carga em ambientes com capacidades distintas. Observa-se que máquinas de diferentes configurações podem atender com qualidade as variações encontradas nos *workflows* científicos. Mesmo com alguns casos de empate estatístico, com o MDSA nota-se que a carga de trabalho é melhor distribuída entre os ambientes disponíveis do que com o *Round Robin*, utilizando corretamente as máquinas para executar as tarefas. Isso é apenas uma indicação da economia que essa solução pode fornecer, agendando tarefas para computadores menores e mais baratos. Inclusive, com o mapeamento de outros *workflows* de proteína e a melhoria do mecanismo do *Decision Maker* a tendência é que essa diferença se acentue. O mecanismo principal da MDAPSP pode ser ofertado para os pesquisadores em PSP para auxiliar na decisão de construção de seus ambientes de recursos computacionais, além de fornecer dados para escalonamento eficiente dos experimentos. A adoção dos padrões que a arquitetura estabelece mostra capacidade de aumentar o desempenho, reduzindo o tempo necessário para o processamento de um *workflow*.

Em uma outra avaliação, é apresentada a taxa de utilização dos recursos computacionais da arquitetura MDAPSP. O objetivo dessa observação é mostrar o comportamento dos escalonadores nas múltiplas infraestruturas e como as variações afetam seu funcionamento e o tempo de resposta. Para essa análise é definida uma amostra dos diversos experimentos realizados que se incluem nos cenários 4, 5 e 6 da análise anterior, nos quais a carga de trabalho é definida pelo *workflow* da proteína 1AGY.

Na Figura 53 é apresentada a taxa de utilização de uma amostra do cenário 4, no qual os escalonadores são executados tendo como infraestrutura os *clusters*. Nessa configuração, os escalonadores *Round Robin* e MDSA definiram as mesmas máquinas para processamento, apresentando comportamento praticamente idêntico. A utilização das VMs apresenta uma linearidade, porém os pontos de sincronização dos *workflows* acabam por incluir atrasos na execução. O WLSA por sua vez selecionou o *cluster* Halley, de maior capacidade, utilizando uma porcentagem maior da capacidade de máquina, porém operando por menor tempo. Nas VMs desse *datacenter* as execuções são completadas em menos tempo, abrindo espaço para a alocação de outros serviços, se necessário

Figura 53 – Taxa de utilização das VMs, com 3 *datacenters*.

Fonte: Elaborada pelo autor.

No cenário 5 no qual o *datacenter workstation* é incluso na plataforma, percebe-se que houve uma piora no funcionamento do *Round Robin* que escalonou tarefas para a máquina de menor capacidade, causando sobrecargas e afetando o tempo de resposta, como exibido na Figura 54. O MDSA utiliza-se basicamente do *cluster* Cosmos na maior parte do tempo dessa amostra, sendo capaz de reduzir o tempo de processamento. Isso mostra que esse escalonador é capaz de indicar adequadamente o ambiente de *cluster* para uma execução, porém não define o melhor dos recursos disponíveis em algumas das execuções. E o WLSA aciona o *cluster* Halley, de maior capacidade, o que mostra uma redução considerável no *makespan* e utilização adequada das VMs.

Já para o cenário 6, com a adição do *datacenter* em nuvem aos recursos, percebe-se uma diferença importante no comportamento do escalonador WLSA que opta por executar as requisições nas máquinas de maior capacidade, proporcionando redução significativa nos tempos de respostas. Os demais escalonadores mantiveram seu comportamento inalterado para a amostra.

Figura 54 – Taxa de utilização das VMs, com 4 *datacenters*.

Fonte: Elaborada pelo autor.

Ainda que a taxa de utilização da VM seja menor, o tempo de resposta é reduzido devido à sua alta capacidade. O escalonador MDSA também escalona máquinas para a nuvem se as regras estáticas do mesmo assim definirem, porém a distribuição aleatória da carga de trabalho indicou a necessidade de *cluster* nesse caso. Tais dados são apresentados na Figura 55.

Figura 55 – Taxa de utilização das VMs, com 5 *datacenters*.

Fonte: Elaborada pelo autor.

O *Round Robin*, com acesso a todos os recursos até pode selecionar boas máquinas para a execução dos *workflows* em algumas situações, entretanto observa-se uma taxa de utilização

muito ruim por parte desse escalonador, ocasionado desperdício de recursos. Por outro lado, os dois escalonadores da MDAPSP apresentam melhores níveis nesse quesito, otimizando a seleção e ocupação das VMs, o que reduz desperdícios e libera máquinas para uso distribuído. Avaliações semelhantes a essa serão utilizadas para a evolução dos algoritmos da arquitetura, buscando evoluir a utilização de recursos.

Uma análise de comparação pode ser feita com base na coleta de dados, relacionando as tarefas geradas no simulador *WorkflowSim* com o desenho dos *workflows* científicos considerados para a carga de trabalho. A criação das tarefas de forma paralela indica o comportamento do sistema e influencia a atuação dos escalonadores. Mais uma vez são coletadas amostras do experimentos realizados, para representar o funcionamento interno da arquitetura. Apenas para efeito de observação, as tarefas são abstraídas, sendo representadas por letras. Na Figura 56 são comparadas as tarefas geradas para o *workflow* de proteína 1VII com o seu desenho estrutural, definido na seção 4.2.3. Como pode ser observado no gráfico, as tarefas correspondem aos pontos de divisão e união do grafo, indicando que o comportamento dos mecanismos de gerenciamento segue a lógica definida pelo modelo.

Figura 56 – Tarefas do *workflow* relacionadas à sua estrutura.

Fonte: Elaborada pelo autor.

A Figura 57 compara o *workflow* 1AGY com as tarefas geradas em tempo de simulação. Também para esse caso, nota-se que o modelo definido no DAX é corretamente representado no ambiente de processamento, de forma que os mecanismos da MDAPSP sejam capazes de ordenar a sua criação, sincronização e término. Esse *workflow* representa a maior proteína considerada na avaliação, tendo um número maior de tarefas e maior complexidade nos cálculos de seu processamento. Os dois *workflows* de proteínas geram tarefas pesadas, que processam por períodos longos. Nesses casos, a sincronização é limitada pela tarefa mais lenta, indicando possíveis pontos de estudo que visem aumento de desempenho nesses pontos.

Finalmente, a Figura 58 ilustra o comportamento das tarefas para o *workflow* Montage 25. Esse tem diversas tarefas paralelas logo no seu início, com pontos de sincronização e tarefas

Figura 57 – Tarefas do *workflow* relacionadas à sua estrutura.

Fonte: Elaborada pelo autor.

sequenciais na última parte. Como se pode observar no gráfico de tarefas, a criação das mesmas respeita essa distribuição. Vale ressaltar que esse *workflow* é disponibilizado em outras versões que podem conter até mil tarefas, elevando o tempo de processamento e a complexidade do arquivo DAX. Essa versão com 25 tarefas foi selecionada por se aproximar do número de tarefas dos *workflows* de predição das proteínas selecionadas para a modelagem.

Figura 58 – Tarefas do *workflow* relacionadas à sua estrutura.

Fonte: Elaborada pelo autor.

Na Figura 59 o comportamento dos *workflows* é sobreposto para efeito de comparação. Nota-se que a estrutura das tarefas dos três é semelhante, ainda que a carga imposta pelas tarefas seja diferente. Por exemplo, o Montage 25 tem mais tarefas que 1VII, no entanto, elas são menores e o tempo de processamento é mais curto. A semelhança nas primeiras fases dos dois *workflows* de proteínas faz com que as linhas se sobreponham, entretanto na sequência da avaliação eles descrevem comportamentos diferenciados. Para a proteína 1AGY são definidos

um número maior de tarefas, mais partes paralelas e tempo mais longo de processamento, como se observa no gráfico. Importante ressaltar que nessa representação os tipos e tamanhos das tarefas são abstraídos, sendo utilizada para compreensão do desempenho dos *workflows* de forma geral.

Figura 59 – Comparação sobreposta dos *Workflows*

Fonte: Elaborada pelo autor.

Essa avaliação demonstra que o ambiente da MDAPSP é capaz de lidar com cargas e infraestruturas variadas corretamente e que o modelo simulado representa coerentemente o funcionamento da arquitetura. A proposta de heterogeneidade de recursos demonstra-se acertada, já que *gateways* científicos podem ser incapazes de executar alguns experimentos conforme as necessidades de cada caso. Também podem ocorrer desperdícios de recursos e encarecimento da manutenção dos serviços. Há que se considerar que, atualmente, a MDAPSP não contempla modelos de custo em seus mecanismos, favorecendo a utilização de máquinas de maior potência visando a redução do tempo de resposta. Uma vez que a redução de custo e a redução do *makespan* de um *workflow* são tarefas conflitantes, fazem-se necessários estudos que contemplem algoritmos de otimização multi-objetivo. Tais algoritmos deverão ser incluídos na arquitetura em trabalhos futuros, para que aproximações de custo-benefício possam ser realizadas de forma a atender necessidades variadas dos clientes.

Os dados de proveniência são disponibilizados pelo simulador na saída da execução. O armazenamento dessas informações é realizado pelo usuário o qual deve disponibilizar os mesmos para fins de reprodutibilidade. Em futuras versões, a MDAPSP deverá contar com um banco de dados onde tais informações sejam disponibilizadas.

7.5 Considerações finais

Este Capítulo apresentou o desenho de experimentos, os detalhes do ambiente de avaliação e os testes com os algoritmos de escalonamento desenvolvidos para a MDAPSP. A avaliação teve por objetivo analisar o funcionamento dos mecanismos de forma geral, assim como estressar o ambiente para avaliar a arquitetura em situações chave. Em linhas gerais, a avaliação mostra as vantagens do modelo distribuído na execução de *workflows científicos* e os ganhos trazidos por uma arquitetura modular. Mostra também a necessidade por um conjunto heterogêneo de máquinas como o proposto, devido às prováveis variações de carga observadas em ambientes científicos. Ambos os escalonadores desenvolvidos para o *Broker* da arquitetura destacam-se como soluções eficientes e apontam para o sucesso da proposta da MDAPSP.

O algoritmo MDSA apresenta avanços na distribuição das tarefas seguindo regras pré-estabelecidas. Com auxílio da tecnologia do SVR do algoritmo do *Decision Maker* cria uma lista de escalonamento baseada nas faixas de valores dos *workflows científicos* e define os *datacenters* mais indicados para cada caso. Apesar de implementar uma política relativamente simples, o MDSA aperfeiçoa a organização das tarefas em ambiente distribuído, apontando para melhorias no estado da arte em *gateways científicos*. O problema prático encontrado em outras políticas de escalonamento é o uso de VMs de todos os *datacenters* disponíveis, causando alocação desnecessária dos recursos para executar um *workflow* pequeno ou relativamente simples, o qual necessitaria de máquinas mais simples. A diferenciação dos *workflows* pela avaliação dos atributos de PSP e o direcionamento das tarefas para o *datacenter* mais adequado permite projetar evoluções para o ambiente real, tais como aumento da disponibilidade da infraestrutura, economia de energia (possibilidade de deixar *datacenters* em *stand-by*) e economia com custo de serviços de nuvem. O tempo médio de resposta dos *workflows* obtido com o MDSA é considerado satisfatório, pois escolhe o *datacenter* de destino das tarefas conforme o valor proporcional ao tamanho da carga.

Por sua vez o WLSA aprofundou ainda mais os ganhos de desempenho na arquitetura MDAPSP. O escalonador infere o melhor ambiente de processamento dos *workflows*. A inclusão da sobrecarga de entrada e saída detalha o comportamento do ambiente e reflete experimentos do mundo real. As taxas de utilização das VMs são altas, reduzindo potenciais desperdícios de recursos. O algoritmo WLSA foi o que apresentou os melhores resultados com relação ao tempo de processamento. A análise de tamanho da carga e o cumprimento dos prazos se mostram uma técnica eficiente e com qualidade. Considerando as sobrecargas na manipulação de arquivos das tarefas, esse escalonador é capaz de reduzir significativamente o tempo de resposta dos *workflows* e promover excelentes taxas de ocupação dos recursos. O WLSA destaca-se como o mecanismo mais indicado para integrar o *Broker* da arquitetura para a maioria dos casos, despontando como um algoritmo eficiente e inovador. A evolução da arquitetura contará com versões otimizadas do mesmo, mantendo sua alta capacidade de atuação.

Adicionalmente, a MDAPSP se mostra competente no processamento de *workflows* para

PSP, seu objetivo inicial. Tamanhos de proteínas recorrentes nos experimentos de *folding* foram processados com sucesso e em diversos tipos de infraestrutura. A redução do tempo de resposta é um indicativo importante para o estabelecimento da arquitetura. Por fim, a possibilidade de processar *workflows* de outros domínios da ciência preconiza sua adequação para atuar em *gateways* científicos e campos genéricos.

CONCLUSÃO

8.1 Considerações iniciais

Este Capítulo apresenta as conclusões deste projeto de doutorado, os resultados alcançados, contribuições e os próximos passos. A MDAPSP representa uma forma inovadora de uso de arquiteturas orientadas a serviço no contexto de *e-science*, fornecendo ferramentas de forma integrada e colaborativa para auxílio às pesquisas, mais especificamente em PSP.

8.2 Argumentação geral

A crescente complexidade dos experimentos científicos *in silico* e a necessidade de colaboração ativa entre diversos campos de conhecimento aumentam a exigência por soluções interdisciplinares. As pesquisas em PSP demandam alta capacidade computacional, recursos especializados e métodos avançados de experimentação e análise. Por outro lado, o conhecimento de um cientista na área de computação não deve ser um obstáculo para a utilização de sistemas avançados de pesquisa. Nesse aspecto, os *gateways* científicos têm oferecido recursos de forma transparente, com padronização de métodos e formulações para execução de testes e análises. Aliado à estrutura dos *workflows* científicos, permitem a reprodução de experimentos, reutilização de dados e compartilhamento de informações. No entanto, esses *gateways* podem sofrer com problemas de desempenho que impactam nos resultados e na evolução das pesquisas.

Esta tese apresentou a MDAPSP, uma arquitetura modular para estruturação otimizada de *gateways* científicos. Essa arquitetura é baseada nos conceitos gerais de SOA para fornecer integração, disponibilidade e fraco acoplamento. A flexibilidade permitida por sua estrutura tem potencial para atuar com eficiência em meios interdisciplinares. Apresentada em forma conceitual, os módulos são propostos fornecendo a base para sua futura implementação. Ressalta-se que essa tese não propõe um novo *gateway* científico ou um SWMS original, mas um modelo

de constituição dos *gateways* de forma a aumentar sua compatibilidade e usabilidade. Tampouco intervém nos procedimentos específicos de PSP, mas usa tais experimentos como carga de trabalho e analisa potenciais ganhos no processamento desses experimentos na arquitetura.

Uma análise detalhada do Galaxy Koala foi realizada com objetivo de compreender seu funcionamento e apontar lacunas que afetam o seu desempenho. Isso permitiu determinar as principais necessidades do *gateway* e a modelagem da carga de trabalho em um *workflow* científico, proporcionando estudos detalhados desses dados, configurando uma novidade no campo. Identificado como um dos problemas a serem resolvidos, o provisionamento de recursos é um dos focos da MDAPSP, a qual contempla um ambiente heterogêneo de computação, integrando máquinas *workstation*, *clusters* e servidores em nuvem. Para otimizar a utilização de tais recursos, foi desenvolvido o *Decision Maker*, um módulo com algoritmo de aprendizado de máquina do tipo SVR que analisa um histórico de execuções e indica conforme a distribuição dos dados o melhor ambiente computacional para execução de um experimento.

Além disso, a MDAPSP é avaliada em ambiente simulado, que permite a análise de diversos aspectos técnicos, bem como o projeto detalhado de seu funcionamento. Os algoritmos desenvolvidos descrevem a estrutura fundamental da arquitetura e a estruturação dos recursos em múltiplos *datacenters*. No simulador são apresentados dois algoritmos originais de escalonamento de tarefas de *workflows* científicos que alcançam bons resultados na distribuição de carga, cumprimento de prazos e taxas de ocupação de recursos. Comparados a algoritmos clássicos utilizados como *benchmarking*, os escalonadores MDSA e WLSA apresentam resultados superiores em redução do tempo de processamento, demonstrando sua eficiência.

Esses algoritmos, aliados ao *Decision Maker* representam a estrutura fundamental da arquitetura, estabelecendo uma base sólida para o futuro protótipo. A fundamentação da mesma pode ser utilizada para outros *gateways* cujas necessidades se assemelham às do Galaxy, ampliando o alcance dessa pesquisa. A prova de conceito da MDAPSP é convincente, indicando o sucesso do seu projeto e seu potencial de impacto tanto em *e-science* quanto em aspectos específicos de pesquisas em PSP.

A ideia do *Decision Maker* bem como o início do seu desenvolvimento foram concebidos no período do doutorado-sanduíche, na Universidade de Leicester - Reino Unido, sob orientação do professor Stephan Reiff-Marganiec. Esse período foi financiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPQ), dentro do programa Ciência sem Fronteiras.

8.3 Contribuições

As contribuições principais dessa tese são:

- Arquitetura modular que estabelece as bases para um futuro protótipo, indicando os componentes fundamentais, justificando sua construção e projetando sua capacidade.

- Módulo *Decision Maker*, que avalia o histórico de execuções e é capaz de indicar o ambiente computacional mais indicado para um experimento em PSP, usando a tecnologia de SVR.
- Algoritmos de escalonamento MDSA e WLSA projetados em simulador, que avançam os conceitos disponíveis atualmente e apontam o acerto na distribuição de recursos em ambiente heterogêneo.

Entre outras contribuições complementares pode-se citar:

- Evolução dos algoritmos do *WorkflowSim*, com relação à implementação de múltiplos *datacenter* e escalonamento de tarefas.
- Estudo de manutenção do *gateway* Galaxy, com suas necessidades principais e comprometimento de capacidade.
- Modelagem de carga de experimentos em PSP em arquivos do tipo DAX, utilizado em *workflows* científicos, permitindo estudos diversos.

8.4 Publicações

Durante o período do doutorado foram publicados os seguintes artigos em conferência:

- OLIVEIRA, E. M.; ESTRELLA, J. C. ; REIFF-MARGANIEC, S. **A Distributed Environment Decision Maker Based on Machine Learning Techniques**. Em: 2017 IEEE Symposium on Service Oriented System Engineering (SOSE), São Francisco - EUA.
Qualis: B1
- OLIVEIRA, E. M.; ESTRELLA, J. C. ; COSTA, F. G. ; DELBEM, A. C. B. ; REIFF-MARGANIEC, S. **Optimising Scientific Workflow Execution using Desktops, Clusters and Clouds**. Em: International Symposium on Computing and Networking (CANDAR), 2017, Aomori - Japão.
Qualis: B2

Foi aceito para publicação em periódico o artigo abaixo:

- OLIVEIRA, E. M.; ESTRELLA, J. C.; DELBEM, A. C. B. ; NUNES, L. H.; SHISHIDO, H. Y.; REIFF-MARGANIEC, S. **Selection of computational environments for PSP processing on scientific gateways**. Heliyon Journal.
Cite Score: 0.62. JSR: 0.173

8.5 Dificuldades

A natureza interdisciplinar do projeto aumenta consideravelmente os desafios da sua realização. É preciso compreender os métodos, práticas, necessidades e estado da arte de um campo fora do domínio atual do laboratório em que esta tese foi desenvolvida. O levantamento bibliográfico e estudo de trabalhos foram longos, devido aos muitos pontos de investigação. O mapeamento do Galaxy Koala levou diversos meses, que envolveram compreensão dos mecanismos, instalação local, testes variados, coleta de dados e análise das limitações. Também a modelagem dos dados dos procedimentos de proteína em *workflows* científicos foi um trabalho complexo, de forma a representar com fidelidade o comportamento dos dados, as funções principais, as dependências e o tamanho das tarefas.

Tanto o modelo de aprendizado de máquina quanto os algoritmos de simulação têm diversos pontos de complexidade alta para sua concretização. A definição da carga de trabalho, do ambiente computacional e das estratégias de implementação também foram desafiadoras, devido ao número de fatores envolvidos e às amarras da integração de projetos de áreas distintas. É preciso validar os parâmetros sob diversos pontos de vista, para evitar falhas de avaliação.

As colaborações foram fundamentais para a evolução do trabalho. O aprendizado e a experiência foram enriquecedores. A interconexão da computação com as demais áreas do conhecimento tende a continuar crescendo e mais iniciativas de cooperação são fundamentais. A ausência de outros pesquisadores atuando paralelamente no projeto também conta entre as dificuldades. É esperado que a continuidade do desenvolvimento da MDAPSP envolva alguns profissionais trabalhando de forma colaborativa e incremental. Entre outros fatores, esses foram os principais que causaram o adiamento das publicações advindas do projeto, sendo publicadas no último quarto do período.

8.6 Trabalhos Futuros

O principal foco da sequência deste trabalho é a prototipação da MDAPSP, criando um ambiente funcional disponível para os pesquisadores. É esperado que inicialmente a arquitetura forneça ferramentas para a condução de experimentos de PSP, porém nada impede que sua modelagem seja generalizada para outros domínios. Esse protótipo deve seguir os estudos e padronizações descritos nessa tese, continuando as investigações realizadas. O mecanismo do *Decision Maker*, os escalonadores da arquitetura e o modelo heterogêneo de provisionamento computacional são as bases para a construção da arquitetura.

Paralelamente o avanço do modelo simulado pode guiar o processo de prototipação e progredir nas análises em cenários sintéticos, guiando o investimento em máquinas, as tecnologias necessárias, estudos de compatibilidade e análise de modelos de dados, por exemplo.

A MDAPSP prevê ainda o desenvolvimento dos seguintes pontos:

- **Avanço no uso de aprendizado de máquina:** com o aumento da complexidade dos parâmetros utilizados, outros algoritmos de aprendizado podem ser avaliados no contexto do *Decision Maker*, assim como outros potenciais módulos da arquitetura. Ajustes conforme a classe do usuário ou modificações em tempo de execução podem ser considerados para uma decisão mais acertada e precisa.
- **Algoritmos multiobjetivos:** diversos dos pontos de decisão na MDAPSP necessitam da definição de custo-benefício. Nesses casos, algoritmos que busquem uma solução balanceada entre dois parâmetros conflitantes podem ser considerados, para definição automatizada do melhor caso, inclusive variando o peso dos parâmetros conforme a decisão do usuário.
- **Variação dos preditores:** a proposta da arquitetura, assim como o modelo de simulação, permitem a extensão dos estudos para os demais preditores disponíveis no Koala e em outros *gateways*. É esperado que o modelo de carga de cada um deles seja diferente, implicando em diferenças nos esquemas de escalonamento de tarefas.
- **Estudo do cálculo de minimização:** considerar o procedimento de minimização das energias que atuam na estrutura das proteínas. Essa análise em ambiente distribuído pode oferecer ganhos de desempenho significativos.
- **Paralelização de PSP:** os procedimentos em PSP são majoritariamente sequenciais até o momento, sendo a computação paralela uma das grandes lacunas ainda a serem preenchidas.
- **Modelo de economia de eletricidade:** a proposta prevê o estudo de consumo de energia em *datacenters*, investigando a migração e a escalabilidade vertical e horizontal de máquinas virtuais.
- **Modelo de economia de custos:** a simulação permite o estudo de custos financeiros, que envolvam as instâncias de máquinas virtuais em nuvem, escalonando de acordo com classes de usuários e combinações entre investimento e demais requisitos.
- **Computação social:** a definição de tamanhos de tarefas e pesos de experimentos variados permite a estruturação de um sistema de computação que forneça instâncias gratuitas de máquinas no *cluster* por exemplo, para usuários de baixa renda. Horários vagos ou máquinas ociosas podem ser oferecidos gratuitamente para experimentos agendados, que não onerem o ambiente computacional e ofereçam oportunidades a todos com interesse em pesquisar.

Outros projetos acadêmicos deverão derivar desse, possivelmente envolvendo programação concorrente, segurança de dados e tolerância a falhas. Necessidades apontadas pela Bioinformática

também poderão ser supridas por pesquisas semelhantes a essa. A colaboração interdepartamental representa um avanço para todos os envolvidos e está alinhada com os rumos tomados pela Ciência da Computação e o arcabouço formado pelos conceitos de *e-Science*.

REFERÊNCIAS

- ABDUL-WAHID, B.; YU, L.; RAJAN, D.; FENG, H.; DARVE, E.; THAIN, D.; IZA-GUIRRE, J. Folding proteins at 500 ns/hour with work queue. In: **E-Science (e-Science)**, **2012 IEEE 8th International Conference on**. [s.n.], 2012. p. 1–8. Acessado em 14 out. 2014. Disponível em: <http://www.pubfacts.com/detail/25540799/Folding-Proteins-at-500-nshour-with-Work-Queue>. Citado na página 77.
- ABU-MOSTAFA, Y. S.; MAGDON-ISMAIL, M.; LIN, H.-T. **Learning from data**. [S.l.]: AMLBook New York, NY, USA:, 2012. v. 4. Citado na página 109.
- AFGAN, E.; BAKER, D.; BEEK, M. van den; BLANKENBERG, D.; BOUVIER, D.; ČECH, M.; CHILTON, J.; CLEMENTS, D.; CORAOR, N.; EBERHARD, C.; GRÜNING, B.; GUERLER, A.; HILLMAN-JACKSON, J.; KUSTER, G. V.; RASCHE, E.; SORANZO, N.; TURAGA, N.; TAYLOR, J.; NEKRUTENKO, A.; GOECKS, J. The galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update. **Nucleic Acids Research**, v. 44, n. W1, p. W3, 2016. Acessado em 13 mar. 2017. Disponível em: <http://dx.doi.org/10.1093/nar/gkw343>. Citado nas páginas 29, 58 e 83.
- AGGARWAL, C. C. **Data mining: the textbook**. [S.l.]: Springer, 2015. Citado nas páginas 106 e 107.
- AGOSTINI, F. P.; SOARES-PINTO, D. D. O.; MORET, M. A.; OSTHOFF, C.; PASCUTTI, P. G. Generalized simulated annealing applied to protein folding studies. **Journal of Computational Chemistry**, Wiley Subscription Services, Inc., A Wiley Company, v. 27, n. 11, p. 1142–1155, 2006. ISSN 1096-987X. Acessado em 06 mar. 2014. Disponível em: <http://www.ncbi.nlm.nih.gov/pubmed/16732545>. Citado na página 38.
- AGUILERA, A.; GRUNZKE, R.; HABICH, D.; LUONG, J.; SCHOLLBACH, D.; MARKWARDT, U.; GARCKE, J. Advancing a gateway infrastructure for wind turbine data analysis. **J. Grid Comput.**, Springer-Verlag New York, Inc., Secaucus, NJ, USA, v. 14, n. 4, p. 499–514, dez. 2016. ISSN 1570-7873. Acessado em 13 mar. 2017. Disponível em: <https://doi.org/10.1007/s10723-016-9376-9>. Citado na página 71.
- AHMAD, B.; LEITNER, I.; KRIEGER, M. T. Pvd: A lightweight distributed monitoring architecture for cloud infrastructures. In: **2015 IEEE Fourth Symposium on Network Cloud Computing and Applications (NCCA)**. [s.n.], 2015. p. 107–110. Acessado em 13 fev. 2017. Disponível em: ieeexplore.ieee.org/document/7340055. Citado na página 74.
- ALBERTS, B.; JOHNSON, A.; LEWIS, J.; RAFF, M.; KEITH, R.; PETER, W. The shape and structure of proteins. **Molecular Biology of the Cell**, 2002. Acessado em 06 mar. 2014. Disponível em: <http://www.ncbi.nlm.nih.gov/books/NBK26830/>. Citado na página 40.
- ALM, E.; BAKER, D. Prediction of protein-folding mechanisms from free-energy landscapes derived from native structures. **Journal of Molecular Biology**, v. 96, p. 11305–11310, September 1999. Acessado em 12 dez. 2014. Disponível em: <http://www.ncbi.nlm.nih.gov/pubmed/10500172>. Citado nas páginas 27 e 37.

ALNAHDI, A.; LIU, S. H. Identifying characteristic attributes for estimating cost of service in service oriented architecture. In: **2017 IEEE International Conference on Services Computing (SCC)**. [s.n.], 2017. p. 467–470. Acessado em 23 nov. 2017. Disponível em: <http://ieeexplore.ieee.org/document/8035019/>>. Citado na página 76.

ALPER, P.; BELHAJJAME, K.; GOBLE, C. A. Automatic vs manual provenance abstractions: Mind the gap. In: **Proceedings of the 8th USENIX Conference on Theory and Practice of Provenance**. Berkeley, CA, USA: USENIX Association, 2016. (TaPP'16), p. 34–39. Acessado em 23 nov. 2017. Disponível em: <http://dl.acm.org/citation.cfm?id=3026947.3026954>>. Citado na página 71.

_____. Static analysis of taverna workflows to predict provenance patterns. **Future Generation Computer Systems**, v. 75, p. 310 – 329, 2017. ISSN 0167-739X. Acessado em 23 nov. 2017. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X17300225>>. Nenhuma citação no texto.

AMATO, N. M.; DILL, K. A.; SONG, G. Using motion planning to map protein folding landscapes and analyze folding kinetics of known native structures. In: **Proceedings of the Sixth Annual International Conference on Computational Biology**. New York, NY, USA: ACM, 2002. (RECOMB '02), p. 2–11. ISBN 1-58113-498-3. Acessado em 12 dez. 2014. Disponível em: <http://doi.acm.org/10.1145/565196.565198>>. Citado na página 37.

ANEES, T.; ZEILINGER, H. Performance evaluation of a service availability model. In: **Intelligent Engineering Systems (INES), 2014 18th International Conference on**. [s.n.], 2014. p. 115–120. Acessado em 16 jan. 2015. Disponível em: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=6909353>>. Citado na página 76.

ARABNEJAD, H.; BARBOSA, J. G. Multi-qos constrained and profit-aware scheduling approach for concurrent workflows on heterogeneous systems. **Future Generation Computer Systems**, v. 68, p. 211 – 221, 2017. ISSN 0167-739X. Acessado em 14 out. 2017. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X16303740>>. Citado nas páginas 71 e 75.

ARDIZZONE, V.; BARBERA, R.; CALANDUCCI, A.; FARGETTA, M.; INGRÀ, E.; PORRO, I.; ROCCA, G. L.; MONFORTE, S.; RICCI, R.; ROTONDO, R.; SCARDACI, D.; SCHE-
NONE, A. The decide science gateway. **Journal of Grid Computing**, v. 10, n. 4, p. 689–707, Dec 2012. ISSN 1572-9184. Acessado em 23 nov. 2014. Disponível em: <https://doi.org/10.1007/s10723-012-9242-3>>. Citado na página 58.

ARMBRUST, M.; FOX, A.; GRIFFITH, R.; JOSEPH, A. D.; KATZ, R. H.; KONWINSKI, A.; LEE, G.; PATTERSON, D. A.; RABKIN, A.; STOICA, I.; ZAHARIA, M. **Above the Clouds: A Berkeley View of Cloud Computing**. [S.l.], 2009. Acessado em 13 out. 2014. Disponível em: <http://www2.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html>>. Citado nas páginas 47, 69 e 121.

ARSLAN, F. Towards Service Oriented Architecture (SOA) for Massive Multiplayer Online Games (MMOG). **2012 UKSim 14th International Conference on Computer Modelling and Simulation**, Ieee, p. 538–543, mar. 2012. Acessado em 13 out. 2013. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6205504>>. Citado na página 30.

BAGHERMOUSAVI, S.; RASHIDI, H.; HAGHIGHI, H. An approach to reduce qos monitoring overhead in esb. In: **e-Commerce in Developing Countries: With Focus on e-Trust (ECDC), 2014 8th International Conference on**. [s.n.], 2014. p. 1–5. Acessado em 07 jan. 2015. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6836748>. Citado na página 77.

BAHRAMI, M.; SINGHAL, M. Dccsoa: A dynamic cloud computing service-oriented architecture. In: **2015 IEEE International Conference on Information Reuse and Integration**. [s.n.], 2015. p. 158–165. Acessado em 14 jun. 2016. Disponível em: <http://ieeexplore.ieee.org/document/7300969/>. Citado nas páginas 30, 77 e 80.

BAI, L.; YANG, L. A unified deep learning model for protein structure prediction. In: **2017 3rd IEEE International Conference on Cybernetics (CYBCONF)**. [s.n.], 2017. p. 1–6. Acessado em 23 nov. 2017. Disponível em: <http://ieeexplore.ieee.org/document/7985752/>. Citado na página 68.

BALASKO, A.; FARKAS, Z.; KACSUK, P. Building science gateways by utilizing the generic ws-pgrade/guse workflow system. **Computer Science**, v. 14, n. 2, p. 307, 2013. ISSN 2300-7036. Acessado em 26 fev. 2017. Disponível em: <https://journals.agh.edu.pl/csci/article/view/284>. Citado nas páginas 29, 55, 56, 57, 58 e 73.

BALIS, B. Hyperflow: A model of computation, programming approach and enactment engine for complex distributed workflows. **Future Generation Computer Systems**, v. 55, p. 147 – 162, 2016. ISSN 0167-739X. Acessado em 23 nov. 2017. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X15002770>. Citado na página 72.

BANHALMI, A.; PACZOLAY, D.; SVEGH, A.; ANTAL, G.; BILICKI, V. Development of a novel semantic-based system integration framework. In: **Engineering of Computer Based Systems (ECBS-EERC), 2013 3rd Eastern European Regional Conference on the**. [s.n.], 2013. p. 18–24. Acessado em 16 jan. 2014. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6664505. Citado na página 76.

BARRADAS-BAUTISTA, D.; ROSELL, M.; PALLARA, C.; FERNÁNDEZ-RECIO, J. Structural prediction of protein–protein interactions by docking: Application to biomedical problems. In: . Academic Press, 2017, (Advances in Protein Chemistry and Structural Biology). Acessado em 12 dez. 2017. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1876162317300433>. Citado na página 68.

BASKIN, D.; OLSON, N. E.; LUCAS, L.; SMIT odd. **GeneSifter: Next Generation Data Management and Analysis for Next Generation Sequencing**. 2009. Acessado em 13 out. 2014. Disponível em: https://www.researchgate.net/publication/241643396_GeneSifter_Next_Generation_Data_Management_and_Analysis_for_Next_Generation-Sequencing. Citado na página 58.

BASNEY, J.; MARTIN, S.; NAVARRO, J.; PIERCE, M.; SCAVO, T.; STRAND, L.; URAM, T.; WILKINS-DIEHR, N.; WU, W.; YOUN, C. The problem solving environments of teragrid, science gateways, and the intersection of the two. In: **2008 IEEE Fourth International Conference on eScience**. [s.n.], 2008. p. 725–734. Acessado em 15 jul. 2015. Disponível em: <http://ieeexplore.ieee.org/document/4736891/>. Citado nas páginas 56 e 57.

- BASSEM, C.; BESTAVROS, A. Multi-capacity bin packing with dependent items and its application to the packing of brokered workloads in virtualized environments. **Future Generation Computer Systems**, v. 72, n. Supplement C, p. 129 – 144, 2017. ISSN 0167-739X. Acessado em 13 jan. 2018. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X16302771>>. Citado na página 74.
- BENEVENTANO, D.; ORSINI, M.; PO, L.; SALA, A.; SORRENTINO, S. An ontology-based data integration system for data and multimedia sources. In: **Semantic Computing, 2009. ICSC '09. IEEE International Conference on**. [s.n.], 2009. p. 606–611. Acessado em 12 fev. 2014. Disponível em: <<https://www.computer.org/csdl/proceedings/icsc/2009/3800/00/3800a606.pdf>>. Citado na página 76.
- BENHADDI, M.; BAINA, K.; HASSAN, A. E. Soa for the masses: End users as services composers. In: **Computer Systems and Applications (AICCSA), 2013 ACS International Conference on**. [s.n.], 2013. p. 1–4. ISSN 2161-5322. Acessado em 13 out. 2014. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6616485>. Citado nas páginas 30 e 76.
- BHAGAT, J.; TANO, F.; NZUOBONTANE, E.; LAURENT, T.; ORLOWSKI, J.; ROOS, M.; WOLSTENCROFT, K.; ALEKSEJEVS, S.; STEVENS, R.; PETTIFER, S.; LOPEZ, R.; GOBLE, C. A. Biocatalogue: a universal catalogue of web services for the life sciences. **Nucleic Acids Research**, v. 38, n. suppl 2, p. W689–W694, 2010. Acessado em 06 mar. 2014. Disponível em: <<http://www.ncbi.nlm.nih.gov/pubmed/20484378>>. Citado nas páginas 58 e 75.
- BHAGWAN, J.; KUMAR, S. An intense review of task scheduling algorithms in cloud computing. **International Journal of Advanced Research in Computer and Communication Engineering**, v. 5, n. 11, nov 2016. Acessado em 13 jul. 2017. Disponível em: <<https://goo.gl/JePCYK>>. Citado na página 78.
- BHATNAGAR, R.; PATE, J. Performance analysis of a grid monitoring system - ganglia. **International Journal of Emerging Technology and Advanced Engineering**, v. 3, p. 362–365, aug 2013. Acessado em 13 jul. 2016. Disponível em: <<https://pdfs.semanticscholar.org/e454/37307a2a61a68a7314beed5b58c39946d912.pdf>>. Citado na página 102.
- BIANCHI, V.; CEOL, A.; OGIER, A. G.; PRETIS, S. de; GALEOTA, E.; KISHORE, K.; BORA, P.; CROCI, O.; CAMPANER, S.; AMATI, B.; MORELLI, M. J.; PELIZZOLA, M. Integrated Systems for NGS Data Management and Analysis: Open Issues and Available Solutions. **Front Genet**, v. 7, p. 75, 2016. Acessado em 15 jul. 2017. Disponível em: <<https://core.ac.uk/download/pdf/82852085.pdf>>. Citado na página 70.
- BOCHENINA, K. A comparative study of scheduling algorithms for the multiple deadline-constrained workflows in heterogeneous computing systems with time windows. **Procedia Computer Science**, v. 29, p. 509 – 522, 2014. ISSN 1877-0509. Acessado em 13 abr. 2015. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1877050914002233>>. Citado nas páginas 59 e 60.
- BORGETTO, D.; MAURER, M.; DA-COSTA, G.; PIERSON, J.; BRANDIC, I. Energy-efficient and sla-aware management of iaas clouds. In: **Future Energy Systems: Where Energy, Computing and Communication Meet (e-Energy), 2012 Third International Conference on**. [s.n.], 2012. p. 1–10. Acessado em 26 nov. 2014. Disponível em: <ieeexplore.ieee.org/document/6221120>. Citado na página 77.

- BORNICH, C.; GRYTTE, I.; HOVIG, E.; PAULSEN, J.; CECH, M.; SANDVE, G. K. Galaxy portal: interacting with the galaxy platform through mobile devices. **Bioinformatics**, v. 32, n. 11, p. 1743, 2016. Acessado em 13 jul. 2017. Disponível em: <http://dx.doi.org/10.1093/bioinformatics/btw042>. Citado nas páginas 27 e 73.
- BOSIN, A.; DESSI, N.; BAIRAPPAN, M.; PES, B. Will soa accommodate the next step of e-science? In: **2010 10th Annual International Conference on New Technologies of Distributed Systems (NOTERE)**. [s.n.], 2010. p. 303–308. ISSN 2162-1896. Acessado em 15 mai. 2015. Disponível em: ieeexplore.ieee.org/document/5536603. Citado nas páginas 75 e 80.
- BOURHIS, P.; DEUTCH, D.; MOSKOVITCH, Y. Analyzing data-centric applications: Why, what-if, and how-to. In: **2016 IEEE 32nd International Conference on Data Engineering (ICDE)**. [s.n.], 2016. p. 779–790. Acessado em 14 out. 2017. Disponível em: www.cs.tau.ac.il/~moskovitch1/docs/ICDE16_research_383.pdf. Citado na página 71.
- BRASIL, C. R. S.; DELBEM, A. C. B.; SILVA, F. L. B. da. Multiobjective evolutionary algorithm with many tables for purely ab initio protein structure prediction. **Journal of Computational Chemistry**, v. 34, n. 20, p. 1719–1734, 2013. ISSN 1096-987X. Acessado em 12 abr. 2014. Disponível em: <http://dx.doi.org/10.1002/jcc.23315>. Citado nas páginas 37, 39, 41 e 88.
- BROWN, K.; GROLINGER, K.; CAPRETZ, M. Data providing web service-based integration framework for use in a health care context. In: **Electrical and Computer Engineering (CCECE), 2011 24th Canadian Conference on**. [s.n.], 2011. p. 001069–001072. ISSN 0840-7789. Acessado em 15 fev. 2014. Disponível em: ieeexplore.ieee.org/document/6030625/. Citado nas páginas 76 e 80.
- BUTNARU, D.; BUSE, G.; PFLÜGER, D. A parallel and distributed surrogate model implementation for computational steering. In: **2012 11th International Symposium on Parallel and Distributed Computing**. [s.n.], 2012. p. 203–210. ISSN 2379-5352. Acessado em 15 out. 2015. Disponível em: ieeexplore.ieee.org/document/6341513. Citado na página 100.
- BUYA, R.; YEO, C. S.; VENUGOPAL, S.; BROBERG, J.; BRANDIC, I. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. **Future Generation Computer Systems**, v. 25, n. 6, p. 599 – 616, 2009. ISSN 0167-739X. Acessado em 13 out. 2014. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X08001957>. Citado nas páginas 30 e 46.
- CABIDDU, D.; ATTENE, M. Processing large geometric datasets in distributed environments. In: _____. **Transactions on Computational Science XXIX**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017. p. 97–120. ISBN 978-3-662-54563-8. Acessado em 13 jan. 2018. Disponível em: https://doi.org/10.1007/978-3-662-54563-8_6. Citado na página 79.
- CALHEIROS, R. N.; RANJAN, R.; BELOGLAZOV, A.; ROSE, C. A. F. D.; BUYA, R. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. **Softw. Pract. Exper.**, John Wiley & Sons, Inc., New York, NY, USA, v. 41, n. 1, p. 23–50, jan. 2011. ISSN 0038-0644. Acessado em 13 jan. 2018. Disponível em: <http://dx.doi.org/10.1002/spe.995>. Citado nas páginas 59, 60 e 129.
- CAMBIUCCI, W. **Arquitetura de Soluções**. 2014. Acessado em 16 abr. 2014. Disponível em: <http://blogs.msdn.com/b/wcamb/archive/2010/05/07/computa-o-em-nuvem-algumas-perguntas-sobre-desafios-em-projetos.aspx>. Citado na página 48.

- CAMILLO, M. H.; FANUCCHI, R. Z.; ROMERO, M. E.; LIMA, T. W. de; SOARES, A. da S.; DELBEM, A. C. B.; MARQUES, L. T.; MACIEL, C. D.; LONDON, J. B. A. Combining exhaustive search and multi-objective evolutionary algorithm for service restoration in large-scale distribution systems. **Electric Power Systems Research**, v. 134, p. 1 – 8, 2016. ISSN 0378-7796. Acessado em 14 set. 2017. Disponível em: <http://www.sciencedirect.com/science/article/pii/S037877961500379X>. Citado na página 78.
- CHAIM, R. M.; OLIVEIRA, E. C.; ARAUJO, A. P. F. Technical specifications of a service-oriented architecture for semantic interoperability of ehr — electronic health records. In: **2017 12th Iberian Conference on Information Systems and Technologies (CISTI)**. [s.n.], 2017. p. 1–6. Acessado em 15 dez. 2017. Disponível em: ieeexplore.ieee.org/document/7975923/. Citado na página 75.
- CHAMORRO, A.; DIVINA, F.; BACARDIT, J.; AGUILAR-RUIZ, J. An efficient decision rule-based system for the protein residue-residue contact prediction. In: **Evolutionary Computation (CEC), 2013 IEEE Congress on**. [s.n.], 2013. p. 592–597. Acessado em 19 nov. 2014. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6557622. Citado na página 65.
- CHARD, K.; LIDMAN, M.; BRYAN, J.; HOWE, T.; MCCOLLAM, B.; ANANTHAKRISHNAN, R.; TUECKE, S.; FOSTER, I. Globus nexus: Research identity, profile, and group management as a service. In: **2014 IEEE 10th International Conference on e-Science**. [s.n.], 2014. v. 1, p. 31–38. Acessado em 28 mar. 2015. Disponível em: ieeexplore.ieee.org/document/6972246/. Citado na página 73.
- CHARD, R.; CHARD, K.; BUBENDORFER, K.; LACINSKI, L.; MADDURI, R.; FOSTER, I. Cost-aware cloud provisioning. In: **2015 IEEE 11th International Conference on e-Science**. [s.n.], 2015. p. 136–144. Acessado em 07 nov. 2016. Disponível em: ieeexplore.ieee.org/document/7304284. Citado nas páginas 30, 73 e 75.
- CHAUDHARI, N.; BHADORIA, R. S.; PRASAD, S. Information handling and processing using enterprise service bus in service-oriented architecture system. In: **2016 8th International Conference on Computational Intelligence and Communication Networks (CICN)**. [s.n.], 2016. p. 418–421. Acessado em 23 nov. 2017. Disponível em: ieeexplore.ieee.org/document/8082679/. Citado na página 77.
- CHELARU, F.; SMITH, L.; GOLDSTEIN, N.; BRAVO, H. C. Epiviz: interactive visual analytics for functional genomics data. **Nature Methods**, Nature Publishing Group, a division of Macmillan Publishers Limited. All Rights Reserved., v. 11, n. 9, p. 938–940, ago. 2014. ISSN 1548-7091. Acessado em 19 nov. 2015. Disponível em: <http://dx.doi.org/10.1038/nmeth.3038>. Citado na página 29.
- CHEN, W.; DEELMAN, E. Workflowsim: A toolkit for simulating scientific workflows in distributed environments. In: **Proceedings of the 2012 IEEE 8th International Conference on E-Science (e-Science)**. Washington, DC, USA: IEEE Computer Society, 2012. (E-SCIENCE '12), p. 1–8. ISBN 978-1-4673-4467-8. Acessado em 15 jan. 2018. Disponível em: <http://dx.doi.org/10.1109/eScience.2012.6404430>. Citado nas páginas 31, 60, 61 e 129.
- CHEN, Y.; SUN, Y.; WU, H.; WU, Z.; HUANG, Z. A new method based on SOA for ACCS software integration. In: **Computer Design and Applications (ICCD), 2010 International Conference on**. [s.n.], 2010. v. 2, p. V2–319–V2–322. Acessado em 2 mar. 2014. Disponível em: <http://goo.gl/KVTbJC>. Citado nas páginas 30, 53 e 77.

CHEN, Y.; ZHOU, Z. Service-oriented computing and software integration in computing curriculum. In: **Parallel Distributed Processing Symposium Workshops (IPDPSW), 2014 IEEE International**. [s.n.], 2014. p. 1091–1098. Acessado em 09 fev. 2015. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6969503>. Citado nas páginas 42, 43 e 47.

CHENG, P.-H.; CHEN, H.-Y.; KAO, H.-Y. Protein surface search in dna-binding protein prediction by delaunay triangulation modeling. In: **Computer Symposium (ICS), 2010 International**. [s.n.], 2010. p. 783–788. Acessado em 09 nov. 2013. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5685406. Citado nas páginas 67 e 69.

CHRYSOS, G.; SOTIRIADES, E.; ROUSOPOULOS, C.; DOLLAS, A.; PAPADOPOULOS, A.; KIRMITZOGLU, I.; PROMPONAS, V.; THEOCHARIDES, T.; PETIHAKIS, G.; LAGNEL, J.; VAVYLIS, P.; KOTOULAS, G. Opportunities from the use of fpgas as platforms for bioinformatics algorithms. In: **2012 IEEE 12th International Conference on Bioinformatics Bioengineering (BIBE)**. [s.n.], 2012. p. 559–565. Acessado em 23 nov. 2015. Disponível em: ieeexplore.ieee.org/document/6399733. Citado na página 68.

CLASSE, T.; BRAGA, R.; DAVID, J. M. N.; CAMPOS, F.; ARAÚJO, M. A.; STRÖELE, V. A collaborative approach to support e-science activities. In: **2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design (CSCWD)**. [s.n.], 2016. p. 20–25. Acessado em 18 mai. 2017. Disponível em: ieeexplore.ieee.org/document/7565956. Citado na página 28.

CLEMENT, S. J.; MCKEE, D. W.; XU, J. Service-oriented reference architecture for smart cities. In: **2017 IEEE Symposium on Service-Oriented System Engineering (SOSE)**. [s.n.], 2017. p. 81–85. Acessado em 15 dez. 2017. Disponível em: ieeexplore.ieee.org/document/7943295. Citado na página 75.

COCK, P. J.; GRÜNING, B. A.; PASZKIEWICZ, K.; PRITCHARD, L. Galaxy tools and workflows for sequence analysis with applications in molecular plant pathology. **PeerJ**, v. 1, p. e167, set. 2013. ISSN 2167-8359. Acessado em 15 out. 2016. Disponível em: <https://doi.org/10.7717/peerj.167>. Citado nas páginas 104 e 118.

COHEN-BOULAKIA, S.; CHEN, J.; MISSIER, P.; GOBLE, C.; WILLIAMS, A.; FROIDEVAUX, C. Distilling structure in taverna scientific workflows: a refactoring approach. **BMC Bioinformatics**, v. 15, n. Suppl 1, p. S12, 2014. ISSN 1471-2105. Acessado em 26 mar. 2015. Disponível em: <http://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-15-S1-S12>. Citado na página 72.

COLLADO, A. A. C. **Management of generic and multi-platform workflows for exploiting heterogeneous environments on e-Science**. Tese (Doutorado) — Universitat Politècnica de València, 2017. Acessado em 15 fev. 2018. Disponível em: <https://riunet.upv.es/handle/10251/86179>. Citado nas páginas 54, 79 e 80.

COOPER, S.; KHATIB, F.; TREUILLE, A.; BARBERO, J.; LEE, J.; BEENEN, M.; LEAVERFAY, A.; BAKER, D.; POPOVIC, Z. Predicting protein structures with a multiplayer online game. **Nature**, 2010. Acessado em 16 jul. 2014. Disponível em: <http://www.nature.com/nature/journal/v466/n7307/full/nature09304.html>. Citado na página 42.

COTES-RUIZ, I. T.; PRADO, R. P.; GARCÍA-GALÁN, S.; MUÑOZ-EXPÓSITO, J. E.; RUIZ-REYES, N. Dynamic voltage frequency scaling simulator for real workflows energy-aware

management in green cloud computing. **PLOS ONE**, Public Library of Science, v. 12, n. 1, p. 1–30, 01 2017. Acessado em 15 jan. 2018. Disponível em: <<https://doi.org/10.1371/journal.pone.0169803>>. Citado nas páginas 61 e 62.

COUVARES, P.; KOSAR, T.; ROY, A.; WEBER, J.; WENGER, K. Workflow management in condor. In: _____. **Workflows for e-Science: Scientific Workflows for Grids**. London: Springer London, 2007. p. 357–375. ISBN 978-1-84628-757-2. Acessado em 14 set. 2015. Disponível em: <https://doi.org/10.1007/978-1-84628-757-2_22>. Citado na página 53.

CRANDELL, M. **2017 State of the Cloud Report**. 2017. [Http://www.rightscale.com/lp/2017-state-of-the-cloud-report](http://www.rightscale.com/lp/2017-state-of-the-cloud-report). Acessado em 15 jan. 2018. Citado na página 30.

CRICHTON, D.; HUGHES, J. S.; HARDMAN, S.; LAW, E.; BEEBE, R.; MORGAN, T.; GRAY-ZECK, E. A scalable planetary science information architecture for big science data. In: **2014 IEEE 10th International Conference on e-Science**. [s.n.], 2014. v. 1, p. 196–203. Acessado em 18 jul. 2015. Disponível em: <<https://www.researchgate.net/publication/281854433>>. Citado na página 75.

CRIVELLI, S.; DOOLEY, R.; HOLMES, R.; MOCK, S. The wefold gateway: Enabling large-scale science coepetition. In: **Cluster Computing (CLUSTER), 2013 IEEE International Conference on**. [s.n.], 2013. p. 1–3. Acessado em 14 out. 2014. Disponível em: <<http://goo.gl/NwmIzK>>. Citado na página 118.

DAHIMENE, S.; ROMÃO, A.; BOTELHO, H. M.; PALMA, M.; CORREIA, A. R.; FARINHA, C. M.; KING-UNDERWOOD, J.; WILLIAMS, J.; HIRST, S.; GOMES, C. M.; AMARAL, M. D. **Screening for novel F508del-CFTR correctors identified by pharmacophore modelling through automated microscopy and NBD1 folding assays**. 2013. S45 p. Acessado em 16 jan. 2015. Disponível em: <<https://goo.gl/xCSrmU>>. Citado na página 40.

DEAN, J.; GHEMAWAT, S. Mapreduce: Simplified data processing on large clusters. **Commun. ACM**, ACM, New York, NY, USA, v. 51, n. 1, p. 107–113, jan. 2008. ISSN 0001-0782. Acessado em 13 abr. 2015. Disponível em: <<http://dl.acm.org/citation.cfm?id=1327492>>. Citado nas páginas 30 e 48.

DEELMAN, E.; GANNON, D.; SHIELDS, M.; TAYLOR, I. Workflows and e-science: An overview of workflow system features and capabilities. **Future Generation Computer Systems**, v. 25, n. 5, p. 528 – 540, 2009. ISSN 0167-739X. Acessado em 14 set. 2015. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X08000861>>. Citado nas páginas 29, 50, 51, 52 e 53.

DEELMAN, E.; VAHI, K.; JUVE, G.; RYNGE, M.; CALLAGHAN, S.; MAECHLING, P. J.; MAYANI, R.; CHEN, W.; SILVA, R. F. da; LIVNY, M.; WENGER, K. Pegasus, a workflow management system for science automation. **Future Generation Computer Systems**, v. 46, p. 17 – 35, 2015. ISSN 0167-739X. Acessado em 13 ago. 2016. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0167739X14002015>>. Citado nas páginas 29, 51, 60, 120 e 135.

DEFELICIBUS, A. **Koala: sistema para integração de métodos de predição e análise de estruturas de proteína**. Dissertação (Mestrado) — Universidade de São Paulo, São Carlos - SP, 2016. Acessado em 15 dez. 2017. Citado nas páginas 38, 41, 84, 90, 91, 93 e 98.

DELDARI, A.; NAGHIBZADEH, M.; REZAEIAN, A.; ABRISHAMI, H. A clustering approach to schedule workflows to run on the cloud. In: **2016 Eighth International Conference on Information and Knowledge Technology (IKT)**. [s.n.], 2016. p. 122–127. Acessado em 28 fev. 2017. Disponível em: <ieeexplore.ieee.org/document/7777750/>. Citado na página 78.

DIAS, J.; OGASAWARA, E.; OLIVEIRA, D. de; PORTO, F.; VALDURIEZ, P.; MATTOSO, M. Algebraic dataflows for big data analysis. In: **2013 IEEE International Conference on Big Data**. [s.n.], 2013. p. 150–155. Acessado em 14 abr. 2014. Disponível em: <ieeexplore.ieee.org/document/6691567/>. Citado nas páginas 119 e 121.

DILL, K. A.; BROMBERG, S. **Molecular driving forces: statistical thermodynamics in biology, chemistry, physics, and nanoscience**. Garland Science, 2011. Disponível em: <<http://www.garlandscience.com/product/isbn/9780815344308>>. Citado na página 41.

D'MELLO, D. A.; SALIAN, S.; SAHANA; SHETTY, N.; RODRIGUES, C.; D'SOUZA, C. A service broker based architecture for effective discovery and automatic publishing of internet web services. In: **2012 2nd IEEE International Conference on Parallel, Distributed and Grid Computing**. [s.n.], 2012. p. 724–729. Acessado em 15 nov. 2014. Disponível em: <ieeexplore.ieee.org/document/6449911/>. Citado na página 43.

DORN, M.; SILVA, M. B. e; BURIOL, L. S.; LAMB, L. C. Three-dimensional protein structure prediction: Methods and computational strategies. **Computational Biology and Chemistry**, v. 53, p. 251 – 276, 2014. ISSN 1476-9271. Acessado em 13 jan. 2015. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1476927114001248>>. Citado nas páginas 27, 28 e 40.

DRIEMEIER, C. E.; LING, L. Y.; PONTES, A. O.; SANCHES, G. M.; FRANCO, H. C. J.; MAGALHÃES, P. S. G.; FERREIRA, J. E. Data analysis workflow for experiments in sugarcane precision agriculture. In: **2014 IEEE 10th International Conference on e-Science**. [s.n.], 2014. v. 1, p. 163–168. Acessado em 3 ago. 2015. Disponível em: <ieeexplore.ieee.org/document/6972261/>. Citado na página 69.

DUBEY, S. P.; KINI, N. G.; KUMAR, M. S.; BALAJI, S. Ab initio protein structure prediction using gpu computing. **Perspectives in Science**, v. 8, p. 645 – 647, 2016. ISSN 2213-0209. Acessado em 12 dez. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S2213020916301859>>. Citado na página 68.

DURILLO, J. J.; FARD, H. M.; PRODAN, R. Moheft: A multi-objective list-based method for workflow scheduling. In: **4th IEEE International Conference on Cloud Computing Technology and Science Proceedings**. [s.n.], 2012. p. 185–192. Acessado em 12 jul. 2015. Disponível em: <ieeexplore.ieee.org/document/6427573/>. Citado na página 71.

EBENHOFER, G.; BAUER, H.; PLASCH, M.; ZAMBAL, S.; AKKALADEVI, S.; PICHLER, A. A system integration approach for service-oriented robotics. In: **Emerging Technologies Factory Automation (ETFA), 2013 IEEE 18th Conference on**. [s.n.], 2013. p. 1–8. ISSN 1946-0740. Acessado em 16 jan. 2015. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6648029>. Citado na página 76.

ELIASSON, J.; DELSING, J.; RAAAYATINEZHAD, A.; KYUSAKOV, R. A soa-based framework for integration of intelligent rock bolts with internet of things. In: **Industrial Technology (ICIT), 2013 IEEE International Conference on**. [s.n.], 2013. p. 1962–1967. Acessado em 13 out. 2014. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6505979>. Citado nas páginas 30 e 76.

ERL, T. **SOA: Principles of Service Design**. [S.l.]: Prentice Hall, 2007. v. 1. ISBN 9780132344821. Citado na página 43.

ERRADI, A.; MAHESHWARI, P.; TOSIC, V. Recovery policies for enhancing web services reliability. In: **Web Services, 2006. ICWS '06. International Conference on**. [s.n.], 2006. p. 189–196. Acessado em 22 set. 2014. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4032027>. Citado nas páginas 42 e 44.

ESTRELLA, J. C. **WSARCH: Uma arquitetura para a provisão de web services com qualidade de serviço**. Tese (Doutorado) — Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação, 2010. Acessado em 14 mar. 2014. Disponível em: <<http://www.teses.usp.br/teses/disponiveis/55/55134/tde-13072010-134913/pt-br.php>>. Citado nas páginas 43, 45 e 101.

FABRE, M. G.; PULIDO, G. T.; TELLO, E. R. Handling constraints in the hp model for protein structure prediction by multiobjective optimization. In: **Evolutionary Computation (CEC), 2013 IEEE Congress on**. [s.n.], 2013. p. 2728–2735. Acessado em 3 ago. 2014. Disponível em: <<http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6557899>>. Citado nas páginas 65 e 69.

FACCIOLI, R.; SILVA, I. da; DELBEM, A.; BRANCINI, G.; CALIRI, A. Protpred-gromacs: Evolutionay algorithm with gromacs for protein structure prediction. **International Symposium on Mathematical and Computational Biology**, v. 5, p. 10, 2011. Acessado em 13 abr. 2015. Disponível em: <<https://goo.gl/qzjbs9>>. Citado na página 90.

FACCIOLI, R. A. **Implementação de um Framework de Computação Evolutiva Multi-Objetivo para Predição Ab Initio da Estrutura Terciária de Proteínas**. Tese (Doutorado) — Instituto de Ciências Matemáticas e de Computação - USP, 2012. Acessado em 13 fev. 2014. Disponível em: <<http://www.teses.usp.br/teses/disponiveis/18/18153/tde-09052013-145839/pt-br.php>>. Citado nas páginas 27, 39, 41, 67, 102 e 107.

FACCIOLI, R. A.; BORTOT, L. O.; DELBEM, A. C. B. Multi-Objective Evolutionary Algorithm NSGA-II for Protein Structure Prediction using Structural and Energetic Properties. **International Journal of Natural Computing Research (IJNCR)**, v. 4, n. 1, p. 43–53, January 2014. Acessado em 15 fev. 2015. Disponível em: <<https://ideas.repec.org/a/igg/jncr00/v4y2014i1p43-53.html>>. Citado na página 83.

FACI, N.; ABDELDELJELIL, H.; MAAMAR, Z.; BENSLIMANE, D. Using diversity to design and deploy fault tolerant web services. In: **Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2011 20th IEEE International Workshops on**. [s.n.], 2011. p. 73–78. ISSN 1524-4547. Acessado em 13 set. 2014. Disponível em: <<http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5990004>>. Citado na página 44.

FAHRINGER, T.; PRODAN, R.; DUAN, R.; NERIERI, F.; PODLIPNIG, S.; QIN, J.; SIDDIQUI, M.; TRUONG, H.-L.; VILLAZON, A.; WIECZOREK, M. Askalon: a grid application development and computing environment. In: **The 6th IEEE/ACM International Workshop on Grid Computing, 2005**. [s.n.], 2005. p. 10 pp.–. ISSN 2152-1085. Acessado em 13 out. 2014. Disponível em: <ieeexplore.ieee.org/document/1542733/>. Citado na página 53.

FARAHNAKIAN, F.; LILJEBERG, P.; PLOSILA, J. Energy-efficient virtual machines consolidation in cloud data centers using reinforcement learning. In: **2014 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing**. [s.n.], 2014. p. 500–507. ISSN 1066-6192. Acessado em 15 ago. 2016. Disponível em: <ieeexplore.ieee.org/document/6787321>. Citado na página 78.

FARD, H. M.; DELDARI, H. An economic approach for scheduling dependent tasks in grid computing. In: **2008 11th IEEE International Conference on Computational Science and Engineering - Workshops**. [s.n.], 2008. p. 71–76. Acessado em 12 jul. 2015. Disponível em: [<ieeexplore.ieee.org/document/4625042/>](http://ieeexplore.ieee.org/document/4625042/). Citado na página 71.

FARD, H. M.; FAHRINGER, T.; PRODAN, R. Budget-constrained resource provisioning for scientific applications in clouds. In: **2013 IEEE 5th International Conference on Cloud Computing Technology and Science**. [s.n.], 2013. v. 1, p. 315–322. Acessado em 12 jul. 2015. Disponível em: [<ieeexplore.ieee.org/document/6753813/>](http://ieeexplore.ieee.org/document/6753813/). Citado na página 31.

FARD, H. M.; PRODAN, R.; BARRIONUEVO, J. J. D.; FAHRINGER, T. A multi-objective approach for workflow scheduling in heterogeneous environments. In: **2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012)**. [s.n.], 2012. p. 300–309. Acessado em 12 jul. 2015. Disponível em: [<ieeexplore.ieee.org/document/6217435/>](http://ieeexplore.ieee.org/document/6217435/). Citado na página 28.

FARD, H. M.; PRODAN, R.; FAHRINGER, T. A truthful dynamic workflow scheduling mechanism for commercial multicloud environments. **IEEE Transactions on Parallel and Distributed Systems**, v. 24, n. 6, p. 1203–1212, June 2013. ISSN 1045-9219. Acessado em 12 jul. 2015. Disponível em: [<ieeexplore.ieee.org/document/6295608/>](http://ieeexplore.ieee.org/document/6295608/). Citado nas páginas 71 e 75.

_____. Multi-objective list scheduling of workflow applications in distributed computing infrastructures. **Journal of Parallel and Distributed Computing**, v. 74, n. 3, p. 2152 – 2165, 2014. ISSN 0743-7315. Acessado em 12 jul. 2015. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0743731513002384>. Nenhuma citação no texto.

FARD, H. M.; PRODAN, R.; MOSER, G.; FAHRINGER, T. A bi-criteria truthful mechanism for scheduling of workflows in clouds. In: **2011 IEEE Third International Conference on Cloud Computing Technology and Science**. [s.n.], 2011. p. 599–605. Acessado em 12 jul. 2015. Disponível em: [<ieeexplore.ieee.org/document/6133201/>](http://ieeexplore.ieee.org/document/6133201/). Citado na página 71.

FARD, H. M.; RISTOV, S.; PRODAN, R. Handling the uncertainty in resource performance for executing workflow applications in clouds. In: **Proceedings of the 9th International Conference on Utility and Cloud Computing**. New York, NY, USA: ACM, 2016. (UCC '16), p. 89–98. ISBN 978-1-4503-4616-0. Acessado em 12 jul. 2017. Disponível em: <http://doi.acm.org/10.1145/2996890.2996902>. Citado nas páginas 71 e 75.

FELJAN, A. V.; MOHALIK, S. K.; JAYARAMAN, M. B.; BADRINATH, R. Soa-pe: A service-oriented architecture for planning and execution in cyber-physical systems. In: **2015 International Conference on Smart Sensors and Systems (IC-SSS)**. [s.n.], 2015. p. 1–6. Acessado em 16 out. 2016. Disponível em: [<https://ieeexplore.ieee.org/document/7873602/>](https://ieeexplore.ieee.org/document/7873602/). Citado nas páginas 44, 76 e 80.

FERNANDEZ, H.; TEDESCHI, C.; PRIOL, T. A chemistry-inspired workflow management system for decentralizing workflow execution. **IEEE Transactions on Services Computing**, v. 9, n. 2, p. 213–226, March 2016. ISSN 1939-1374. Acessado em 28 fev. 2017. Disponível em: [<ieeexplore.ieee.org/document/6514026/>](http://ieeexplore.ieee.org/document/6514026/). Citado nas páginas 31, 77 e 80.

FIGLIORE, S.; ELIA, D.; BLANQUER, I.; BRASILEIRO, F. V.; NUZZO, A.; NASSISI, P.; RUFINO, I. A.; SEIJMONSBERGEN, A. C.; ANDERS, N. S.; GALVAO, C. de O.; CUNHA,

- J. E. de B.; CABALLER, M.; SOUSA-BAENA, M. S.; CANHOS, V. P.; ALOISIO, G. Bioclimate: A science gateway for climate change and biodiversity research in the {EUBrazil-CloudConnect} project. **Future Generation Computer Systems**, p. –, 2017. ISSN 0167-739X. Acessado em 15 fev. 2018. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0167739X17312591>>. Citado nas páginas 56, 79 e 80.
- FLANAGAN, M. T. **Michael Thomas Flanagan's Java Scientific Library**. 2016. <https://www.ee.ucl.ac.uk/mflanaga/java/>. Citado na página 129.
- FLOUDAS, C. Computational methods in protein structure prediction. **Biotechnology and Bioengineering**, Wiley Subscription Services, Inc., A Wiley Company, v. 97, n. 2, p. 207–213, 2007. ISSN 1097-0290. Acessado em 13 mar. 2017. Disponível em: <<http://dx.doi.org/10.1002/bit.21411>>. Citado nas páginas 40 e 41.
- FOLDING@HOME. **Folding@Home**. 2013. Acessado em 21 nov. 2013. Disponível em: <<http://folding.stanford.edu/>>. Citado nas páginas 42 e 46.
- FOLDIT. **Fold It**. 2013. Acessado em 21 nov. 2013. Disponível em: <<http://fold.it/portal/>>. Citado na página 42.
- GAN, Q.; ZHENG, J.-H. A new algorithm to improve efficiency of resource scheduling in clouding computing based on extended support vector machine. **International Journal of Grid and Distributed Computing**, v. 9, n. 3, p. 125–134, 2016. Acessado em 13 fev. 2017. Disponível em: <www.sersc.org/journals/IJGDC/vol9_no3/15.pdf>. Citado na página 78.
- GANGA, K.; KARTHIK, S. A fault tolerant approach in scientific workflow systems based on cloud computing. In: **2013 International Conference on Pattern Recognition, Informatics and Mobile Engineering**. [s.n.], 2013. p. 387–390. Acessado em 14 abr. 2014. Disponível em: <ieeexplore.ieee.org/document/6496507/>. Citado na página 121.
- GARZA, L. de la; AICHELER, F.; KOHLBACHER, O. From the desktop to the grid and cloud: Conversion of knime workflows to guse. **Proc. Int. Workshop Science Gateways (IWSG 2016)**, 2016. Acessado em 23 nov. 2017. Disponível em: <<https://peerj.com/preprints/2849.pdf>>. Citado nas páginas 30, 73, 75 e 117.
- GARZA, L. de la; VEIT, J.; SZOLEK, A.; RÖTTIG, M.; AICHE, S.; GESING, S.; REINERT, K.; KOHLBACHER, O. From the desktop to the grid: scalable bioinformatics via workflow conversion. **BMC Bioinformatics**, v. 17, n. 1, p. 127, Mar 2016. ISSN 1471-2105. Acessado em 23 nov. 2017. Disponível em: <<https://doi.org/10.1186/s12859-016-0978-9>>. Citado nas páginas 73 e 119.
- GERALDO, M. T.; VALENTE, G. **Predição de Estruturas de Proteínas e construção de redes de interação proteica**. 2014. Acessado em 20 nov. 2014. Disponível em: <<http://goo.gl/MS1bTJ>>. Citado na página 41.
- GESING, S.; DOOLEY, R.; PIERCE, M.; KRÜGER, J.; GRUNZKE, R.; HERRES-PAWLIS, S.; HOFFMANN, A. Gathering requirements for advancing simulations in hpc infrastructures via science gateways. **Future Generation Computer Systems**, 2017. ISSN 0167-739X. Acessado em 23 nov. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X17303011>>. Citado nas páginas 29, 54, 55, 69 e 83.

GESING, S.; KRÜGER, J.; GRUNZKE, R.; HERRES-PAWLIS, S.; HOFFMANN, A. Using science gateways for bridging the differences between research infrastructures. **Journal of Grid Computing**, v. 14, n. 4, p. 545–557, 2016. ISSN 1572-9184. Acessado em 19 mar. 2017. Disponível em: <<http://dx.doi.org/10.1007/s10723-016-9385-8>>. Citado nas páginas 29 e 56.

GESING, S.; WILKINS-DIEHR, N.; DAHAN, M.; LAWRENCE, K.; ZENTNER, M.; PIERCE, M.; HAYDEN, L.; MARRU, S. Science gateways: The long road to the birth of an institute. **Hawaii International Conference on System Sciences (HICSS)**, 2017. Acessado em 13 jan. 2018. Disponível em: <<https://pdfs.semanticscholar.org/6136/cdb994e2ff7a3c840ab8e3c8e9fc658e2072.pdf>>. Citado nas páginas 28 e 55.

GHASEMI, M.; MATTA, I. Pricing differentiated brokered internet services. In: **2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)**. [s.n.], 2016. p. 124–129. Acessado em 15 nov. 2017. Disponível em: <ieeexplore.ieee.org/abstract/document/7562058/>. Citado na página 43.

GIANNOPOULOU, E. G.; ELEMENTO, O. An integrated chip-seq analysis platform with customizable workflows. **BMC Bioinformatics**, v. 12, n. 1, p. 277, Jul 2011. ISSN 1471-2105. Acessado em 15 nov. 2014. Disponível em: <<https://doi.org/10.1186/1471-2105-12-277>>. Citado na página 58.

GOECKS, J.; NEKRUTENKO, A.; TAYLOR, J.; TEAM, T. G. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. **Genome Biology**, v. 11, n. 8, p. R86, 2010. ISSN 1465-6906. Acessado em 06 mar. 2014. Disponível em: <<http://www.ncbi.nlm.nih.gov/pubmed/20738864>>. Citado nas páginas 29 e 83.

GOHER, S. Z.; BLOODSWORTH, P.; RASOOL, R. U.; MCCLATCHEY, R. Cloud provider capacity augmentation through automated resource bartering. **Future Generation Computer Systems**, v. 81, p. 203 – 218, 2018. ISSN 0167-739X. Acessado em 06 mar. 2018. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X17304867>>. Citado nas páginas 79 e 80.

GOODALE, T.; JHA, S.; KAISER, H.; KIELMANN, T.; KLEIJER, P.; MERZKY, A.; SHALF, J.; SMITH, C. A Simple API for Grid Applications (SAGA). [S.l.], 2007. Acessado em 18 jul. 2015. Disponível em: <<http://ogf.org/documents/GFD.90.pdf>>. Citado na página 56.

GÖRLACH, K.; SONNTAG, M.; KARASTOYANOVA, D.; LEYMAN, F.; REITER, M. Conventional workflow technology for scientific simulation. In: _____. **Guide to e-Science: Next Generation Scientific Research and Discovery**. London: Springer London, 2011. p. 323–352. ISBN 978-0-85729-439-5. Acessado em 14 set. 2015. Disponível em: <https://doi.org/10.1007/978-0-85729-439-5_12>. Citado nas páginas 28 e 52.

GROTH, P.; GIL, Y. Analyzing the gap between workflows and their natural language descriptions. In: **2009 Congress on Services - I**. [s.n.], 2009. p. 299–305. ISSN 2378-3818. Acessado em 22 abr. 2014. Disponível em: <<https://dl.acm.org/citation.cfm?id=1591566>>. Citado na página 70.

GRUNZKE, R. **Generic Metadata Handling in Scientific Data Life Cycles**. Tese (Doutorado) — Technische Universität Dresden, 2016. Acessado em 13 jan. 2017. Disponível em: <www.qucosa.de/urnnbn/urn:nbn:de:bsz:14-qucosa-202070>. Citado na página 53.

GRÜNING, B. A.; RASCHE, E.; REBOLLEDO-JARAMILLO, B.; EBERHARD, C.; HOUWART, T.; CHILTON, J.; CORAOR, N.; BACKOFEN, R.; TAYLOR, J.; NEKRUTENKO, A. Jupyter and galaxy: Easing entry barriers into complex data analyses for biomedical researchers. **PLOS Computational Biology**, Public Library of Science, v. 13, n. 5, p. 1–10, 05 2017. Acessado em 23 nov. 2017. Disponível em: <<https://doi.org/10.1371/journal.pcbi.1005425>>. Citado na página 74.

GUGNANI, S.; BLANCO, C.; KISS, T.; TERSTYANSZKY, G. Extending science gateway frameworks to support big data applications in the cloud. **Journal of Grid Computing**, v. 14, n. 4, p. 589–601, Dec 2016. ISSN 1572-9184. Acessado em 23 nov. 2017. Disponível em: <<https://doi.org/10.1007/s10723-016-9369-8>>. Citado nas páginas 56 e 73.

GUIMARAES, F. P.; CÉLESTIN, P.; BATISTA, D. M.; RODRIGUES, G. N.; MELO, A. C. M. A. de. A framework for adaptive fault-tolerant execution of workflows in the grid: Empirical and theoretical analysis. **Journal of Grid Computing**, v. 12, n. 1, p. 127–151, 2014. ISSN 1572-9184. Acessado em 13 ago. 2016. Disponível em: <<http://dx.doi.org/10.1007/s10723-013-9281-4>>. Citado nas páginas 57 e 79.

GÜNTERT, P. Automated nmr structure calculation with cyana. In: _____. **Protein NMR Techniques**. Totowa, NJ: Humana Press, 2004. p. 353–378. ISBN 978-1-59259-809-0. Acessado em 13 jan. 2015. Disponível em: <<https://doi.org/10.1385/1-59259-809-9:353>>. Citado na página 28.

GUO, Y.; ZHU, H.; YANG, L. Service-oriented network virtualization architecture for internet of things. **China Communications**, v. 13, n. 9, p. 163–172, Sept 2016. ISSN 1673-5447. Acessado em 23 nov. 2017. Disponível em: <ieeexplore.ieee.org/document/7582308/>. Citado na página 75.

GUPTA, J.; AZHARUDDIN, M.; JANA, P. K. An effective task scheduling approach for cloud computing environment. In: _____. **Proceedings of the International Conference on Signal, Networks, Computing, and Systems: ICSNCS 2016, Volume 2**. New Delhi: Springer India, 2016. p. 163–169. ISBN 978-81-322-3589-7. Acessado em 13 abr. 2017. Disponível em: <https://doi.org/10.1007/978-81-322-3589-7_17>. Citado nas páginas 30, 74 e 75.

GUSEW, W.; SCHEUERMANN, B. Starting workflow tasks before they're ready. In: **2016 IEEE 12th International Conference on e-Science (e-Science)**. [s.n.], 2016. p. 155–164. Acessado em 23 nov. 2017. Disponível em: <ieeexplore.ieee.org/document/7870896>. Citado na página 74.

HAZEKAMP, N.; KREMER-HERMAN, N.; TOVAR, B.; MENG, H.; CHOUDHURY, O.; EMRICH, S.; THAIN, D. Combining static and dynamic storage management for data intensive scientific workflows. **IEEE Transactions on Parallel and Distributed Systems**, PP, n. 99, p. 1–1, 2017. ISSN 1045-9219. Acessado em 19 nov. 2017. Disponível em: <ieeexplore.ieee.org/document/8078264/>. Citado nas páginas 74 e 75.

HAZEKAMP, N.; SARRO, J.; CHOUDHURY, O.; GESING, S.; EMRICH, S.; THAIN, D. Scaling up bioinformatics workflows with dynamic job expansion: A case study using galaxy and makeflow. In: **2015 IEEE 11th International Conference on e-Science**. [s.n.], 2015. p. 332–341. Acessado em 23 nov. 2017. Disponível em: <ieeexplore.ieee.org/document/7304316/>. Citado na página 74.

HE, H. A resource reservation based framework for qos-aware resource provision in cloud computing. v. 9, p. 193–204, 09 2016. Acessado em 14 out. 2017. Disponível em: <www.sersc.org/journals/IJGDC/vol9_no9/17.pdf>. Citado na página 72.

HETTNE, K.; DHARURI, H.; ZHAO, J.; WOLSTENCROFT, K.; BELHAJJAME, K.; SOILAND-REYES, S.; MINA, E.; THOMPSON, M.; CRUICKSHANK, D.; VERDESMONTENEGRO, L.; GARRIDO, J.; ROURE, D. de; CORCHO, O.; KLYNE, G.; SCHOUWEN, R. van; HOEN, P. A. 't; BECHHOFFER, S.; GOBLE, C.; ROOS, M. Structuring research methods and data with the research object model: genomics workflows as a case study. **Journal of Biomedical Semantics**, v. 5, n. 1, p. 41, 2014. ISSN 2041-1480. Acessado em 26 mar. 2015. Disponível em: <<http://jbiomedsem.biomedcentral.com/articles/10.1186/2041-1480-5-41>>. Citado nas páginas 50 e 72.

HIDEN, H.; WOODMAN, S.; WATSON, P. Prediction of workflow execution time using provenance traces: Practical applications in medical data processing. In: **2016 IEEE 12th International Conference on e-Science (e-Science)**. [s.n.], 2016. p. 21–30. Acessado em 23 nov. 2017. Disponível em: <ieeexplore.ieee.org/document/7870882/>. Citado nas páginas 73 e 75.

HOLL, S.; GARIJO, D.; BELHAJJAME, K.; ZIMMERMANN, O.; GIOVANNI, R. D.; OBST, M.; GOBLE, C. On specifying and sharing scientific workflow optimization results using research objects. In: **Proceedings of the 8th Workshop on Workflows in Support of Large-Scale Science**. New York, NY, USA: ACM, 2013. (WORKS '13), p. 28–37. ISBN 978-1-4503-2502-8. Acessado em 06 mar. 2015. Disponível em: <<http://dl.acm.org/citation.cfm?id=2534251>>. Citado na página 72.

HUANG, Y. P. The service oriented architecture in decision support system. In: **2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)**. [s.n.], 2016. p. 868–872. Acessado em 16 out. 2017. Disponível em: <ieeexplore.ieee.org/document/7603291/>. Citado nas páginas 30 e 77.

HUIYANG, X.; MEINA, S.; JUNDE, S. A new service integration system for modern service industry based on soa. In: **Wireless, Mobile and Sensor Networks, 2007. (CCWMSN07). IET Conference on**. [s.n.], 2007. p. 398–401. ISSN 0537-9989. Acessado em 2 mar. 2014. Disponível em: <<http://goo.gl/psl09l>>. Citado na página 76.

HUMPHREY, M.; STEELE, J.; KIM, I. K.; KAHN, M. G.; BONDY, J.; AMES, M. Clouddrn: A lightweight, end-to-end system for sharing distributed research data in the cloud. In: **2013 IEEE 9th International Conference on e-Science**. [s.n.], 2013. p. 254–261. Acessado em 28 mar. 2015. Disponível em: <ieeexplore.ieee.org/document/6683915/>. Citado na página 73.

HUNTER, J.; LEAR, R. **JDOM**. 2014. [Http://www.jdom.org/index.html](http://www.jdom.org/index.html). Citado na página 129.

IBRAHIM, R.; MOHAMMED, Y. O. Comparison of different software architectures for data integration. **Journal of Information Systems Research and Innovation**, Information Service Systems Innovation (ISSI) Research Group, Universiti Teknologi Malaysia, v. 1, p. 68–77, August 2012. Acessado em 2 mar. 2014. Disponível em: <<http://eprints.utm.my/31627/>>. Citado na página 76.

INDHUMATHI, V.; NASIRA, G. M. Service oriented architecture for load balancing with fault tolerant in grid computing. In: **2016 IEEE International Conference on Advances in Computer Applications (ICACA)**. [s.n.], 2016. p. 313–317. Acessado em 23 nov. 2017. Disponível em: <ieeexplore.ieee.org/document/7887972/>. Citado nas páginas 78 e 80.

ISLAM, M.; CHETTY, M. Clustered memetic algorithm with local heuristics for ab initio protein structure prediction. **Evolutionary Computation, IEEE Transactions on**, v. 17, n. 4, p. 558–576, 2013. ISSN 1089-778X. Acessado em 2 ago. 2014. Disponível em: <<https://www.deepdyve.com/lp/institute-of-electrical-and-electronics-engineers/clustered-memetic-algorithm-with-local-heuristics-for-ab-initio-lJsbZd9TzL>>. Citado na página 37.

JAIN, R. **Search Tips Advanced Search Art of Computer Systems Performance Analysis Techniques For Experimental Design Measurements Simulation And Modeling**. [S.l.]: Wiley Computer Publishing, John Wiley and Sons, Inc, 1991. ISBN 0471503363. Citado na página 147.

JIAN-WEI, L.; GUANG-HUI, C.; HAI-EN, L.; YUAN, L.; XIONG-LIN, L. Prediction of protein secondary structure using multilayer feed-forward neural networks. In: **Control and Decision Conference (CCDC), 2013 25th Chinese**. [s.n.], 2013. p. 1346–1351. Acessado em 3 ago. 2014. Disponível em: <<https://goo.gl/YXWs4A>>. Citado nas páginas 41 e 66.

JONES, E.; OLIPHANT, T.; PETERSON, P. *et al.* **SciPy: Open source scientific tools for Python**. 2001. Acessado em 15 out. 2017. Disponível em: <<http://www.scipy.org/>>. Citado na página 110.

JRAD, F.; TAO, J.; STREIT, A. A broker-based framework for multi-cloud workflows. In: **Proceedings of the 2013 International Workshop on Multi-cloud Applications and Federated Clouds**. New York, NY, USA: ACM, 2013. (MultiCloud '13), p. 61–68. ISBN 978-1-4503-2050-4. Acessado em 18 nov. 2017. Disponível em: <<http://doi.acm.org/10.1145/2462326.2462339>>. Citado na página 54.

JUVE, G.; CHERVENAK, A.; DEELMAN, E.; BHARATHI, S.; MEHTA, G.; VAHI, K. Characterizing and profiling scientific workflows. **Future Generation Computer Systems**, v. 29, n. 3, p. 682 – 692, 2013. ISSN 0167-739X. Acessado em 14 set. 2015. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X12001732>>. Citado na página 53.

KACSUK, P. P-grade portal family for grid infrastructures. **Concurr. Comput. : Pract. Exper.**, John Wiley and Sons Ltd., Chichester, UK, v. 23, n. 3, p. 235–245, mar. 2011. ISSN 1532-0626. Acessado em 15 out. 2015. Disponível em: <<http://dx.doi.org.ez67.periodicos.capes.gov.br/10.1002/cpe.1654>>. Citado nas páginas 55, 57 e 70.

KACSUK, P.; KISS, T.; SIPOS, G. Solving the grid interoperability problem by p-grade portal at workflow level. **Future Generation Computer Systems**, v. 24, n. 7, p. 744 – 751, 2008. ISSN 0167-739X. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X08000113>>. Citado na página 58.

KAIL, E.; KACSUK, P.; KOZLOVSZKY, M. A novel approach to user-steering in scientific workflows. In: **2015 IEEE 10th Jubilee International Symposium on Applied Computational Intelligence and Informatics**. [s.n.], 2015. p. 233–236. Acessado em 15 out. 2017. Disponível em: <ieeexplore.ieee.org/document/7208205/>. Citado na página 100.

KALEGARI, D.; LOPES, H. An improved parallel differential evolution approach for protein structure prediction using both 2d and 3d off-lattice models. In: **Differential Evolution (SDE), 2013 IEEE Symposium on**. [s.n.], 2013. p. 143–150. Acessado em 28 jun. 2014. Disponível em: <<http://bioserver.cpgei.ct.utfpr.edu.br/hslopes/publicacoes/2013/ieeescci2013-2.pdf>>. Citado nas páginas 27, 37 e 38.

KAROCZKAI, K.; KERTESZ, A.; KACSUK, P. Brokering solution for science gateways using multiple distributed computing infrastructures. In: **2015 7th International Workshop on Science Gateways**. [s.n.], 2015. p. 28–33. Acessado em 14 ago. 2017. Disponível em: [<ieeexplore.ieee.org/document/7217925/>](http://ieeexplore.ieee.org/document/7217925/). Citado na página 58.

_____. A meta-brokering framework for science gateways. **Journal of Grid Computing**, v. 14, n. 4, p. 687–703, Dec 2016. ISSN 1572-9184. Acessado em 14 ago. 2017. Disponível em: <https://doi.org/10.1007/s10723-016-9378-7>. Citado nas páginas 79 e 80.

KAVITHA, K.; SARITHA, R.; VINOD, C. Computational prediction of continuous b-cell epitopes using random forest classifier. In: **Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on**. [s.n.], 2013. p. 1–5. Acessado em 19 nov. 2014. Disponível em: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=6726820>. Citado nas páginas 66 e 69.

KAZEMIAN, H.; YUSUF, S. An anfis approach to transmembrane protein prediction. In: **Fuzzy Systems (FUZZ-IEEE), 2014 IEEE International Conference on**. [s.n.], 2014. p. 1360–1365. Acessado em 13 jan. 2015. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6891769. Citado na página 66.

KELSEY, T.; MCCAFFERY, M.; KOTTHOFF, L. Web-scale distributed science ai search across disconnected and heterogeneous infrastructures. In: **2014 IEEE 10th International Conference on e-Science**. [s.n.], 2014. v. 1, p. 39–46. Acessado em 28 mar. 2015. Disponível em: <http://ieeexplore.ieee.org/document/6972247/>. Citado na página 73.

KHAN, S.; SHAKIL, K. A.; ALAM, M. Workflow-based big data analytics in the cloud environment present research status and future prospects. **CoRR**, abs/1711.02087, 2017. Acessado em 19 nov. 2017. Disponível em: <http://arxiv.org/abs/1711.02087>. Citado na página 70.

KHATIB, F.; DIMAIO, F.; COOPER, S.; KAZMIERCZYK, M.; GILSKI, M.; KRZYWDA, S.; ZABRANSKA, H.; PICHOVA, I.; THOMPSON, J.; POPOVIC, Z.; JASKOLSKI, M.; BAKER, D. Crystal structure of a monomeric retroviral protease solved by protein folding game players. **Nat Struct Mol Biol**, 2011. Acessado em 16 jul. 2014. Disponível em: <http://www.nature.com/nsmb/journal/v18/n10/full/nsmb.2119.html>. Citado na página 42.

KHOJA, R.; MAROLIA, M.; ACHARYA, T.; CHAKRABARTI, C. A coprocessor architecture for fast protein structure prediction. **Pattern Recognition**, v. 39, n. 12, p. 2494 – 2505, 2006. ISSN 0031-3203. Acessado em 23 nov. 2015. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0031320305004802>. Citado na página 68.

KIMOVSKI, D.; SAURABH, N.; GEC, S.; ŠTEFANIČ, P.; KECSKEMETI, G.; STANKOVSKI, V.; PRODAN, R.; FAHRINGER, T. Towards an environment for efficient and transparent virtual machine operations: The entice approach. In: **2016 5th IEEE International Conference on Cloud Networking (Cloudnet)**. [s.n.], 2016. p. 242–247. Acessado em 13 fev. 2018. Disponível em: <http://ieeexplore.ieee.org/abstract/document/7776611/>. Citado na página 30.

KOOHI, A. Prediction of drug-target interactions using popular collaborative filtering methods. In: **Genomic Signal Processing and Statistics (GENSIPS), 2013 IEEE International Workshop on**. [s.n.], 2013. p. 58–61. Acessado em 13 ago. 2014. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6735931. Citado na página 66.

KOVÁCS, J.; KACSUK, P.; LOMAKA, A. Using a private desktop grid system for accelerating drug discovery. **Future Generation Computer Systems**, v. 27, n. 6, p. 657 – 666, 2011. ISSN 0167-739X. Acessado em 18 jul. 2015. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X10002578>. Citado nas páginas 57 e 72.

KOZLOVSZKY, M.; KARÓCZKAI, K.; MÁRTON, I.; KACSUK, P.; GOTTDANK, T. Dci bridge: Executing ws-pgrade workflows in distributed computing infrastructures. In: _____. **Science Gateways for Distributed Computing Infrastructures: Development Framework and Exploitation by Scientific User Communities**. Cham: Springer International Publishing, 2014. p. 51–67. ISBN 978-3-319-11268-8. Acessado em 14 out. 2017. Disponível em: https://doi.org/10.1007/978-3-319-11268-8_4. Citado nas páginas 72, 75 e 100.

KRAFZIG, D.; BANKE, K.; SLAMA, D. **Enterprise SOA: Service-Oriented Architecture Best Practices (The Coad Series)**. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2004. ISBN 0131465759. Citado na página 44.

KRIEGER, M. T.; TORRENO, O.; TRELLES, O.; KRANZLMÜLLER, D. Building an open source cloud environment with auto-scaling resources for executing bioinformatics and biomedical workflows. **Future Generation Computer Systems**, v. 67, p. 329 – 340, 2017. ISSN 0167-739X. Acessado em 18 out. 2017. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X16300218>. Citado nas páginas 78 e 80.

KRISHNAJI, A.; RAO, A. A. An improved hybrid neuro fuzzy genetic system (i-hnfgs) for protein secondary structure prediction from amino acid sequence. In: **Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference on**. [s.n.], 2013. p. 1218–1223. Acessado em 28 jun. 2014. Disponível em: <http://goo.gl/5ojktQ>. Citado nas páginas 65 e 69.

KRÜGER, J.; GRUNZKE, R.; GESING, S.; BREUERS, S.; BRINKMANN, A.; GARZA, L. de la; KOHLBACHER, O.; KRUSE, M.; NAGEL, W. E.; PACKSCHIES, L.; MÜLLER-PFEFFERKORN, R.; SCHÄFER, P.; SCHÄRFE, C.; STEINKE, T.; SCHLEMMER, T.; WARZECHA, K. D.; ZINK, A.; HERRES-PAWLIS, S. The mosgrid science gateway - a complete solution for molecular simulations. **Journal of Chemical Theory and Computation**, American Chemical Society, v. 10, n. 6, p. 2232–2245, Jun 2014. ISSN 1549-9618. Acessado em 23 nov. 2015. Disponível em: <https://doi.org/10.1021/ct500159h>. Citado nas páginas 58 e 72.

KSHETRI, N.; MURUGESAN, S. Cloud Computing and EU Data Privacy Regulations. **Computer Society**, n. 3, p. 86–89, mar. 2013. ISSN 0018-9162. Acessado em 15 mar. 2014. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6489955>. Citado na página 30.

KUMARI, G.; KANDAN, B.; MISHRA, A. Experience sharing on soa based heterogeneous systems integration. In: **Services - Part I, 2008. IEEE Congress on**. [s.n.], 2008. p. 107–108. Acessado em 2 mar. 2014. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4578311>. Citado na página 76.

KUMRAI, T.; OTA, K.; DONG, M.; KISHIGAMI, J.; SUNG, D. K. Multiobjective optimization in cloud brokering systems for connected internet of things. **IEEE Internet of Things Journal**, v. 4, n. 2, p. 404–413, April 2017. ISSN 2327-4662. Acessado em 15 nov. 2017. Disponível em: <http://ieeexplore.ieee.org/document/7467407/>. Citado na página 43.

KUNHI, L. M.; RAJU, K.; CHIPLUNKAR, N. N. Performance analysis of protein structure clustering techniques and cuda implementation of rmsd computation. In: **2016 IEEE Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)**. [s.n.], 2016. p. 18–21. Acessado em 15 fev. 2017. Disponível em: [<ieeexplore.ieee.org/document/7806220/>](http://ieeexplore.ieee.org/document/7806220/). Citado nas páginas 27, 68 e 69.

KUO, T. H.; CHEN, C. H.; KUNG, H. Y.; LIAO, Y. S. Applications of the web service middleware framework based on the bpel. In: **2016 IEEE 5th Global Conference on Consumer Electronics**. [s.n.], 2016. p. 1–5. Acessado em 28 fev. 2017. Disponível em: [<ieeexplore.ieee.org/document/7800522/>](http://ieeexplore.ieee.org/document/7800522/). Citado na página 77.

KUROWSKI, K.; DZIUBECKI, P.; GRABOWSKI, P.; KRYSIŃSKI, M.; PIONTEK, T.; SZEJNFELD, D. Easy development and integration of science gateways with vine toolkit. In: _____. **eScience on Distributed Computing Infrastructure: Achievements of PL-Grid Plus Domain-Specific Services and Tools**. Cham: Springer International Publishing, 2014. p. 147–163. ISBN 978-3-319-10894-0. Acessado em 18 jul. 2015. Disponível em: https://doi.org/10.1007/978-3-319-10894-0_11. Citado nas páginas 56 e 57.

LEHNINGER, A.; NELSON, D. L.; COX, M. M. **Lehninger Principles of Biochemistry**. Fifth edition. [S.l.]: W. H. Freeman, 2008. Hardcover. Citado na página 39.

LI, D.; TSUI, B.; XUE, C.; HAGA, J. H.; ICHIKAWA, K.; DATE, S. Protein structure modeling in a grid computing environment. In: **2013 IEEE 9th International Conference on e-Science**. [s.n.], 2013. p. 301–306. Acessado em 14 out. 2016. Disponível em: [<ieeexplore.ieee.org/document/6683921/>](http://ieeexplore.ieee.org/document/6683921/). Citado na página 73.

LI, H.; LIU, C. Prediction of protein structures using a map-reduce hadoop framework based simulated annealing algorithm. In: **2013 IEEE International Conference on Bioinformatics and Biomedicine**. [s.n.], 2013. p. 6–10. Acessado em 23 nov. 2015. Disponível em: [<ieeexplore.ieee.org/document/6732710/>](http://ieeexplore.ieee.org/document/6732710/). Citado nas páginas 68 e 69.

LI, J.; SONG, B. Web services integration on data mining based on soa. In: **Intelligence Information Processing and Trusted Computing (IPTC), 2010 International Symposium on**. [s.n.], 2010. p. 532–534. Acessado em 26 fev. 2014. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5663638>. Citado nas páginas 76 e 117.

LIN, D.-C.; MAYAKONDA, A.; DINH, H. Q.; HUANG, P.; LIN, L.; LIU, X.; DING, L.-W.; WANG, J.; BERMAN, B.; SONG, E.; YIN, D.; KOEFFLER, H. P. Genomic and epigenomic heterogeneity of hepatocellular carcinoma. **Cancer Research**, American Association for Cancer Research, 2017. ISSN 0008-5472. Acessado em 16 out. 2017. Disponível em: <http://cancerres.aacrjournals.org/content/early/2017/02/18/0008-5472.CAN-16-2822>. Citado nas páginas 51 e 52.

LINDEMANN, J.; SANDBERG, G. An extendable grid application portal. In: _____. **Advances in Grid Computing - EGC 2005: European Grid Conference, Amsterdam, The Netherlands, February 14-16, 2005, Revised Selected Papers**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 1012–1021. ISBN 978-3-540-32036-4. Acessado em 18 jul. 2015. Disponível em: https://doi.org/10.1007/11508380_103. Citado na página 58.

LINDERMAN, M. D.; BRUGGNER, R.; ATHALYE, V.; MENG, T. H.; ASADI, N. B.; NOLAN, G. P. High-throughput bayesian network learning using heterogeneous multicore computers. In: **Proceedings of the 24th ACM International Conference on Supercomputing**. New York,

- NY, USA: ACM, 2010. (ICS '10), p. 95–104. ISBN 978-1-4503-0018-6. Acessado em 14 abr. 2014. Disponível em: <<http://doi.acm.org/10.1145/1810085.1810101>>. Citado na página 121.
- LIU, B.; MADDURI, R. K.; SOTOMAYOR, B.; CHARD, K.; LACINSKI, L.; DAVE, U. J.; LI, J.; LIU, C.; FOSTER, I. T. Cloud-based bioinformatics workflow platform for large-scale next-generation sequencing analyses. **Journal of Biomedical Informatics**, v. 49, p. 119 – 133, 2014. ISSN 1532-0464. Acessado em 18 jul. 2015. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1532046414000070>>. Citado na página 74.
- LIU, B.; SOTOMAYOR, B.; MADDURI, R.; CHARD, K.; FOSTER, I. Deploying bioinformatics workflows on clouds with galaxy and globus provision. In: **2012 SC Companion: High Performance Computing, Networking Storage and Analysis**. [S.l.: s.n.], 2012. p. 1087–1095. Acessado em 14 set. 2015. Citado nas páginas 52 e 53.
- LIU, H.; LIN, F.; QIAN, K.; SEAH, H. S.; LEE, Y. T. Visual analysis with dynamic geometric complementarity and physicochemical matching in protein docking. In: **Information Visualisation (IV), 2014 18th International Conference on**. [s.n.], 2014. p. 315–320. ISSN 1550-6037. Acessado em 13 jan. 2015. Disponível em: <<http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6902922>>. Citado na página 66.
- LIU, J.; PACITTI, E.; VALDURIEZ, P.; MATTOSO, M. A survey of data-intensive scientific workflow management. **Journal of Grid Computing**, v. 13, n. 4, p. 457–493, Dec 2015. ISSN 1572-9184. Acessado em 11 jan. 2018. Disponível em: <<https://doi.org/10.1007/s10723-015-9329-8>>. Citado nas páginas 51, 69, 71 e 120.
- LIU, L.; YANG, Y.; LI, L. Glue the gap between applications and resources in e-science. In: **2008 First IEEE International Conference on Ubi-Media Computing**. [s.n.], 2008. p. 284–289. Acessado em 14 jun. 2016. Disponível em: <ieeexplore.ieee.org/document/4570904/>. Citado nas páginas 57, 75 e 80.
- LIU, Y.; GORTON, I.; WYNNE, A. Architecture-based adaptivity support for service oriented scientific workflows. In: **2013 IEEE Seventh International Symposium on Service-Oriented System Engineering**. [s.n.], 2013. p. 309–314. Acessado em 28 fev. 2014. Disponível em: <ieeexplore.ieee.org/document/6525537/>. Citado na página 75.
- LONG, W.; YUQING, L.; QINGXIN, X. Using cloudsim to model and simulate cloud computing environment. In: **Computational Intelligence and Security (CIS), 2013 9th International Conference on**. [s.n.], 2013. p. 323–328. Acessado em 16 jan. 2015. Disponível em: <<http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6746411>>. Citado na página 60.
- LU, X.; KOGA, T. Service oriented architecture for global system wide information management. In: **2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)**. [s.n.], 2015. p. 638–639. Acessado em 26 maio 2016. Disponível em: <ieeexplore.ieee.org/document/7398585/>. Citado nas páginas 30, 76 e 80.
- LU, Y.; YANG, L.; YUAN, H.; WANG, Y.; LUO, H.; TANG, Y. Y. A novel method for protein structure retrieval using tableau representation and sparse coding. In: **Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on**. [s.n.], 2014. p. 4042–4046. Acessado em 13 abr. 2015. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6974565&tag=1>. Citado nas páginas 67, 69 e 92.

LUCENA, D. de; LIMA, T. Woerle de; SOARES, A. da S.; DELBEM, A.; FILHO, A. R. G.; COELHO, C.; LAUREANO, G. Multi-objective evolutionary algorithm for variable selection in calibration problems: A case study for protein concentration prediction. In: **Evolutionary Computation (CEC), 2013 IEEE Congress on**. [s.n.], 2013. p. 1053–1059. Acessado em 16 jan. 2014. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6557683>. Citado nas páginas 67 e 91.

LUDÄSCHER, B.; ALTINTAS, I.; BERKLEY, C.; HIGGINS, D.; JAEGER, E.; JONES, M.; LEE, E. A.; TAO, J.; ZHAO, Y. Scientific workflow management and the kepler system: Research articles. **Concurr. Comput. : Pract. Exper.**, John Wiley and Sons Ltd., Chichester, UK, v. 18, n. 10, p. 1039–1065, ago. 2006. ISSN 1532-0626. Acessado em 13 ago. 2016. Disponível em: <<http://dx.doi.org/10.1002/cpe.v18:10>>. Citado na página 50.

LYASHOV, M. V.; BEREZA, A. N.; BABAEV, A. M.; ALEKSEENKO, J. V.; NAZVANTSEV, D. S. The technique of creating distributed computing systems based on service-oriented architecture. In: **2016 IEEE 10th International Conference on Application of Information and Communication Technologies (AICT)**. [s.n.], 2016. p. 1–5. Acessado em 23 nov. 2017. Disponível em: <ieeexplore.ieee.org/abstract/document/7991710/>. Citado nas páginas 78 e 80.

MACEDO, E. de A.; MELO, A. C. Magalhaes Alves de; PFITSCHER, G. H.; BOUKER-CHE, A. Multiple biological sequence alignment in heterogeneous multicore clusters with user-selectable task allocation policies. **The Journal of Supercomputing**, v. 63, n. 3, p. 740–756, 2013. ISSN 1573-0484. Acessado em 13 ago. 2016. Disponível em: <<http://dx.doi.org/10.1007/s11227-012-0768-8>>. Citado nas páginas 30 e 78.

MADDURI, R.; CHARD, K.; CHARD, R.; LACINSKI, L.; RODRIGUEZ, A.; SULAKHE, D.; KELLY, D.; DAVE, U.; FOSTER, I. The globus galaxies platform: delivering science gateways as a service. **Concurrency and Computation: Practice and Experience**, p. n/a, abr. 2015. ISSN 15320626. Acessado em 18 jul. 2016. Disponível em: <<http://dx.doi.org/10.1002/cpe.3486>>. Citado nas páginas 55, 56 e 104.

MAHESHWARI, K.; JUNG, E. S.; MENG, J.; VISHWANATH, V.; KETTIMUTHU, R. Model-driven multisite workflow scheduling. In: **2013 IEEE International Conference on Cluster Computing (CLUSTER)**. [s.n.], 2013. p. 1–5. ISSN 1552-5244. Acessado em 22 set. 2014. Disponível em: <<http://ieeexplore.ieee.org.ez67.periodicos.capes.gov.br/document/6702647/>>. Citado na página 54.

MAHESHWARI, K.; JUNG, E.-S.; MENG, J.; MOROZOV, V.; VISHWANATH, V.; KETTIMUTHU, R. Workflow performance improvement using model-based scheduling over multiple clusters and clouds. **Future Generation Computer Systems**, v. 54, p. 206 – 218, 2016. ISSN 0167-739X. Acessado em 22 set. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X15000795>>. Citado na página 79.

MAHESHWARI, K.; RODRIGUEZ, A.; KELLY, D.; MADDURI, R.; WOZNIAC, J.; WILDE, M.; FOSTER, I. Enabling multi-task computation on galaxy-based gateways using swift. In: **2013 IEEE International Conference on Cluster Computing (CLUSTER)**. [s.n.], 2013. p. 1–3. ISSN 1552-5244. Acessado em 22 set. 2014. Disponível em: <<http://ieeexplore.ieee.org.ez67.periodicos.capes.gov.br/document/6702701/?reload=true>>. Citado na página 79.

MALARVIZHI, R.; KALYANI, S. Soa based open data model for information integration in smart grid. In: **Advanced Computing (ICoAC), 2013 Fifth International Conference on**.

- [s.n.], 2013. p. 143–148. Acessado em 07 fev. 2014. Disponível em: <<http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=6921941>>. Citado nas páginas 44 e 76.
- MALIK, S.; SAINI, P.; RANI, S. Energy efficient resource allocation for heterogeneous workload in cloud computing. In: _____. **Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications : FICTA 2016, Volume 1**. Singapore: Springer Singapore, 2017. p. 89–97. ISBN 978-981-10-3153-3. Acessado em 23 nov. 2017. Disponível em: <https://doi.org/10.1007/978-981-10-3153-3_9>. Citado na página 73.
- MANDAL, A.; DAS, I.; BHATTACHARJEE, D. A software tool for extraction of annotation data from a pdb file. In: **Emerging Trends and Applications in Computer Science (NCE-TACS), 2012 3rd National Conference on**. [s.n.], 2012. p. 31–37. Acessado em 09 nov. 2013. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6203293>. Citado nas páginas 41, 67 e 92.
- MANSOUR, H.; DILLON, T. Dependability and rollback recovery for composite web services. **Services Computing, IEEE Transactions on**, v. 4, n. 4, p. 328–339, 2011. ISSN 1939-1374. Acessado em 22 set. 2014. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5467018>. Citado na página 44.
- MARINHO, A.; MURTA, L.; WERNER, C.; BRAGANHOLO, V.; CRUZ, S. M. S. d.; OGASAWARA, E.; MATTOSO, M. Provmanager: a provenance management system for scientific workflows. **Concurrency and Computation: Practice and Experience**, John Wiley and Sons, Ltd, v. 24, n. 13, p. 1513–1530, 2012. ISSN 1532-0634. Acessado em 16 out. 2017. Disponível em: <<http://dx.doi.org/10.1002/cpe.1870>>. Citado na página 71.
- MARINHO, A.; OLIVEIRA, D. de; OGASAWARA, E.; SILVA, V.; OCAÑA, K.; MURTA, L.; BRAGANHOLO, V.; MATTOSO, M. Deriving scientific workflows from algebraic experiment lines: A practical approach. **Future Generation Computer Systems**, v. 68, p. 111 – 127, 2017. ISSN 0167-739X. Acessado em 14 nov. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X1630276X>>. Citado na página 72.
- MAROSI, A.; KOVÁCS, J.; KACSUK, P. Towards a volunteer cloud system. **Future Generation Computer Systems**, v. 29, n. 6, p. 1442 – 1451, 2013. ISSN 0167-739X. Acessado em 18 jul. 2015. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X12000660>>. Citado na página 57.
- MARQUEZ-CHAMORRO, A. E.; ASECIO-CORTES, G.; SANTISTEBAN-TOCA, C. E.; AGUILAR-RUIZ, J. S. Soft computing methods for the prediction of protein tertiary structures: A survey. **Applied Soft Computing**, v. 35, p. 398 – 410, 2015. ISSN 1568-4946. Acessado em 12 dez. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1568494615003737>>. Citado nas páginas 68 e 69.
- MATH, C. **Apache Commons Math**. 2016. [Http://commons.apache.org/proper/commons-math/](http://commons.apache.org/proper/commons-math/). Citado na página 129.
- MATSUNAGA, A.; THOMPSON, A.; FIGUEIREDO, R. J.; GERMAIN-AUBREY, C. C.; COLLINS, M.; BEAMAN, R. S.; MACFADDEN, B. J.; RICCARDI, G.; SOLTIS, P. S.; PAGE, L. M.; FORTES, J. A. B. A computational- and storage-cloud for integration of biodiversity collections. In: **2013 IEEE 9th International Conference on e-Science**. [s.n.], 2013. p. 78–87. Acessado em 14 out. 2016. Disponível em: <ieeexplore.ieee.org/document/6683894/>. Citado na página 69.

MATTOSO, M.; DIAS, J.; OCAÑA, K. A.; OGASAWARA, E.; COSTA, F.; HORTA, F.; SILVA, V.; OLIVEIRA, D. de. Dynamic steering of hpc scientific workflows: A survey. **Future Generation Computer Systems**, v. 46, p. 100 – 113, 2015. ISSN 0167-739X. Acessado em 16 out. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X14002519>>. Citado nas páginas 72, 75, 92 e 100.

MATTOSO, M.; nA, K. O.; HORTA, F.; DIAS, J.; OGASAWARA, E.; SILVA, V.; OLIVEIRA, D. de; COSTA, F.; ARAÚJO, I. User-steering of hpc workflows: State-of-the-art and future directions. In: **Proceedings of the 2Nd ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies**. New York, NY, USA: ACM, 2013. (SWEET '13), p. 4:1–4:6. ISBN 978-1-4503-2349-9. Acessado em 14 abr. 2014. Disponível em: <<http://doi.acm.org/10.1145/2499896.2499900>>. Citado nas páginas 28 e 50.

MATTOSO, M.; WERNER, C.; TRAVASSOS, G. H.; BRAGANHOLO, V.; OGASAWARA, E.; OLIVEIRA, D.; CRUZ, S.; MARTINHO, W.; MURTA, L. Towards supporting the life cycle of large scale scientific experiments. **International Journal of Business Process Integration and Management**, v. 5, n. 1, p. 79–92, 2010. Acessado em 15 nov. 2014. Disponível em: <<https://www.inderscienceonline.com/doi/abs/10.1504/IJBPIIM.2010.033176>>. Citado nas páginas 50, 53 e 54.

MEIYAN, W. Analysis to the Weaknesses and Safety in Cloud Computing. **2013 Third International Conference on Intelligent System Design and Engineering Applications**, Ieee, p. 358–361, jan. 2013. Acessado em 15 mar. 2014. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6456341>>. Citado nas páginas 30 e 48.

MENON, K.; ANALA, K.; TRUPTI, S.; SOOD, N. Cloud computing: Applications in biological research and future prospects. In: **Cloud Computing Technologies, Applications and Management (ICCCTAM), 2012 International Conference on**. [s.n.], 2012. p. 102–107. Acessado em 14 out. 2014. Disponível em: <<http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6488081>>. Citado nas páginas 47, 77, 80 e 121.

MILLER, C. A.; QIAO, Y.; DISERA, T.; D'ASTOUS, B.; MARTH, G. T. bam.iobio: a web-based, real-time, sequence alignment file inspector. **Nat Meth**, Nature Publishing Group, a division of Macmillan Publishers Limited. All Rights Reserved., v. 11, n. 12, p. 1189, dez. 2014. Acessado em 19 nov. 2015. Disponível em: <<http://dx.doi.org/10.1038/nmeth.3174>>. Citado na página 29.

MISHRA, S.; RAO, C. S. P. Performance comparison of some evolutionary algorithms on job shop scheduling problems. **IOP Conference Series: Materials Science and Engineering**, v. 149, n. 1, p. 012041, 2016. Acessado em 23 nov. 2017. Disponível em: <<http://stacks.iop.org/1757-899X/149/i=1/a=012041>>. Citado na página 74.

MORENO, A. R.; TIRADO, Ó. T.; SALAZAR, O. T. Out of core computation of hsps for large biological sequences. In: _____. **Advances in Computational Intelligence: 12th International Work-Conference on Artificial Neural Networks, IWANN 2013, Puerto de la Cruz, Tenerife, Spain, June 12-14, 2013, Proceedings, Part II**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. p. 189–199. ISBN 978-3-642-38682-4. Acessado em 22 abr. 2014. Disponível em: <http://dx.doi.org/10.1007/978-3-642-38682-4_22>. Citado na página 70.

MROZEK, D.; GOSK, P.; MAŁYSIAK-MROZEK, B. Scaling ab initio predictions of 3d protein structures in microsoft azure cloud. **Journal of Grid Computing**, v. 13, n. 4, p. 561–585, 2015.

Acessado em 18 mai. 2016. Disponível em: <<http://dx.doi.org/10.1007/s10723-015-9353-8>>. Citado na página 78.

NAKANDALA, S.; PAMIDIGHANTAM, S.; YODAGE, S.; DOSHI, N.; ABEYSINGHE, E.; KANKANAMALAGE, C. P.; MARRU, S.; PIERCE, M. Anatomy of the seagrid science gateway. In: **Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale**. New York, NY, USA: ACM, 2016. (XSEDE16), p. 40:1–40:8. ISBN 978-1-4503-4755-6. Disponível em: <<http://doi.acm.org/10.1145/2949550.2949591>>. Citado na página 72.

NELSON, E. K.; PIEHLER, B.; ECKELS, J.; RAUCH, A.; BELLEW, M.; HUSSEY, P.; RAMSAY, S.; NATHE, C.; LUM, K.; KROUSE, K.; STEARNS, D.; CONNOLLY, B.; SKILLMAN, T.; IGRA, M. Labkey server: An open source platform for scientific data integration, analysis and collaboration. **BMC Bioinformatics**, v. 12, n. 1, p. 71, Mar 2011. ISSN 1471-2105. Acessado em 12 dez. 2017. Disponível em: <<https://doi.org/10.1186/1471-2105-12-71>>. Citado na página 58.

OGASAWARA, E.; DIAS, J.; SILVA, V.; CHIRIGATI, F.; OLIVEIRA, D. de; PORTO, F.; VALDURIEZ, P.; MATTOSO, M. Chiron: a parallel engine for algebraic scientific workflows. **Concurrency and Computation: Practice and Experience**, v. 25, n. 16, p. 2327–2341, 2013. ISSN 1532-0634. Acessado em 14 abr. 2014. Disponível em: <<http://dx.doi.org/10.1002/cpe.3032>>. Citado na página 28.

OKADA, K.; FLORES, L.; WONG, M.; PETKOVIC, D. Microenvironment-based protein function analysis by random forest. In: **Pattern Recognition (ICPR), 2014 22nd International Conference on**. [s.n.], 2014. p. 3138–3143. ISSN 1051-4651. Acessado em 13 jan. 2015. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6977253>. Citado na página 66.

OLABARRIAGA, S.; PIERANTONI, G.; TAFFONI, G.; SCIACCA, E.; JAGHOORI, M.; KORKHOV, V.; CASTELLI, G.; VUERLI, C.; BECCIANI, U.; CARLEY, E.; BENTLEY, B. Scientific workflow management – for whom? In: **2014 IEEE 10th International Conference on e-Science**. [s.n.], 2014. v. 1, p. 298–305. Acessado em 13 ago 2015. Disponível em: <<https://ieeexplore.ieee.org/document/6972277/>>. Citado na página 70.

OLIVEIRA, D. de; OCAÑA, K. A.; OGASAWARA, E.; DIAS, J.; GONÇALVES, J.; BAIÃO, F.; MATTOSO, M. Performance evaluation of parallel strategies in public clouds: A study with phylogenomic workflows. **Future Generation Computer Systems**, v. 29, n. 7, p. 1816 – 1825, 2013. ISSN 0167-739X. Acessado em 16 out. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X13000034>>. Citado nas páginas 72 e 75.

OLIVEIRA, D. de; OGASAWARA, E.; OCAÑA, K.; BAIÃO, F.; MATTOSO, M. An adaptive parallel execution strategy for cloud-based scientific workflows. **Concurrency and Computation: Practice and Experience**, John Wiley and Sons, Ltd, v. 24, n. 13, p. 1531–1550, 2012. ISSN 1532-0634. Acessado em 16 out. 2017. Disponível em: <<http://dx.doi.org/10.1002/cpe.1880>>. Citado na página 72.

OLIVEIRA, E. M.; ESTRELLA, J. C.; KUEHNE, B. T.; FILHO, D. M. L.; ADAMI, L. J.; NUNES, L. H.; NAKAMURA, L. H.; LIBARDI, R. M.; SOUZA, P. S. L.; REIFF-MARGANIEC, S. Design and implementation of fault tolerance techniques to improve qos in soa. In: **10th International Conference on Network and Service Management (CNSM) and Workshop**. [s.n.], 2014. p. 37–45. ISSN 2165-9605. Acessado em 13 jan. 2015. Disponível em: <ieeexplore.ieee.org/document/7014139>. Citado na página 101.

OLIVEIRA, L. L. de; OLIVEIRA, P. de; TINÓS, R. A multiobjective approach to the genetic code adaptability problem. **BMC Bioinformatics**, v. 16, p. 52:1–52:20, 2015. Disponível em: <<https://doi.org/10.1186/s12859-015-0480-9>>. Citado na página 66.

OSTBERG, P. O.; ELMROTH, E. Decentralized prioritization-based management systems for distributed computing. In: **2013 IEEE 9th International Conference on e-Science**. [s.n.], 2013. p. 228–237. Acessado em 22 mar. 2014. Disponível em: <doi.ieeecomputersociety.org/10.1109/eScience.2013.44>. Citado na página 78.

OZDOGAN, A. G.; AKGUN, K. O.; KAYA, C. Service governance in service oriented architecture. In: **2017 International Conference on Computer Science and Engineering (UBMK)**. [s.n.], 2017. p. 195–200. Acessado em 23 nov. 2017. Disponível em: <ieeexplore.ieee.org/document/8093374/>. Citado nas páginas 76 e 80.

PADMAVATHI, S.; DHARANI, V. P.; MAITHREYE, S.; DEVI, M. M.; DURAIRAJ, S. Multi-agent framework for cloud service composition. In: **2016 International Conference on Emerging Trends in Engineering, Technology and Science (ICETETS)**. [s.n.], 2016. p. 1–5. Acessado em 28 fev. 2017. Disponível em: <ieeexplore.ieee.org/document/7603011/>. Citado na página 78.

PANADERO, J.; ARMAS, J. de; SERRA, X.; MARQUES, J. M. Multi criteria biased randomized method for resource allocation in distributed systems: Application in a volunteer computing system. **Future Generation Computer Systems**, v. 82, p. 29 – 40, 2018. ISSN 0167-739X. Acessado em 15 fev. 2018. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0167739X17315236>>. Citado nas páginas 79 e 80.

PAPAZOGLU, M. Service-oriented computing: concepts, characteristics and directions. In: **Web Information Systems Engineering, 2003. WISE 2003. Proceedings of the Fourth International Conference on**. [s.n.], 2003. p. 3–12. Acessado em 22 set. 2014. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1254461>. Citado na página 44.

PARDO, M. H. S.; CENTURION, A. M.; EUSTÁQUIO, P. S. F.; SANTANA, R. H. C.; BRUSCHI, S. M.; SANTANA, M. J. Evaluating the influence of the client behavior in cloud computing. **PLOS ONE**, Public Library of Science, v. 11, n. 7, p. 1–28, 07 2016. Acessado em 14 dez. 2017. Disponível em: <<https://doi.org/10.1371/journal.pone.0158291>>. Citado na página 129.

PATEL, S.; PANCHAL, H.; ANJARIA, K. Dna sequence analysis by orf finder amp; genomatix tool: Bioinformatics analysis of some tree species of leguminosae family. In: **2012 IEEE International Conference on Bioinformatics and Biomedicine Workshops**. [s.n.], 2012. p. 922–926. Acessado em 13 out. 2014. Disponível em: <ieeexplore.ieee.org/document/6470265/>. Citado na página 58.

PATTERSON, E.; MCBURNEY, R.; SCHMIDT, H.; BALDINI, I.; MOJSILOVIĆ, A.; VARSHNEY, K. R. Dataflow representation of data analyses: Toward a platform for collaborative data science. **IBM Journal of Research and Development**, v. 61, n. 6, p. 9:1–9:13, Nov 2017. ISSN 0018-8646. Acessado em 15 fev. 2018. Disponível em: <ieeexplore.ieee.org/abstract/document/8167722/>. Citado na página 73.

PAULSON, L. D. Mainframes, cobol still popular. **IT Professional**, v. 3, n. 5, p. 12–14, Sep 2001. ISSN 1520-9202. Acessado em 18 mai. 2017. Disponível em: <ieeexplore.ieee.org/document/952975/>. Citado na página 47.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011. Acessado em 15 out. 2017. Disponível em: <www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf>. Citado na página 109.

PENG, G.; WANG, H.; ZHANG, H.; DONG, J. Knowledge-based resource allocation for collaborative simulation development in a multi-tenant cloud computing environment. **IEEE Transactions on Services Computing**, Institute of Electrical and Electronics Engineers Inc., 2016. ISSN 1939-1374. Acessado em 15 jul. 2017. Disponível em: <<https://www.computer.org/csdl/trans/sc/preprint/07383312.pdf>>. Citado na página 78.

PERERA, S.; MARRU, S.; GANNON, D. Monitoring and managing e-science cyberinfrastructures: A case study. In: **2008 IEEE Fourth International Conference on eScience**. [s.n.], 2008. p. 402–403. Acessado em 15 out. 2015. Disponível em: <ieeexplore.ieee.org/document/4736817/>. Citado nas páginas 57 e 101.

PHUOC, N.; KIM, S.-R. Protein fold prediction using cluster merging. In: **Computer Sciences and Convergence Information Technology (ICCIT), 2011 6th International Conference on**. [s.n.], 2011. p. 293–298. Acessado em 25 abr. 2014. Disponível em: <<http://ieeexplore.ieee.org/document/6316623/>>. Citado nas páginas 41, 67 e 118.

PIERANTONI, G.; FROST, D.; GESING, S.; OLABARRIAGA, S.; JAGHOORI, M.; TERSTYANSZKY, G.; ARSHAD, J. A model for information and action flows connecting science gateways to distributed computing infrastructures. 06 2017. Acessado em 14 out. 2017. Disponível em: <ceur-ws.org/Vol-1871/paper1.pdf>. Citado na página 72.

PLANKENSTEINER, K.; PRODAN, R.; JANETSCHEK, M.; FAHRINGER, T.; MONTAGNAT, J.; ROGERS, D.; HARVEY, I.; TAYLOR, I.; BALASKÓ, Á.; KACSUK, P. Fine-grain interoperability of scientific workflows in distributed computing infrastructures. **Journal of Grid Computing**, v. 11, n. 3, p. 429–455, Sep 2013. ISSN 1572-9184. Acessado em 14 abr. 2014. Disponível em: <<https://doi.org/10.1007/s10723-013-9261-8>>. Citado na página 119.

PODZIMEK, A.; BULEJ, L.; CHEN, L. Y.; BINDER, W.; TÛMA, P. Robust partial-load experiments with showstopper. **Future Generation Computer Systems**, v. 64, n. Supplement C, p. 15 – 38, 2016. ISSN 0167-739X. Acessado em 23 nov. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X16301078>>. Citado na página 74.

PONMARY, P. L. D.; RAJ, D.; SHARMILA, D. Generation of unified data structure and data warehouse for protein data banks. In: **Conference on Computational Intelligence and Multimedia Applications, 2007. International Conference on**. [s.n.], 2007. v. 2, p. 3–7. Acessado em 09 fev. 2015. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4426660>. Citado nas páginas 41 e 67.

POUCHARD, L.; MALIK, A.; DAM, H. V.; XIE, C.; XU, W.; DAM, K. K. V. Capturing provenance as a diagnostic tool for workflow performance evaluation and optimization. In: **2017 New York Scientific Data Summit (NYSDS)**. [s.n.], 2017. p. 1–8. Acessado em 15 dez. 2017. Disponível em: <ieeexplore.ieee.org/abstract/document/8085043>. Citado nas páginas 54 e 101.

PRASETYO, N. A.; BANDUNG, Y. A design of software requirement engineering framework based on knowledge management and service-oriented architecture decision (soad) modeling framework. In: **2015 International Conference on Information Technology Systems and Innovation (ICITSI)**. [s.n.], 2015. p. 1–6. Acessado em 15 out. 2017. Disponível em: [<ieeexplore.ieee.org/document/7437708/>](http://ieeexplore.ieee.org/document/7437708/). Citado na página 76.

PRODAN, R.; WIECZOREK, M.; FARD, H. M. Double auction-based scheduling of scientific applications in distributed grid and cloud environments. **Journal of Grid Computing**, v. 9, n. 4, p. 531–548, Dec 2011. ISSN 1572-9184. Acessado em 12 jul. 2015. Disponível em: <https://doi.org/10.1007/s10723-011-9196-x>. Citado na página 71.

QASHA, R.; CAIA, J.; WATSON, P. A framework for scientific workflow reproducibility in the cloud. In: **2016 IEEE 12th International Conference on e-Science (e-Science)**. [s.n.], 2016. p. 81–90. Acessado em 23 nov. 2017. Disponível em: [<ieeexplore.ieee.org/document/7870888/>](http://ieeexplore.ieee.org/document/7870888/). Citado na página 29.

QASIM, R.; BEGUM, K.; JAHAN, N.; ASHRAFI, T.; IDRIS, S.; RAHMAN, R. Subcellular localization of proteins using automated fuzzy inference system. In: **Informatics, Electronics Vision (ICIEV), 2013 International Conference on**. [s.n.], 2013. p. 1–5. Acessado em 19 nov. 2014. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6572606. Citado nas páginas 66 e 69.

QIFENG, W.; ZHANGJIAN, W. Web services-based system integration approach for manufacturing execution system. In: **Internet Computing Information Services (ICICIS), 2011 International Conference on**. [s.n.], 2011. p. 469–472. Acessado em 13 out. 2014. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6063300. Citado na página 76.

QING-RUI, G.; HAI-TAO, Z. Research of using soa build enterprise application integration strategy. In: **Information Management and Engineering (ICIME), 2010 The 2nd IEEE International Conference on**. [s.n.], 2010. p. 435–438. Acessado em 2 mar. 2014. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5478139>. Citado na página 77.

QUAN, L.; LI, H.; XIA, X.; LI, Q. Packing protein side-chains by parallel ant colonies. In: **Bioinformatics and Biomedicine (BIBM), 2013 IEEE International Conference on**. [s.n.], 2013. p. 10–16. Acessado em 19 nov. 2014. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6732452. Citado nas páginas 67 e 69.

RASHID, M.; NEWTON, M.; HOQUE, M.; SATTAR, A. A local search embedded genetic algorithm for simplified protein structure prediction. In: **Evolutionary Computation (CEC), 2013 IEEE Congress on**. [s.n.], 2013. p. 1091–1098. Acessado em 28 jun. 2014. Disponível em: <http://goo.gl/VFHkKw>. Citado na página 66.

REICH, M.; LIEFELD, T.; GOULD, J.; LERNER, J.; TAMAYO, P.; MESIROV, J. P. Genepattern 2.0. **Nat Genet**, Nature Publishing Group, v. 38, n. 5, p. 500–501, may 2006. ISSN 1061-4036. Acessado em 19 nov. 2015. Disponível em: <http://dx.doi.org/10.1038/ng0506-500>. Citado na página 29.

RITTINGHOUSE, J.; RANSOME, J. **Cloud Computing: Implementation, Management, and Security**. 1st. ed. Boca Raton, FL, USA: CRC Press, Inc., 2009. ISBN 1439806802, 9781439806807. Citado na página 48.

RIZZI, A.; POSSEMATO, F.; LIVI, L.; SEBASTIANI, A.; GIULIANI, A.; MASCIOLI, F. A dissimilarity-based classifier for generalized sequences by a granular computing approach. In: **Neural Networks (IJCNN), The 2013 International Joint Conference on**. [s.n.], 2013. p. 1–8. ISSN 2161-4393. Acessado em 19 nov. 2014. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6707041>.> Citado na página 66.

ROSENTHAL, A.; MORK, P.; LI, M. H.; STANFORD, J.; KOESTER, D.; REYNOLDS, P. Cloud computing: A new business paradigm for biomedical information sharing. **Journal of Biomedical Informatics**, v. 43, n. 2, p. 342 – 353, 2010. ISSN 1532-0464. Acessado em 13 mai. 2016. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1532046409001154>.> Citado na página 47.

ROURE, D. D.; MANUEL, J.; HETTNE, K.; BELHAJJAME, K.; PALMA, R.; KLYNE, G.; MISSIER, P.; RUIZ, J. E.; GOBLE, C. Towards the preservation of scientific workflows. In: **In Procs. of the 8th International Conference on Preservation of Digital Objects (iPRES 2011)**. ACM. [s.n.], 2011. Acessado em 06 mar. 2015. Disponível em: <http://users.ox.ac.uk/~oerc0033/preprints/wfpreservev10.pdf>.> Citado na página 72.

ROUT, S. B.; MISHRA, S.; MISHRA, S. A review on application of artificial neural network(ann) on protein secondary structure prediction. In: **2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)**. [s.n.], 2017. p. 1–6. Acessado em 23 nov. 2017. Disponível em: ieeexplore.ieee.org/abstract/document/8118041/>. Citado na página 68.

RUSINKIEWICZ, M.; SHETH, A. Specification and execution of transactional workflows. In: **IN MODERN DATABASE SYSTEMS: THE OBJECT MODEL, INTEROPERABILITY, AND**. [S.l.]: Addison-Wesley/ ACM Press, 1994. Citado na página 50.

S, B.; RAJENDRAN, C.; RP, S. Penalty based mathematical models for web service composition in a geo-distributed cloud environment. In: **2017 IEEE International Conference on Web Services (ICWS)**. [s.n.], 2017. p. 886–889. Acessado em 28 nov. 2017. Disponível em: ieeexplore.ieee.org/document/8029856/>. Citado na página 79.

SANDES, E. F. D. O.; BOUKERCHE, A.; MELO, A. C. M. A. D. Parallel optimal pairwise biological sequence comparison: Algorithms, platforms, and classification. **ACM Comput. Surv.**, ACM, New York, NY, USA, v. 48, n. 4, p. 63:1–63:36, mar. 2016. ISSN 0360-0300. Acessado em 13 ago. 2016. Disponível em: <http://doi.acm.org/10.1145/2893488>>. Citado nas páginas 29 e 79.

SANDES, E. F. d. O.; MIRANDA, G.; MARTORELL, X.; AYGUADE, E.; TEODORO, G.; MELO, A. C. M. Cudalign 4.0: Incremental speculative traceback for exact chromosome-wide alignment in gpu clusters. **IEEE Transactions on Parallel and Distributed Systems**, v. 27, n. 10, p. 2838–2850, Oct 2016. ISSN 1045-9219. Acessado em 13 ago. 2016. Disponível em: ieeexplore.ieee.org/document/7374729/>. Citado na página 45.

SANDES, E. F. D. O.; MIRANDA, G.; MARTORELL, X.; AYGUADE, E.; TEODORO, G.; MELO, A. C. M. A. D. Masa: A multiplatform architecture for sequence aligners with block pruning. **ACM Trans. Parallel Comput.**, ACM, New York, NY, USA, v. 2, n. 4, p. 28:1–28:31, fev. 2016. ISSN 2329-4949. Acessado em 13 ago. 2016. Disponível em: <http://doi.acm.org/10.1145/2858656>>. Citado nas páginas 45, 79, 80 e 121.

SANDES, E. F. de O.; RALHA, C. G.; MELO, A. C. M. de. An agent-based solution for dynamic multi-node wavefront balancing in biological sequence comparison. **Expert Systems with Applications**, v. 41, n. 10, p. 4929 – 4938, 2014. ISSN 0957-4174. Acessado em 13 ago. 2016. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0957417414000542>>. Citado na página 79.

SANSOM, C. Up in a cloud? **Nature Biotechnology**, Nature Publishing Group, v. 28, n. 1, p. 13–15, jan. 2010. ISSN 1087-0156. Acessado em 13 out. 2014. Disponível em: <<http://dx.doi.org/10.1038/nbt0110-13>>. Citado nas páginas 48 e 69.

SCHADT, E. E.; LINDERMAN, M. D.; SORENSON, J.; LEE, L.; NOLAN, G. P. Computational solutions to large-scale data management and analysis. **Nature reviews. Genetics**, Nature Publishing Group, a division of Macmillan Publishers Limited. All Rights Reserved., v. 11, n. 9, p. 647–657, set. 2010. ISSN 1471-0064. Acessado em 13 mai. 2016. Disponível em: <<http://dx.doi.org/10.1038/nrg2857>>. Citado nas páginas 69, 121 e 122.

SCHAFFER, H. E. X as a service, cloud computing, and the need for good judgment. **IT Professional**, v. 11, n. 5, p. 4–5, Sept 2009. ISSN 1520-9202. Acessado em 18 mai. 2017. Disponível em: <ieeexplore.ieee.org/document/5271524/>. Citado na página 47.

SCOTT, L. P. B.; ROGER, H.; SILVA, A. A **Study of Hydrophobic Effect on the Protein Folding Using Monte Carlo**. 2008. Acessado em 06 mar. 2014. Disponível em: <<http://www.ccp2008.ufop.br/files/submission/200836.pdf>>. Citado na página 39.

SENTURK, I. F.; BALAKRISHNAN, P.; ABU-DOLEH, A.; KAYA, K.; MALLUHI, Q.; ÇATALYÜREK Ümit V. A resource provisioning framework for bioinformatics applications in multi-cloud environments. **Future Generation Computer Systems**, p. –, 2016. ISSN 0167-739X. Acessado em 18 jul. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X16301911>>. Citado nas páginas 30 e 103.

SHARMA, A.; PAPANIKOLAOU, A.; MANOLAKOS, E. Accelerating all-to-all protein structures comparison with talign using a noc many-cores processor architecture. In: **Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2013 IEEE 27th International**. [s.n.], 2013. p. 510–519. Acessado em 13 dez. 2014. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6650926>. Citado nas páginas 42 e 68.

SHEN, N.; FAN, Y.; PAMIDIGHANTAM, S. E-science infrastructures for molecular modeling and parametrization. **Journal of Computational Science**, v. 5, n. 4, p. 576–589, 2014. Disponível em: <https://www.researchgate.net/publication/260113689_E-Science_Infrastructures_for_Molecular_Modeling_and_Parametrization>. Citado na página 71.

SHI, K.; GAO, F.; XU, Q.; XU, G. Integration framework with semantic aspect of heterogeneous system based on ontology and esb. In: **Control and Decision Conference (2014 CCDC), The 26th Chinese**. [s.n.], 2014. p. 4143–4148. Acessado em 07 jan. 2015. Disponível em: <<http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=6852907>>. Citado na página 77.

SILVA, M. A. A. d.; CAVALCANTI, M. C.; BELLOZE, K. T.; SILVA-JUNIOR, F. Agile semantic annotation of scientific texts at the biomedical scenario. In: **2014 IEEE 10th International Conference on e-Science**. [s.n.], 2014. v. 1, p. 100–107. Acessado em 28 mar. 2015. Disponível em: <ieeexplore.ieee.org/abstract/document/6972254/>. Citado na página 73.

SILVA, V.; LEITE, J.; CAMATA, J. J.; OLIVEIRA, D. de; COUTINHO, A. L.; VALDURIEZ, P.; MATTOSO, M. Raw data queries during data-intensive parallel workflow execution. **Future Generation Computer Systems**, v. 75, p. 402 – 422, 2017. ISSN 0167-739X. Acessado em 23 nov. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X17300237>>. Citado na página 72.

SIMONET, A.; CHARD, K.; FEDAK, G.; FOSTER, I. Using active data to provide smart data surveillance to e-science users. In: **2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing**. [s.n.], 2015. p. 269–273. ISSN 1066-6192. Acessado em 15 out. 2015. Disponível em: <ieeexplore.ieee.org/document/7092731/>. Citado nas páginas 100 e 101.

SINGH, M. P.; HUHS, M. N. **Service-oriented computing : semantics, processes, agents**. Chichester, England: J. Wiley and Sons, 2005. ISBN 0-470-09148-7. Citado na página 44.

SINGH, P. Big genomic data in bioinformatics cloud. **Applied Microbiology: open access**, OMICS International,, v. 2, n. 2, p. 1–4, 2016. ISSN 2471-9315. Acessado em 13 fev. 2017. Disponível em: <<https://www.omicsonline.org/open-access/big-genomic-data-in-bioinformatics-cloud-2471-9315-1000113.php?aid=73709>>. Citado nas páginas 45 e 46.

SLEATOR, R. D. Prediction of protein functions. In: _____. **Functional Genomics: Methods and Protocols**. New York, NY: Springer New York, 2012. p. 15–24. ISBN 978-1-61779-424-7. Acessado em 13 mai. 2016. Disponível em: <https://doi.org/10.1007/978-1-61779-424-7_2>. Citado na página 118.

SMITH, C. W.; ABEYSINGHE, E. The phasta science gateway: Web-based execution of adaptive computational fluid dynamics simulations. In: **Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact**. New York, NY, USA: ACM, 2017. (PEARC17), p. 70:1–70:4. ISBN 978-1-4503-5272-7. Acessado em 13 nov. 2017. Disponível em: <<http://doi.acm.org/10.1145/3093338.3104151>>. Citado na página 71.

SMOLA, A. J.; SCHÖLKOPF, B. A tutorial on support vector regression. **Statistics and computing**, Kluwer Academic Publishers, v. 14, n. 3, p. 199–222, 2004. Acessado em 15 out. 2017. Disponível em: <<https://alex.smola.org/papers/2004/SmoSch04.pdf>>. Citado nas páginas 107 e 111.

SONNTAG, M.; KARASTOYANOVA, D.; DEELMAN, E. Bridging the gap between business and scientific workflows: Humans in the loop of scientific workflows. In: **2010 IEEE Sixth International Conference on e-Science**. [s.n.], 2010. p. 206–213. Acessado em 22 abr. 2014. Disponível em: <<https://ieeexplore.ieee.org/document/5693919/>>. Citado nas páginas 50 e 70.

SOVIC, I.; ANTULOV-FANTULIN, N.; CANADI, I.; PISKOREC, M.; SIKI, M. Parallel protein docking tool. In: **MIPRO, 2010 Proceedings of the 33rd International Convention**. [s.n.], 2010. p. 1333–1338. Acessado em 09 nov. 2013. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5533683>. Citado nas páginas 41, 42 e 67.

SPOEL, D. v. d.; LINDHAL, E.; HESS, B.; BUUREN, A. van; APOL, E.; MEULENHOF, P.; TIELEMANN, D.; SIJBERS, A.; FEENSTRA, K.; DRUNEN, R. van *et al.* **Gromacs User Manual version 4.5. 5**, 2011. 2013. Acessado em 13 abr. 2015. Disponível em: <<ftp://ftp.gromacs.org/pub/manual/manual-5.0.4.pdf>>. Citado na página 90.

SRINIVASULU, P.; BABU, M. S.; VENKAT, R.; RAJESH, K. Cloud service oriented architecture (csoa) for agriculture through internet of things (iot) and big data. In: **2017 IEEE International Conference on Electrical, Instrumentation and Communication Engineering (ICEICE)**. [s.n.], 2017. p. 1–6. Acessado em 18 nov. 2017. Disponível em: [<ieeexplore.ieee.org/abstract/document/8191906/>](http://ieeexplore.ieee.org/abstract/document/8191906/). Citado nas páginas 45 e 75.

STANDLEY, D. M.; KINJO, A. R.; KINOSHITA, K.; NAKAMURA, H. Protein structure databases with new web services for structural biology and biomedical research. **Briefings in Bioinformatics**, v. 9, n. 4, p. 276–285, 2008. Acessado em 13 abr. 2015. Disponível em: <http://www.ncbi.nlm.nih.gov/pubmed/18430752>.> Citado nas páginas 38 e 39.

STEIN, L. D. Towards a cyberinfrastructure for the biological sciences: progress, visions and challenges. **Nature Reviews Genetics**, Nature Publishing Group, v. 9, n. 9, p. 678–688, set. 2008. ISSN 1471-0056. Acessado em 14 abr. 2014. Disponível em: <http://dx.doi.org/10.1038/nrg2414>>. Citado na página 28.

STEWART, C. A.; KATZ, D. S.; HART, D. L.; LANTRIP, D.; MCCAULAY, D. S.; MOORE, R. L. **Technical Report: Survey of cyberinfrastructure needs and interests of NSF-funded principal investigators**. [S.l.], 2011. Acessado em 18 jul. 2015. Disponível em: <http://hdl.handle.net/2022/9917>>. Citado na página 121.

STITZ, H.; GRATZL, S.; KRIEGER, M.; STREIT, M. Cloudgazer: A divide-and-conquer approach to monitoring and optimizing cloud-based networks. In: **2015 IEEE Pacific Visualization Symposium (PacificVis)**. [s.n.], 2015. p. 175–182. ISSN 2165-8765. Acessado em 19 ago. 2017. Disponível em: <http://ieeexplore.ieee.org/document/7156375/>>. Citado na página 73.

SUN, X.; OUYANG, Z.; YUE, D. Short-term load forecasting based on multivariate linear regression. In: **2017 IEEE Conference on Energy Internet and Energy System Integration (EI2)**. [s.n.], 2017. p. 1–5. Acessado em 18 nov. 2017. Disponível em: <http://ieeexplore.ieee.org/document/8245401/>>. Citado na página 93.

TAHA, K.; YOO, P.; ZAABI, M. A. ipfpi: A system for improving protein function prediction through cumulative iterations. **Computational Biology and Bioinformatics, IEEE/ACM Transactions on**, PP, n. 99, p. 1–1, 2014. ISSN 1545-5963. Acessado em 13 jan. 2015. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6871330>>. Citado nas páginas 67 e 69.

TAN, W.; ZHANG, J.; FOSTER, I. Network analysis of scientific workflows: A gateway to reuse. **Computer**, v. 43, n. 9, p. 54–61, Sept 2010. ISSN 0018-9162. Acessado em 06 mar. 2015. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5569056>. Citado nas páginas 44 e 50.

TAYLOR, I.; SHIELDS, M.; WANG, I.; RANA, O. Triana applications within grid computing and peer to peer environments. **Journal of Grid Computing**, v. 1, n. 2, p. 199–217, Jun 2003. ISSN 1572-9184. Acessado em 18 jul. 2015. Disponível em: <https://doi.org/10.1023/B:GRID.0000024074.63139.ce>>. Citado na página 53.

THAMAN, J.; SINGH, M. Current perspective in task scheduling techniques in cloud computing : A review. **International Journal in Foundations of Computer Science & Technology (IJFCST)**, v. 6, n. 1, jan 2016. Acessado em 19 ago. 2017. Disponível em: <https://www.researchgate.net/publication/294107758>>. Citado na página 78.

THOMAS, J.; THOMAS, M.; GHINEA, G. Modeling of web services flow. In: **E-Commerce, 2003. CEC 2003. IEEE International Conference on**. [s.n.], 2003. p. 391–398. Acessado em 13 set. 2014. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1210275>. Citado na página 44.

THOMAS, S.; TANASE, G.; DALE, L. K.; MOREIRA, J. M.; RAUCHWERGER, L.; AMATO, N. M. Parallel protein folding with stapl. **Concurrency and Computation: Practice and Experience**, John Wiley and Sons, Ltd., v. 17, n. 14, p. 1643–1656, 2005. ISSN 1532-0634. Acessado em 14 out. 2014. Disponível em: <<http://dx.doi.org/10.1002/cpe.950>>. Citado nas páginas 39 e 67.

TOOSI, A. N.; SINNOTT, R. O.; BUYYA, R. Resource provisioning for data-intensive applications with deadline constraints on hybrid clouds using aneka. **Future Generation Computer Systems**, 2017. ISSN 0167-739X. Acessado em 18 dez. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X17301863>>. Citado nas páginas 49, 79 e 80.

TORTEROLO, L.; PORRO, I.; FATO, M.; MELATO, M.; CALANDUCCI, A.; BARBERA, R. Building science gateways with enginframe: a life science example. In: . [s.n.], 2009. Acessado em 18 jul. 2015. Disponível em: <ceur-ws.org/Vol-513/paper10.pdf>. Citado nas páginas 56 e 121.

VAHI, K.; HARVEY, I.; SAMAK, T.; GUNTER, D.; EVANS, K.; ROGERS, D.; TAYLOR, I.; GOODE, M.; SILVA, F.; AL-SHAKARCHI, E.; MEHTA, G.; DEELMAN, E.; JONES, A. A case study into using common real-time workflow monitoring infrastructure for scientific workflows. **Journal of Grid Computing**, v. 11, n. 3, p. 381–406, Sep 2013. ISSN 1572-9184. Acessado em 19 set. 2015. Disponível em: <<https://doi.org/10.1007/s10723-013-9265-4>>. Citado nas páginas 50 e 120.

VAIDYANATHAN, R.; KOLAROV, A.; CARUSO, F.; FORBES, G. A novel cross-layer modeling framework for analyzing SOA-based information services. **2011 - MILCOM 2011 Military Communications Conference**, Ieee, p. 2019–2024, nov. 2011. Acessado em 13 out. 2013. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6127615>>. Citado na página 44.

VECCHIOLA, C.; PANDEY, S.; BUYYA, R. High-Performance Cloud Computing: A View of Scientific Applications. **2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks**, Ieee, p. 4–16, 2009. Acessado em 26 nov. 2013. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5381983>>. Citado na página 47.

VELEV, D.; ZLATEVA, P. **Cloud infrastructure security**. [S.l.]: Springer Berlin Heidelberg, 2011. 140–148 p. Citado na página 47.

VIJAYAKUMAR, S.; LAKSHMI, P. A fuzzy inference system for predicting allergenicity and allergic cross-reactivity in proteins. In: **Bioinformatics and Biomedicine (BIBM), 2013 IEEE International Conference on**. [s.n.], 2013. p. 49–52. Acessado em 14 jul. 2014. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6732458>. Citado na página 66.

VILLANUEVA-ROSALES, N.; CHAVIRA, L. G.; RIO, N. d.; PENNINGTON, D. science through the integration of data and models: A biodiversity scenario. In: **2015 IEEE 11th International Conference on e-Science**. [s.n.], 2015. p. 171–176. Acessado em 13 out. 2016. Disponível em: <ieeexplore.ieee.org/document/7304288/>. Citado na página 76.

VRBA, P.; FUKSA, M.; KLIMA, M. JADE-JbossESB Gateway: Integration of multi-agent system with enterprise service bus. In: **Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on**. [s.n.], 2014. p. 3663–3668. Acessado em 09 fev. 2015. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6974499. Citado na página 77.

WALLOM, D. C. H.; BOOTH, T.; BOWERY, A.; COLLIER, B.; KERSHAW, P.; PRIYAM, A.; WURM, Y.; FIELD, D. Desktop as a service supporting environmental 'omics. In: **2015 IEEE 11th International Conference on e-Science**. [s.n.], 2015. p. 214–214. Acessado em 14 out. 2016. Disponível em: <https://ieeexplore.ieee.org/document/7304293/>. Citado na página 78.

WALLOM, D. C. H.; TURILLI, M.; DRESCHER, M.; SCARDACI, D.; NEWHOUSE, S. Federating infrastructure as a service cloud computing systems to create a uniform e-infrastructure for research. In: **2015 IEEE 11th International Conference on e-Science**. [s.n.], 2015. p. 155–164. Acessado em 14 out. 2016. Disponível em: <https://ieeexplore.ieee.org/iel7/7303998/7304061/07304286.pdf?arnumber=7304286>. Citado na página 78.

WAN, S.; MAK, M.-W.; ZHANG, B.; WANG, Y.; KUNG, S.-Y. An ensemble classifier with random projection for predicting multi-label protein subcellular localization. In: **Bioinformatics and Biomedicine (BIBM), 2013 IEEE International Conference on**. [s.n.], 2013. p. 35–42. Acessado em 14 jul. 2014. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6732715. Citado nas páginas 67 e 69.

WANG, C.; GONG, L.; WANG, A.; LI, X.; HUNG, P. C. K.; XUEHAI, Z. Solar: Services-oriented deep learning architectures. **IEEE Transactions on Services Computing**, PP, n. 99, p. 1–1, 2017. ISSN 1939-1374. Acessado em 18 nov. 2017. Disponível em: <https://ieeexplore.ieee.org/document/7558064/>. Citado na página 75.

WANG, J.; YU, A.; ZHANG, X.; QU, L. A dynamic data integration model based on soa. In: **Computing, Communication, Control, and Management, 2009. CCCM 2009. ISECS International Colloquium on**. [s.n.], 2009. v. 2, p. 196–199. Acessado em 13 fev. 2014. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5267945>. Citado nas páginas 44, 76 e 80.

WANG, T.; YANG, Z.; TAN, W.; HU, X. Identifying rna-protein interactions using feature dimension reduction method. In: **Computer Science Education (ICCSE), 2013 8th International Conference on**. [s.n.], 2013. p. 969–972. Acessado em 09 nov. 2013. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6554053. Citado na página 66.

WANG, W.; LIU, J. Distinguishing single-stranded and double-stranded dna binding proteins based on structural information. In: **Bioinformatics and Biomedicine (BIBM), 2013 IEEE International Conference on**. [s.n.], 2013. p. 612–612. Acessado em 13 abr. 2015. Disponível em: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6732568. Citado nas páginas 67 e 92.

WASSERMANN, B.; EMMERICH, W.; BUTCHART, B.; CAMERON, N.; CHEN, L.; PATEL, J. Sedna: A bpel-based environment for visual scientific workflow modeling. In: _____. **Workflows for e-Science: Scientific Workflows for Grids**. London: Springer London, 2007. p. 428–449. ISBN 978-1-84628-757-2. Acessado em 17 fev. 2015. Disponível em: https://doi.org/10.1007/978-1-84628-757-2_26. Citado nas páginas 50 e 57.

WEFOLD. **We Fold**. 2013. Acessado em 23 nov. 2013. Disponível em: <http://wefold.org/>. Citado nas páginas 28, 42 e 127.

- WEI, S.-C.; YEH, W.-C. Resource allocation decision model for dependable and cost-effective grid applications based on grid bank. **Future Generation Computer Systems**, v. 77, n. Supplement C, p. 12 – 28, 2017. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X17303941>>. Citado na página 79.
- WFMC. **Workflow Management Coalition**. 2014. Acessado em 08 dez. 2014. Disponível em: <http://www.wfmc.org/>>. Citado na página 50.
- WHITE, T. **Hadoop: The Definitive Guide**. 1st. ed. [S.l.]: O'Reilly Media, Inc., 2009. ISBN 0596521979, 9780596521974. Citado na página 48.
- WILDE, M.; HATEGAN, M.; WOZNIAK, J. M.; CLIFFORD, B.; KATZ, D. S.; FOSTER, I. T. Swift: A language for distributed parallel scripting. **Parallel Computing**, v. 37, p. 633–652, 09/2011 2011. Acessado em 13 ago. 2015. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167819111000524>>. Citado na página 29.
- WILKINS-DIEHR, N.; GANNON, D.; KLIMECK, G.; OSTER, S.; PAMIDIGHANTAM, S. Teragrid science gateways and their impact on science. **Computer**, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 41, n. 11, p. 32–41, nov. 2008. ISSN 0018-9162. Acessado em 13 mar. 2017. Disponível em: <http://dx.doi.org/10.1109/MC.2008.470>>. Citado na página 70.
- WOLSTENCROFT, K.; HAINES, R.; FELLOWS, D.; WILLIAMS, A.; WITHERS, D.; OWEN, S.; SOILAND-REYES, S.; DUNLOP, I.; NENADIC, A.; FISHER, P.; BHAGAT, J.; BELHAJ-JAME, K.; BACALL, F.; HARDISTY, A.; HIDALGA, A. Nieva de la; VARGAS, M. P. B.; SUFI, S.; GOBLE, C. The taverna workflow suite: designing and executing workflows of web services on the desktop, web or in the cloud. **Nucleic Acids Research**, v. 41, n. W1, p. W557–W561, 2013. Acessado em 06 mar. 2014. Disponível em: <http://www.ncbi.nlm.nih.gov/pubmed/23640334>>. Citado nas páginas 29 e 55.
- XU, Z. P.; CUI, J. Z.; QIN, H. N.; MA, K. P. The concept and framework of biodiversity e-science infrastructure in China. In: **2010 5th International ICST Conference on Communications and Networking in China**. [s.n.], 2010. p. 1–5. Acessado em 28 jun. 2014. Disponível em: <http://yadda.icm.edu.pl/yadda/element/bwmeta1.element.ieee-000005684793>>. Citado na página 75.
- YANG, J.; ZHANG, Y. I-tasser server: new development for protein structure and function predictions. **Nucleic Acids Res**, Oxford University Press, v. 43, n. Web Server issue, p. W174–W181, Jul 2015. ISSN 0305-1048. Acessado em 13 jul. 2016. Disponível em: <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4489253/>>. Citado nas páginas 68 e 69.
- YANG, X.; ZHANG, H. Cloud Computing and SOA Convergence Research. **2012 Fifth International Symposium on Computational Intelligence and Design**, Ieee, p. 330–335, out. 2012. Acessado em 20 mar. 2014. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6406988>>. Citado na página 44.
- YANG, Y. Application of soa and web service in implementing heterogeneous system integration. In: **Information Science and Cloud Computing Companion (ISCC-C), 2013 International Conference on**. [s.n.], 2013. p. 1–5. Acessado em 09 fev. 2015. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6973560>>. Citado nas páginas 44 e 76.
- YEH, J.-S.; CHEN, D.-Y.; OUHYOUNG, M. A web-based protein retrieval system by matching visual similarity. In: **Emerging Information Technology Conference, 2005**. [s.n.], 2005. p.

108–110. Acessado em 19 nov. 2014. Disponível em: <<http://bioinformatics.oxfordjournals.org/content/21/13/3056.full.pdf>> Citado na página 67.

YING, F.; LEI, G. Optimal scheduling simulation of software for multi-tenant in cloud computing environment. In: **2014 Fifth International Conference on Intelligent Systems Design and Engineering Applications**. [s.n.], 2014. p. 688–692. Acessado em 12 abr. 2017. Disponível em: <<https://ieeexplore.ieee.org/iel7/6968681/6977522/06977691.pdf?arnumber=6977691>>. Citado na página 78.

YU, J.; BUYYA, R. A taxonomy of workflow management systems for grid computing. **Journal of Grid Computing**, v. 3, n. 3, p. 171–200, Sep 2005. ISSN 1572-9184. Acessado em 14 set. 2015. Disponível em: <<https://doi.org/10.1007/s10723-005-9010-8>>. Citado na página 53.

YUAN, D.; LIU, X.; CUI, L.; ZHANG, T.; LI, W.; CAO, D.; YANG, Y. An algorithm for cost-effectively storing scientific datasets with multiple service providers in the cloud. In: **2013 IEEE 9th International Conference on e-Science**. [s.n.], 2013. p. 285–292. Acessado em 12 mar. 2015. Disponível em: <<https://ieeexplore.ieee.org/document/6683919/>>. Citado nas páginas 49, 95 e 121.

ZAGROVIC, B.; SNOW, C. D.; SHIRTS, M. R.; PANDE, V. S. Simulation of folding of a small alpha-helical protein in atomistic detail using worldwide-distributed computing. **Journal of Molecular Biology**, 2002. Acessado em 16 jul. 2014. Disponível em: <<http://www.ncbi.nlm.nih.gov/pubmed/12417204>>. Citado na página 42.

ZHANG, L.; LI, J.; YU, M. An integration research on service-oriented architecture (soa) for logistics information system. In: **Service Operations and Logistics, and Informatics, 2006. SOLI '06. IEEE International Conference on**. [s.n.], 2006. p. 1059–1063. Acessado em 2 mar. 2014. Disponível em: <<http://goo.gl/sjTdPz>> Citado na página 77.

ZHANG, Y. I-tasser server for protein 3d structure prediction. **BMC Bioinformatics**, v. 9, n. 1, p. 40, Jan 2008. ISSN 1471-2105. Acessado em 13 jul. 2016. Disponível em: <<https://doi.org/10.1186/1471-2105-9-40>>. Citado na página 42.

ZHAO, Y.; LI, Y.; RAICU, I.; LU, S.; TIAN, W.; LIU, H. Enabling scalable scientific workflow management in the cloud. **Future Generation Computer Systems**, v. 46, n. Supplement C, p. 3 – 16, 2015. ISSN 0167-739X. Acessado em 23 nov. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X14002179>>. Citado nas páginas 55, 73 e 75.

ZHAO, Z.; PASCHKE, A.; ZHANG, R. A rule-based agent-oriented approach for supporting weakly-structured scientific workflows. 03 2016. Acessado em 26 abr. 2017. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1570826816000184>>. Citado nas páginas 72 e 75.

ZHENG, W.; QIN, Y.; EMMANUEL, B.; ZHANG, D.; CHEN, J. Cost optimization for deadline-aware scheduling of big-data processing jobs on clouds. **Future Generation Computer Systems**, p. –, 2017. ISSN 0167-739X. Acessado em 15 fev. 2018. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0167739X17317296>>. Citado nas páginas 73 e 75.

ZIMMERMANN, L.; GRUNZKE, R.; KRÜGER, J. Maintaining a science gateway - lessons learned from mosgrid. In: **HICSS**. [s.n.], 2017. Acessado em 18 nov. 2017. Disponível em: <<https://scholarspace.manoa.hawaii.edu/bitstream/10125/41918/1/paper0769.pdf>>. Citado na página 92.

ZIV, G.; HARAN, G. Protein folding, protein collapse, and tanford's transfer model: Lessons from single-molecule fret. **Journal of the American Chemical Society**, v. 131, n. 8, p. 2942–2947, 2009. Acessado em 13 out. 2014. Disponível em: <<https://doi.org/10.1021/ja808305u>>. Citado na página 27.