Universidade Federal do Piauí
Centro de Ciências da Natureza
Programa de Pós-Graduação em Ciência da Computação

Athena: uma Arquitetura e uma Ferramenta para Auxiliar o Desenvolvimento de Sistemas Baseados em Inteligência Computacional

Pedro Almir Martins de Oliveira

Número de Ordem PPGCC: M001

Teresina-PI, 28 de Março de 2016

Pedro Almir Martins de Oliveira

Athena: uma Arquitetura e uma Ferramenta para Auxiliar o Desenvolvimento de Sistemas Baseados em Inteligência Computacional

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da UFPI (área de concentração: Computação Aplicada), como parte dos requisitos necessários para a obtenção do Título de Mestre em Ciência da Computação.

Universidade Federal do Piauí – UFPI

Centro de Ciências da Natureza

Programa de Pós-Graduação em Ciência da Computação

Orientador: Pedro de Alcântara dos Santos Neto

Teresina-PI

28 de Março de 2016

Pedro Almir Martins de Oliveira
Athena: uma Arquitetura e uma Ferramenta para Auxiliar o Desenvolvimento de Sistemas Baseados em Inteligência Computacional/ Pedro Almir Martins de Oliveira. – Teresina-PI, 28 de Março de 2016-
157 p. : il. (algumas color.) ; 30 cm.

Orientador: Pedro de Alcântara dos Santos Neto

Dissertação (Mestrado) – Universidade Federal do Piauí – UFPI
Centro de Ciências da Natureza
Programa de Pós-Graduação em Ciência da Computação, 28 de Março de 2016.

1. Inteligência Computacional. 2. Arquitetura. 3. Ferramenta. I. Pedro de Alcântara dos Santos Neto. II. Universidade Federal do Piauí. III. Athena: uma Arquitetura e uma Ferramenta para Auxiliar o Desenvolvimento de Sistemas Baseados em Inteligência Computacional

CDU 02:141:005.7

Athena: uma Arquitetura e uma Ferramenta para Auxiliar o Desenvolvimento de Sistemas Baseados em Inteligência Computacional

PEDRO ALMIR MARTINS DE OLIVEIRA

Dissertação apresentada ao Programa de Pós–Graduação em Ciência da Computação do Centro de Ciências da Natureza da Universidade Federal do Piauí, como parte integrante dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Aprovado por:

Prof. Pedro de Alcântara dos Santos Neto

(Presidente da Banca Examinadora)

Prof. Alexandre César Muniz de Oliveira

(Examinador Externo)

Prof. Erick Baptista Passos

(Examinador Interno)

Prof. Ricardo de Andrade Lira Rabêlo

(Examinador Interno)

Prof. Vinícius Ponte Machado

(Examinador Interno)

Teresina – PI, 28 de março de 2016

Aos meus pais Angélica M. M. Rosa de Oliveira e Antônio Matias de Oliveira, por serem a pedra fundamental na construção da minha vida.

Agradecimentos

Qual o significado da gratidão? Antes de escrever esta seção, procurei refletir um pouco sobre esse sentimento que torna as pessoas tão felizes. Segundo o dicionário, a palavra gratidão significa a qualidade de quem é grato, reconhecimento, sentimento de lembrança e agradecimento por um bem recebido, ato de reconhecer alguém por uma ação ou benefício alcançado (FERREIRA, 1986). Na psicologia, esse sentimento é descrito como uma ação reativa ou retributiva gerada pela capacidade de um indivíduo sentir satisfação ou um sentimento positivo, em função de uma ação benevolente de outrem (FREITAS; SILVEIRA; PIETA, 2009). Para a filosofia, a gratidão não é apenas a mais rica das virtudes, mas sim a mãe de todas as outras (filósofo romano Marcus Tullius Cicero).

Entretanto, a melhor definição que encontrei foi escrita pelo Pe. Ederson Iarochevski e diz: "**Agradecer é um gesto de amor!** Dizer muito obrigado a uma pessoa é tão belo quando dizer eu te amo. Quando agradecemos, estamos nos desvencilhando de um dos vícios mais perigosos de nosso tempo, a auto-suficiência. Como é triste alguém pensar que não precisa de ninguém para viver. Quando o filho agradece a seus pais pelo esforço e carinho a ele dispensados, sela seu amor em relação aos pais. Quando o esposo diz muito obrigado à sua esposa pelo carinho e cuidado que a mesma tem com ele, está estabelecendo um laço ainda mais forte. Mesma situação quando a esposa reconhece a importância de seu esposo. Quando os amigos reconhecem o quanto são importantes uns para os outros, estão reafirmando uma relação que permanecerá no tempo, independentemente das intempéries que possam surgir. Mas dizer muito obrigado em tempos de individualismos é um ato de coragem e receber um muito obrigado em tempos de auto-suficiências é um ato de humildade." Dessa forma, compreendo que agradecer é um exercício bom para a alma e, como este trabalho é fruto da minha formação acadêmica e profissional, inicio meus agradecimentos pelos profissionais que me guiaram durante esse processo de formação.

Conheci meu orientador, professor Pedro de Alcântara dos Santos Neto, em 2011 e a primeira frase que ouvi foi: "O trabalho engrandece o homem." De fato, foi isso que aconteceu de lá até aqui: muito trabalho e muitos ensinamentos. Professor Pedro, sou grato pela sua dedicação, confiança e pelas oportunidades concedidas durante essa jornada.

Diferente de outros trabalhos, tive o privilégio de ter dois orientadores igualmente fundamentais para a conclusão desta dissertação. O prof. Ricardo de Andrade Lira Rabêlo é um exemplo de ser humano dedicado a ajudar o próximo, mesmo que isso signifique levantar cedo no domingo e ir à UFPI orientar seus alunos. Ele, mesmo sem nenhuma obrigação, revisou cada uma das minhas apresentações antes da prova didática do IFMA. Ao senhor, sou grato pela oportunidade de ter me tornado um professor.

Agradeço também ao professor Ricardo Britto que é nosso elo de ligação com o Instituto de Tecnologia de *Blekinge* (Suécia) e com os grandes pesquisadores da Engenharia de *Software*. Nos últimos anos ele fortaleceu nosso grupo de pesquisa e foi importante na escolha dos caminhos a seguir dentro desta pesquisa. Aproveito também para agradecer ao professor Vinícius Ponte Machado pelos ensinamentos obtidos nas disciplinas de Inteligência Artificial e Aprendizagem de Máquina, bem como pelo apoio na realização do estudo experimental.

Aos professores do Departamento de Computação, do colégio CEBRAPI (Ensino Médio) e aos demais educadores que participaram da minha formação, estendo meus agradecimentos pelos inúmeros incentivos. Ao professor Ricardo Augusto da UFSCar pelos valiosos conselhos a respeito do doutorado. Aos professores Erick Passos e Alexandre César Muniz de Oliveira por terem aceitado participar da banca de avaliação deste trabalho.

Ao pesquisador Anderson Aragão, pelo fornecimento dos dados relacionados ao problema da estimação do tamanho de nanopartículas de prata. Esses dados foram de suma importância para a realização do experimento controlado que avaliou a proposta deste trabalho. Ao amigo e aluno de mestrado, Roney Lira, pelo empenho na divulgação e utilização da Athena.

Quem caminha sozinho pode até chegar mais rápido, mas aquele que vai acompanhado, com certeza vai mais longe (Clarice Lispector). Portanto, não posso deixar de agradecer ao meu amigo e companheiro de trabalho, Ronyerison Braga, pelo zelo e dedicação em cada atividade realizada dentro deste projeto. Tenho certeza que ainda vamos colher bons frutos dessa parceria. Como nós pertencemos ao mesmo laboratório e jogamos futsal juntos, aproveito para estender meus agradecimentos aos amigos do Lab. EaSII, Vanderson, Rafael e Thasciano, e aos amigos do futebol na UFPI, Wermerson, Wender e Moisés Bispo.

Friend will be Friends (Queen). Agradeço aos meus amigos pela oportunidade de viver momentos únicos: Jefferson Henrique (Jeff), Matheus Campanhã (Bolsistinha), Thiago Allisson (Thiagão), Hugo Cordeiro, Lucas Alencar (Caxias), Natan, Ramon, Laysson, Jonas Rafael. Aos meus amigos do saudoso grupo TheBest pelas melhores viagens e por tantos momentos inesquecíveis: Rafael, Dayana, Deinny, Jéssica, Aldair, Luziane, Eduardo, Francilane, Amanda, Willame (Cifra Club) e Danilo (Batera).

Aos amigos da graduação e mestrado em Computação: Jackson Cunha (Jack), Roniel, Kalif, Jonatas, Galeno, Marcos Frazão, Werney, Dennis Sávio, Bruno Vicente, Rogério Figueiredo, Anderson Soares, Rodolfo Paz, Cleiton Moura e Marcus Meirelles.

Aos amigos do IFMA campus Pedreiras, pelo acolhimento na instituição e pela compreensão nos momentos pré-defesa: Gedeon Reis, Ângelo Soares, Patrícia Vilar, Paulo Roberto, Rafael Ferreira, Marcos Reges, Jéssica Silveira, Maria Lúcia, Virgínia e Márcio.

À empresa Infoway, pelo meu amadurecimento profissional e principalmente pela oportunidade de ter trabalhado com bons profissionais: Antônio Emanuel (fera no violão), Rondinele, Idelvane, Danilo Medeiros, Ewerton Costa, Ewerton Leal, Livya Castro, Jardiel, Lucas Aquiles, Ângela, Wan, Ney e Adalton.

Sem o amor, eu nada seria (Renato Russo)! Gostaria de agradecer muito à minha futura esposa, Jayane Alves de Brito. Ela sofreu nas derrotas e vibrou nas vitórias. Sempre esteve ao meu lado de braços abertos e com um sorriso acolhedor. Mais do que minha namorada, noiva ou esposa, ela é minha companheira e tenho muito a retribuir por isso. Linda mulher, mais meiga, não há! Esse agradecimento também se estende às famílias Alves e Brito, pelo acolhimento, apoio, amor e carinho: Umbelina Alves, Sebastião Brito, Jayro e Michelly, Jailson e Maria dos Anjos, Marcos e Kelly, Otacílio e Ana, Wellington e Teresinha, Dilon e Ceiça, Anselmo e Maria Francisca, Luzia Alves, Lourdes, Érica, Rita e Pedro Gabriel.

Aos meus familiares: avós, madrinhas, padrinhos, tias, tios, primos e primas. Em especial à minha tia Maria da Luz por ser meu porto seguro. À minha vó Elvira (*in memoriam*), se agora conquisto mais uma vitória, é porque um dia ela esteve ao meu lado e me ensinou a seguir pelo bom caminho. À minha vó Júlia e às minhas tias Ana e Sheilla, pelo amor, carinho e pelas ótimas conversas acompanhadas sempre de um bom café. Ao meu avô Almir Rosa e aos meus tios Júnior e Patrocínio Neto, pelo apoio nos meus estudos.

Faltam-me expressões suficientemente adequadas para demonstrar toda minha gratidão aos meus pais, Antônio Matias de Oliveira e Angélica M. M. Rosa de Oliveira. Eles me ensinaram que nada é impossível quando se tem fé e determinação, e juntos construíram a base familiar sólida que me apoia em todos os momentos. Muito obrigado pelo amor, carinho e dedicação. À minha irmã Juliana (futura arquiteta), pelo amor incondicional e pelo café das madrugadas de estudo. Como irmão, acabo cobrando muito, mas no fundo isso é apenas zelo por uma joia rara.

Por fim, o mais importante! Agradeço a Deus por tudo. Pelas dificuldades que me ensinaram a ser forte e pelos momentos felizes que me mostram o valor da vida. Obrigado por guiar meus passos!

"Se vi mais longe foi por estar de pé
sobre ombros de gigantes."
(Isaac Newton)

"Coming together is a beginning.
Keeping together is progress.
Working together is success!"
(Henry Ford)

Resumo

A Inteligência Computacional (IC) concentra-se no estudo de mecanismos adaptativos que possibilitam o comportamento inteligente em sistemas complexos e dinâmicos. Essa área reúne diferentes técnicas que exploram a tolerância a precisão, incerteza e verdades parciais para obter flexibilidade, robustez e soluções de baixo custo. As técnicas de IC representam um paradigma computacional emergente, pois vêm obtendo sucesso na resolução de problemas complexos nas mais diversas áreas do conhecimento, por exemplo, classificação e predição de câncer com base no perfil genético do paciente (medicina), obtenção de novos compostos poliméricos (química), identificação de novas espécies de morcegos (biologia), modelagem de séries financeiras (economia), reconhecimento de distúrbios na qualidade da energia elétrica (engenharia elétrica), projeto de circuitos VLSI (arquitetura de computadores), alocação de equipes (engenharia de *software*), combate a crimes cibernéticos (segurança), dentre outros. Esse conjunto de trabalhos representa apenas um retrato da ampla gama de aplicações possíveis para tais técnicas. Entretanto, apesar de todas essas possibilidades, existe uma série de dificuldades relacionadas à construção de sistemas inteligentes: o alto custo de desenvolvimento, a difícil reutilização das implementações, a quantidade de erros resultante da implementação manual, ferramentas inadequadas e com pouco suporte a construção de sistemas híbridos (duas ou mais técnicas integradas), dificuldades para realizar experimentos e para efetivar a integração com outros sistemas. O conjunto dessas dificuldades representa o problema a ser abordado neste trabalho. Nesse contexto, propõe-se a construção de uma arquitetura e uma ferramenta aptas a minimizar o impacto dessas dificuldades no desenvolvimento de Sistemas Baseados em Inteligência Computacional (SBIC). Essa proposta traz consigo a inovação na forma como esses sistemas são desenvolvidos ao unir os algoritmos (técnicas) presentes nessa linha de pesquisa (IC) com aspectos da Computação em Nuvem criando um novo conceito: Inteligência Computacional como Serviço (CIasS, do inglês *Computational Intelligence as a Service*). Foi realizado um mapeamento sistemático sobre ferramentas que apoiam a construção de SBIC para obter uma visão geral dessa linha de pesquisa, além de facilitar a definição dos requisitos necessários para superar, de forma holística, as dificuldades supracitadas. Com os requisitos detalhados, foram desenvolvidas uma arquitetura e uma ferramenta com os seguintes princípios: simplicidade, extensibilidade, *software* como serviço, alta performance e colaboração. A avaliação da proposta foi conclusiva quanto a adequabilidade da ferramenta no apoio ao desenvolvimento de Sistemas Inteligentes. Ambas, arquitetura e ferramenta representam juntas a principal proposta deste trabalho e foram intituladas Athena.

Palavras-chaves: Inteligência Computacional. Sistemas Inteligentes. Sistemas Híbridos. Arquitetura. Ferramenta.

Abstract

Computational Intelligence (CI) is a sub-branch of Artificial Intelligence (AI) and is concentrated in the study of adaptive mechanisms to enable or facilitate intelligent behavior in complex and changing environments. This area combines different techniques that exploit the tolerance for imprecision, uncertainty and partial truth to achieve tractability, robustness, low solution cost and better rapport with reality. These techniques represent an emerging computational paradigm because they have been successful in solving complex problem in the most diverse areas, for example, cancer classification and prediction based on the genetic patient profile (medicine), obtaining new polymeric compounds (chemistry), identifying new species of bats (Biology), financial time series modeling (economy), power-quality disturbance recognition (electrical engineering), VLSI circuit design (computer architecture), agile team allocation (software engineering), cyber security (security), and so on. This works represents only a part of the wide range of applications for these techniques. However, despite their usefulness, developing solutions based in CI techniques is not a trivial activity, since it involves the implementation/adaptation of algorithms to specific context and problems. This work deals with the following difficulties: the high development cost, difficult to reuse implementations, manual implementation is error prone, inadequate tools and with little support for the construction of hybrid systems, difficulties to perform experiments and to integrate with other systems. This work presents architecture and a visual tool developed aiming at offering a simple approach to the development of CI-based software systems, by dragging and dropping components in a visual environment, creating a new concept, that we call CI as a Service (CIaaS). Based on an empirical evaluation and a performance analysis, we can state that Athena can help researchers to solve complex problems using Computational Intelligence, making the creation, distribution and customization of CI approaches easy and allowing access to the results from anywhere. Both architecture and tool represent together the main purpose of this work and are entitled Athena.

Keywords: Computational Intelligence. Artificial Intelligence. Intelligent Systems. Visual Programming. Architecture. Tool.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ACM	*Association for Computing Machinery*
AgNPs	*Nanopartículas de Prata*
AP	*Arquitetura Proposta*
API	*Application Programming Interface*
CE	*Computação Evolutiva*
CISTI	*Iberian Conference on Information Systems and Technologies*
CSV	*Comma-separated values - Arquivo com dados em formato tabular*
DC/UFPI	*Departamento de Computação da UFPI*
EaSII	*Laboratório de Engenharia de Software e Informática Industrial - DC/UFPI*
EC2	*Amazon Elastic Compute Cloud*
EP	*Estudos Primários*
FCL	*Fuzzy Control Language*
FIS	*Fuzzy Inference System*
GAE	*Google App Engine*
GUI	*Graphical User Interface*
HTTP	*HyperText Transfer Protocol*
IA	*Inteligência Artificial*
IC	*Inteligência Computacional*
ICCS	*International Conference on Computational Science*
ICTAI	*International Conference on Tools with Artificial Intelligence*
IE	*Inteligência de Exames*
IEEE	*Institute of Electrical and Electronics Engineers*
IJCNN	*International Joint Conference on Neural Networks*

INPI	*Instituto Nacional da Propriedade Industrial*
JEP	*Java Expression Parser*
MLP	*Multilayer Perceptron*
MSE	*Mapeamento Sistemático de Estudos*
MVC	*Arquitetura Model-View-Controller*
NPs	*Nanopartículas*
PDI	*Processamento Digital de Imagens*
PICOC	*Population, Intervention, Comparison, Outcomes, Context*
PIOC	*Population, Intervention, Outcomes, Context*
PLN	*Processamento de Linguagem Natural*
POO	*Programação Orientada a Objetos*
P&D	*Pesquisa e Desenvolvimento*
QP	*Questão de Pesquisa*
RdP	*Rede de Petri*
RNA	*Redes Neurais Artificiais*
RBF	*Radial Basis Function*
SaaS	*Software as a Service*
SBIC	*Sistemas Baseados em Inteligência Computacional*
SCC	*International Conference on Services Computing*
SCT	*Seleção de Casos de Teste*
SI	*Sistema Inteligente*
SIA	*Sistemas Imunológicos Artificiais*
SF	*Sistemas Fuzzy*
SGBD	*Sistema de Gerenciamento de Banco de Dados*
UFPI	*Universidade Federal do Piauí*

Sumário

Parte I

Introdução

Contexto e Motivação

Desde os primórdios da humanidade, o ser humano busca formas para melhorar sua vida, com o objetivo de garantir e aprimorar sua sobrevivência no mundo. Esse objetivo vem movendo o desenvolvimento de pesquisas nas mais diversas áreas do conhecimento: linguística, economia, medicina, mecânica, química, ecologia, matemática, etc. Cada uma dessas áreas contribui de maneira diferente para alcançar esses objetivos.

A computação, entretanto, tem papel fundamental nessa evolução pois, aliada com as demais áreas, está sendo capaz de resolver problemas antes considerados muito difíceis. A classificação de padrões de escrita e fala (linguística), previsão de ações no mercado financeiro (economia), identificação de anomalias em imagens médicas (medicina), controle de trens de alta velocidade (mecânica), obtenção de novos compostos poliméricos (química), análise entre a influência do clima atual frente à taxa de crescimento das árvores (ecologia), resolução de modelos matemáticos complexos (matemática) e elaboração de novas fórmulas para fabricação de medicamentos (farmácia) são apenas alguns exemplos de problemas que estão sendo solucionados com auxílio da computação (SILVA; SPATTI; FLAUZINO, 2010).

À medida que os estudos em computação foram avançando, os pesquisadores começaram a questionar a possibilidade de expandir os benefícios dessa área, criando máquinas com uma característica intrínseca dos seres humanos: a inteligência. Em 1948, Alan Turing publicou um artigo intitulado *Intelligent Machinery*, no qual refuta uma afirmação antes comumente aceita: *"You cannot make a machine to think for you"* (TURING, 1948). Ainda hoje, muitas das ideias de Turing são tidas como visionárias e estão distantes de se tornarem realidade, entretanto seu trabalho deu origem a uma área de estudos chamada Inteligência Artificial (LUGER, 2004).

A Inteligência Artificial (IA) é a subárea da Ciência da Computação que estuda técnicas para criar Sistemas Inteligentes (SI), ou seja, sistemas cujo comportamento envolve percepção, raciocínio, aprendizado, comunicação e atuação em ambientes complexos (NILSSON, 1998). Além disso, um dos principais objetivos da IA é o desenvolvimento de mecanismos que possam realizar tarefas tão bem quanto os humanos, ou possivelmente ainda melhor (RICH; KNIGHT, 1991).

Entretanto, apesar do grande sucesso obtido pela Inteligência Artificial tradicional, pautada na precisão, certeza e rigor, essa abordagem começou a se mostrar ineficiente em problemas complexos e dinâmicos que exigem uma alta capacidade de adaptação (ZADEH, 1994). Com isso, em meados da década de 1990, Lotfi Zadeh cunhou um novo termo para designar as técnicas que possuem capacidade de serem tolerantes a imprecisão, incerteza e verdade parciais. Essa nova área foi chamada de Inteligência Computacional (IC) (KONAR, 2005).

Por ser uma área relativamente nova, definir a IC não é uma tarefa simples (BEZDEK, 1993; BEZDEK, 1994; EBERHART, 2007; DUCH; MANDZIUK, 2007). Neste trabalho, adotou-se a definição e a taxonomia propostas por Andries Engelbrecht (2007). Segundo Engelbrecht, a Inteligência Computacional é uma subárea da IA concentrada no estudo de mecanismos adaptativos para possibilitar ou facilitar o comportamento inteligente em ambientes complexos e dinâmicos. Essa área reúne diferentes técnicas que exploram a tolerância a precisão, incerteza e verdades parciais para obter flexibilidade, robustez, soluções de baixo custo e boa adequação em problemas reais (ENGELBRECHT, 2007).

As técnicas pertencentes a essa linha de pesquisa dividem-se em cinco subáreas que compartilham uma característica em comum: a inspiração biológica. Essa taxonomia inclui Redes Neurais Artificiais (RNA) inspiradas na estrutura e no funcionamento do sistema nervoso dos seres vivos (HOPFIELD, 1982), Computação Evolutiva (CE) baseada na teoria da evolução proposta por Charles Darwin (FRASER, 1957), Inteligência de Enxames (IE) inspirada no comportamento inteligente observado no convívio social de determinadas populações (BENI; WANG, 1989), Sistemas Imunológicos Artificiais (SIA) inspirados no funcionamento de sistemas imunológicos naturais (CASTRO; TIMMIS, 2002) e Sistemas de Inferência *Fuzzy* (SF) baseados na forma como o ser humano lida com a imprecisão para resolver problemas complexos (ZADEH, 1965).

Essas técnicas representam um paradigma computacional emergente, como pode ser percebido por meio da Figura 1, obtida a partir de uma busca na base de dados *Scopus* utilizando os termos *"Computational Intelligence OR Soft Computing*[1] *"*.

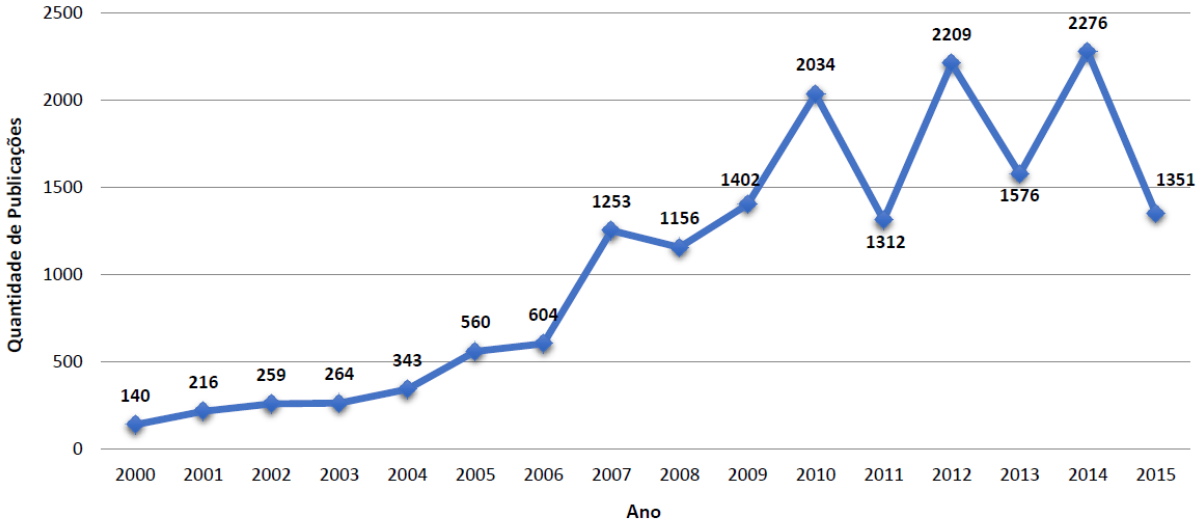


Figura 1 – Número de publicações sobre Inteligência Computacional obtidas na *Scopus*.

[1] Utilizou-se o termo *Soft Computing*, porque segundo Zadeh (1998) (ZADEH, 1998), *Soft Computing* é a base para a Inteligência Computacional e muitas vezes os conceitos estão inter-relacionados.

Entre 2000 e 2015, as publicações envolvendo aspectos teóricos e práticos da IC cresceram em decorrência do sucesso obtido na resolução de diversos problemas reais nas mais variadas linhas de pesquisa, por exemplo:

- **Medicina**: predição de câncer com base no perfil genético do indivíduo (KHAN et al., 2001), classificação de tecidos mamários (BRAZ-JUNIOR et al., 2009), identificação de tumor cerebral a partir das imagens de ressonância magnética (SELVATHI; SELVARAJ; SELVI, 2010), diagnóstico de doença coronária (LAHSASNA et al., 2012), detecção de mácula em imagens de retina (VERAS et al., 2013);

- **Química**: controle do processo de tratamento da água (ZHANG; STANLEY, 1999), modelagem de estruturas em forma de cristal (BAZTERRA; FERRARO; FACELLI, 2002), obtenção de novos compostos poliméricos (ZHANG; FRIEDRICH, 2003);

- **Biologia**: detecção das principais ervas daninhas que atacam as culturas de trigo e soja (EL-FAKI; ZHANG; PETERSON, 2000), identificação e classificação de espécies de morcegos (PARSONS; JONES, 2000) e ratos (TIAN; SHANG, 2006), predição do nível de nitrogênio em plantações de algodão (SUI; THOMASSON, 2006), análise da influência das mudanças climáticas sobre o abastecimento de água para irrigação na região do rio Colorado (Arkansas) (ELGAALI; GARCIA; OJIMA, 2006), análise da influência dos solos calcários na região semiárida do Irã (SHIRANI et al., 2015);

- **Economia**: análise e gerenciamento de risco (RUAN; KACPRZYK; FEDRIZZI, 2001), modelagem de séries financeiras (PAVLIDIS et al., 2006), definição automática de crédito para consumidores (SARLIJA; BENSIC; ZEKIC-SUSAC, 2006);

- **Engenharia Elétrica, Mecânica e Civil**: reconhecimento de distúrbios na qualidade da energia elétrica (ZHU; TSO; LO, 2004), controle de veículos não tripulados (CHO et al., 2006), planejamento das redes de distribuição elétrica (GANGULY; SAHOO; DAS, 2009), análise da resistência à compressão do cimento (THAMMA; BARAI, 2009), maximização dos benefícios hidroelétricos do sistema hidrotérmico brasileiro (RABELO; CARNEIRO; BRAGA, 2009), automação de fábricas (MACHADO; NETO; MELO, 2010);

- **Arquitetura de Computadores**: harware mutável, ou seja, capaz de adaptar sua estrutura de acordo com mudanças no ambiente (IBA; IWATA; HIGUCHI, 1997), projeto de circuitos VLSI (LIENIG, 1997), algoritmo genético aplicado no problema de determinação do leiaute de matrizes de portas (OLIVEIRA; LORENA, 2002);

- **Engenharia de *Software***: controle da qualidade de *software* (EBERT, 1993), reutilização de *software* (MERKL, 1993), predição da eficácia de casos de teste (MAYRHAUSER; ANDERSON; MRAZ, 1995), aplicação de conjuntos *fuzzy* para

melhorar fundamentos do desenvolvimento orientado a objetos (PEDRYCZ; SOSNOWSKI, 1998), seleção dos requisitos a serem incluídos na próxima *release* (ZHANG; HARMAN; MANSOURI, 2007), alocação de equipes ágeis no desenvolvimento de *software* (BRITTO et al., 2012), seleção e priorização de casos de teste (SANTOS-NETO et al., 2012), geração de dados para testes (PATRICK et al., 2015);

- **Mineração de Dados e Aprendizado de Máquina**: mineração de dados em esportes (OMKAR; SENTHILNATH, 2011), classificação de usuários em redes sociais (LIMA; MACHADO, 2012);
- **Segurança**: prevenção de *spams* (CHUAN et al., 2005), combate a crimes cibernéticos (DASGUPTA, 2006), predição de vulnerabilidades em aplicações Web (SHAR; BRIAND; TAN, 2014);
- **Processamento Digital de Imagens (PDI) e Processamento de Linguagens Naturais (PLN)**: detecção de objetos usando a entropia das cores (CHEN; JUANG, 2013), detecção de motociclistas sem capacete em vias públicas (SILVA et al., 2013), análise de sentimento em comentários de produtos na Web (ANCHIETA et al., 2015);
- **Jogos**: aperfeiçoamento de robôs atirador em jogos de primeira pessoa (COLE; LOUIS; MILES, 2004), mapeamento automático de atributos estéticos de personagens (CAMELO; RABELO; PASSOS, 2014).

Esse conjunto de trabalhos representa apenas um retrato da ampla gama de aplicações possíveis para as técnicas de Inteligência Computacional. Esses estudos são motivados pela adequação das técnicas de IC em problemas nos quais abordagens tradicionais, por exemplo modelos matemáticos, são ineficazes ou até mesmo inviáveis. Além disso, as técnicas de IC estão obtendo bons resultados em problemas reais por conta da mimetização de comportamentos inteligentes como o do ser humano (ENGELBRECHT, 2007).

Entretanto, apesar de todas essas possibilidades, a utilização de tais técnicas ainda enfrenta algumas dificuldades que atrasam seu crescimento. Esses empecilhos têm motivado a realização de diversas pesquisas relacionadas ao desenvolvimento de ferramentas que auxiliam a construção de sistemas/aplicações que utilizam técnicas de IC para resolver determinado problema (BRAGA et al., 2015). Essas dificuldades são apresentadas na próxima seção, na qual define-se o problema alvo deste trabalho.

Definição do Problema

Conforme discutido anteriormente, existe um conjunto bem diversificado de aplicações de Inteligência Computacional nos mais diversos setores de pesquisa, entretanto alguns fatores podem dificultar o acesso a essas técnicas. São eles:

- **Alto custo de desenvolvimento (d_1)**: esse fator é quase um consenso entre os pesquisadores que desenvolvem aplicações baseadas em IC (PEER, 2004; PAMPARA; ENGELBRECHT; CLOETE, 2008). Em geral, o desenvolvimento dessas aplicações, sem o auxílio de uma ferramenta, exige um bom conhecimento de programação, além do conhecimento a respeito das técnicas que serão empregadas para solucionar o problema (STOCKT, 2007). Essa dificuldade é ainda mais acentuada para pesquisadores de áreas não ligadas diretamente à Ciência da Computação;

- **Difícil reutilização das implementações (d_2)**: desenvolver aplicações reutilizáveis é um desafio complexo, por isso a maioria das aplicações de IC são fortemente orientadas ao problema alvo. Isso dificulta a reutilização de uma mesma técnica para resolver mais de um problema (PAMPARA; ENGELBRECHT; CLOETE, 2008);

- **A implementação manual é propensa a erros (d_3)**: por conta dos dois fatores supracitados e da dificuldade no entendimento das técnicas de IC, a implementação manual acaba sendo propensa a erros, o que pode gerar resultados inconsistentes e conclusões equivocadas (ALBA et al., 2002);

- **Ferramentas Inadequadas (d_4)**: existem várias ferramentas que buscam auxiliar o trabalho com as técnicas de IC, entretanto muitas dessas ferramentas acabam se tornando inadequadas por negligenciarem fatores como interface gráfica (GUI, do inglês, *Graphical User Interface*), relatórios de execuções inteligíveis e documentação para usuário. Além disso, não foi encontrado outro trabalho que proponha uma ferramenta adaptada a construção de sistemas de Inteligência Computacional Híbridos e que esteja focada na experiência dos usuários finais (pesquisadores e desenvolvedores) (PAREJO et al., 2012);

- **Sistemas Híbridos (d_5)**: desde 1994 já estava claro para Lotfi Zadeh que os Sistemas Inteligentes[2] Híbridos seriam uma linha de pesquisa interessante no futuro (ZADEH, 1994). Essa classe de aplicações combina duas ou mais técnicas para resolver determinado problema, explorando o melhor de cada algoritmo[3]. A principal motivação para o desenvolvimento de Sistemas Inteligentes Híbridos é que uma única técnica, em razão de suas limitações, pode não ser capaz de resolver um dado problema. Por isso, a combinação de várias técnicas pode levar a uma solução mais robusta e eficiente (REZENDE, 2003). Existem várias evidências do sucesso que as abordagens híbridas vem obtendo na solução de diversos problemas (TALBI, 2002; BLUM; ROLI, 2008). Entretanto, a integração das técnicas ainda é complicada e pouco transparente (PAREJO et al., 2012);

[2] Neste trabalho o termo "aplicação ou Sistema Inteligente" será utilizado como sinônimo para "Sistema Baseado em Inteligência Computacional (SBIC)".

[3] O termo "algoritmo" no contexto de IC será utilizado como sinônimo de "técnica de IC".

- **Dificuldades para realizar experimentos (d_6)**: a realização de bons experimentos é um fator crucial para a obtenção de conclusões válidas em qualquer estudo. Em geral, esse é um processo caro e exige muito recurso computacional (processamento e memória). Portanto, a ausência de um ambiente integrado, que permita a realização de experimentos utilizando os benefícios da Computação em Nuvem (escalabilidade, alta disponibilidade e robustez) é mais uma das dificuldades enfrentadas por quem necessita desenvolver Sistemas Baseados em IC (SBIC) (WAGNER et al., 2014);

- **Integração com outros sistemas (d_7)**: a interoperabilidade - capacidade que um sistema tem de trocar informações com outros (semelhantes ou não) de forma transparente - é uma característica importante quando há a necessidade de incorporar técnicas de Inteligência Computacional em processos que já estão em funcionamento. Isso representa mais uma dificuldade enfrentada ao se trabalhar com IC, pois há poucas ferramentas preocupadas com a integração de Sistemas Inteligentes com outros sistemas (PAREJO et al., 2012).

Essas dificuldades podem ser analisadas considerando a seguinte situação: imagine um pesquisador médico que deseja realizar um estudo de classificação e predição de câncer com base no perfil genético do paciente (semelhante ao trabalho proposto por (KHAN et al., 2001)). Suponha também que o pesquisador deseja analisar a combinação de diferentes técnicas de IC e que, ao final da pesquisa, o resultado seja uma aplicação integrada com o sistema que realiza os exames. Além de todas as dificuldades inerentes ao problema estudado, o médico possivelmente precisará superar as dificuldades de desenvolvimento $\{d_1, d_2, d_3, d_4, d_5\}$, de experimentação $\{d_6\}$ e de integração com outros sistemas $\{d_7\}$. Esses desafios não são exclusividade do desenvolvimento de Sistemas Inteligentes, entretanto este trabalho foca no impacto desses obstáculos no âmbito da IC.

Considerando esse contexto, tem-se a seguinte **definição formal do problema**: dado o conjunto das dificuldades $\{d_1, d_2, d_3, d_4, d_5, d_6, d_7\}$, definir uma solução capaz de minimizar o impacto dessas dificuldades no desenvolvimento de Sistemas Inteligentes, considerando o contexto de pesquisadores e profissionais trabalhando com problemas reais. A seguir é apresentada uma visão geral da solução proposta para resolver esse problema.

Visão Geral da Proposta

Este trabalho busca inovar na forma como as aplicações baseadas em IC são desenvolvidas. A proposta é unir as técnicas presentes nessa linha de pesquisa, o conceito de uma arquitetura baseada em componentes e alguns aspectos importantes da Computação em Nuvem para criar um novo conceito: Inteligência Computacional como Serviço (CIaaS, do inglês *Computational Intelligence as a Service* (OLIVEIRA et al., 2014)). Essa proposta

foi inspirada na observação de duas ferramentas bem conceituadas em suas respectivas áreas: CircuitLab[4] (construção de circuitos lógicos digitais) e VisionBlocks (construção de sistemas de visão computacional) (BENDALE et al., 2011).

De posse dessa proposta e do problema definido anteriormente, realizou-se um estudo detalhado sobre as ferramentas existentes. Como resultado dessa etapa, foi possível obter um mapa detalhado dessa linha de pesquisa, além dos requisitos necessários para propor uma solução capaz de auxiliar o desenvolvimento de Sistemas Baseados em IC. Com os requisitos detalhados, foi desenvolvida uma arquitetura com os seguintes princípios: simplicidade, extensibilidade, *software* como serviço, alta performance e colaboração.

Depois que a arquitetura foi concebida, desenvolveu-se uma ferramenta para que fosse possível avaliar a viabilidade da proposta no contexto de pesquisadores e desenvolvedores criando Sistemas Inteligentes para resolver problemas reais. A avaliação experimental da ferramenta apresentou fortes indícios de sua adequação ao problema. Ambas, arquitetura e ferramenta, representam juntas a principal proposta deste trabalho e foram intituladas Athena em homenagem a deusa grega da sabedoria.

Objetivos

O principal objetivo deste trabalho é propor um ambiente integrado (arquitetura e ferramenta) capaz de facilitar a construção, manutenção e aplicação de Sistemas Baseados em Inteligência Computacional. Vale ressaltar que esse objetivo foi definido com o intuito de mitigar as dificuldades apresentadas na Seção *Definição do Problema*.

Para alcançar o objetivo geral, torna-se relevante a inserção dos seguintes objetivos específicos:

1. Apresentar as principais técnicas de Inteligência Computacional, evidenciando suas características fundamentais, vantagens e desvantagens, e áreas de aplicação. Esse estudo foi a base para a definição de um padrão capaz de combinar diferentes técnicas em Sistemas Híbridos;

2. Realizar um mapeamento sistemático sobre ferramentas de IC para identificar os principais trabalhos nessa linha de pesquisa e fornecer uma visão geral em relação às ferramentas que auxiliam o desenvolvimento de aplicações baseadas em IC. Esse estudo serviu para validar e aprimorar as decisões tomadas durante a construção da arquitetura e da ferramenta, pois apresentou lacunas deixadas pelos outros trabalhos (BRAGA et al., 2015);

[4] CircuitLab - pode ser acessada em http://www.circuitlab.com.

3. Especificar uma arquitetura capaz de auxiliar o desenvolvimento de sistemas que aplicam IC para solucionar determinado problema. Essa arquitetura foi desenvolvida de acordo com os seguintes princípios: simplicidade, extensibilidade, *software* como serviço (SaaS, do inglês *Software as a Service*), alta performance e colaboração entre pesquisadores e desenvolvedores;

4. Desenvolver uma ferramenta Web que siga as especificações da arquitetura proposta e que tenha uma preocupação adicional com a experiência do usuário final (OLIVEIRA et al., 2014). Essa ferramenta encontra-se disponível em http://athenasystems.com.br;

5. Avaliar a ferramenta por meio de um estudo experimental realizado no contexto de alunos de graduação, pós-graduação e profissionais desenvolvendo sistemas de IC para resolver problemas reais. Os resultados são conclusivos quanto à adequação da proposta ao problema (OLIVEIRA et al., 2014).

Justificativa

Neste trabalho é proposta uma nova arquitetura para o desenvolvimento de sistemas de IC de forma bastante singular em relação aos trabalhos relacionados. Essa diferenciação foi exigida para que fosse possível criar o conceito de CIaaS. Para isso, uma série de decisões foram tomadas. Nesta seção são explicadas as principais decisões tomadas, de maneira superficial. No Capítulo 3, essas decisões são apresentadas com uma riqueza maior de detalhes.

O desenvolvimento de ferramentas que auxiliam a construção de aplicações baseadas em Inteligência Computacional é uma área bastante explorada atualmente (BRAGA et al., 2015). Essa notoriedade é justificada pela comprovação dos inúmeros benefícios proporcionados pelas técnicas de IC. Entretanto, percebe-se que ainda há diversas lacunas a serem preenchidas nessa linha de pesquisa, por exemplo, ferramentas adaptadas à construção de Sistemas Híbridos. Percebendo essa lacuna, este trabalho propõe um arquitetura que unifica os algoritmos de IC sob um mesmo padrão de funcionamento. Decidiu-se por uma arquitetura baseada em módulos para que fosse possível encapsular a complexidade interna das técnicas, deixando a mostra apenas o que interessa ao usuário final: entradas, configurações e saídas.

Outra área vasta para pesquisa é a construção de ferramentas adaptadas ao contexto de Computação em Nuvem. Esse novo contexto permite um acesso sob demanda a um conjunto de recursos configuráveis (rede, servidores, armazenamento, aplicações e serviços) que podem ser rapidamente alocados e liberados com um esforço mínimo de gerenciamento ou interação com o provedor do serviço (DILLON; WU; CHANG, 2010). Além disso, a Computação em Nuvem traz consigo uma série de vantagens que podem ser estendidas para o desenvolvimento de Sistemas Inteligentes, por exemplo, serviço sob demanda (utilize

apenas quando necessário), ampla rede de acesso e alta disponibilidade (IC acessível de qualquer lugar e a qualquer momento), controle de recursos (poder computacional ajustável para facilitar os experimentos), elasticidade e segurança (JADEJA; MODI, 2012).

O modelo Inteligência Computacional como Serviço foi proposto em contraponto às abordagens tradicionais nas quais os usuários precisam fazer o *download* da ferramenta para desenvolver um Sistema Inteligente. Acredita-se que essa estratégia amplie o acesso às técnicas de IC, uma vez que os usuários podem modelar e executar os sistemas direto no navegador de internet. Outro fator que contribuiu para essa decisão foi a existência de diversos casos de sucesso com um modelo bem similar (*Software* como Serviço), por exemplo, Dropbox, Google Mail, Google Docs, SalesForce, dentre outros.

Por fim, é importante destacar que a Athena não visa substituir ou reimplementar ferramentas como MatLab (VENKATARAMAN, 2009), WEKA (HALL et al., 2009), RapidMiner (HOFMANN; KLINKENBERG, 2013), CILib (PAMPARA; ENGELBRECHT; CLOETE, 2008) ou JMetal (DURILLO; NEBRO; ALBA, 2010). Cada uma dessas ferramentas possui sua particularidade e importância para determinada linha de pesquisa. A Athena busca oferecer uma opção diferenciada para os desenvolvedores que desejam aplicar técnicas de IC para resolver problemas, além de preencher algumas lacunas deixadas por tais ferramentas. Elas podem até ser introduzidas no ambiente da Athena para uso de forma simplificada. Além disso, a arquitetura descrita neste trabalho permite que outros pesquisadores avancem os estudos nessa área.

Contribuições

Destaca-se como principal contribuição deste trabalho a definição de uma arquitetura, capaz de facilitar a utilização de diferentes técnicas de Inteligência Computacional e a construção de uma ferramenta que implementa os conceitos da arquitetura para auxiliar a construção, manutenção e aplicação de Sistemas Baseados em IC. Entretanto, existem outras contribuições, citadas a seguir:

- Realização de um mapeamento sistemático que fornece uma visão geral do estado da arte em relação às ferramentas para desenvolvimento de SBIC, agrupando os resultados e identificando lacunas passíveis de investigação. Além disso, ele facilita a escolha de uma ferramenta para resolver determinado problema, uma vez que exibe características como: técnicas suportadas, plataformas disponíveis, modelos de licenciamento, disponibilidade de interface gráfica, possibilidade de execução paralela e distribuída e a existência de uma API para conexão com outros sistemas;

- A padronização na aplicação das técnicas de IC, facilitando sua integração em Sistemas Híbridos. Essa contribuição é o principal pilar da arquitetura. Além disso,

pode-se destacar a documentação completa da arquitetura, o que possibilita que outros pesquisadores utilizem-na como base para trabalhos futuros;

- Uma ferramenta que permite a construção de Sistemas Inteligentes de forma simples, dinâmica e escalável. A Athena funciona independente de plataforma e permite que seus usuários compartilhem resultados de experimentos com outros pesquisadores ao redor do mundo;

- Uma avaliação experimental conduzida de acordo com as recomendações da Engenharia de *Software* Experimental, na qual foi possível constatar que a Athena pode reduzir os custos associados ao desenvolvimento de Sistemas Baseados em Inteligência Computacional, mantendo o nível de eficácia e elevando a percepção de qualidade.

Parte desta pesquisa já foi publicada em diferentes meios acadêmicos. Em 2014, a Athena foi apresentada em um evento internacional (IEEE ICTAI) relacionado a ferramentas de Inteligência Artificial e em 2015 publicou-se o mapeamento sistemático no Simpósio Brasileiro de Automação Inteligente (SBAI).

- Oliveira, P.; Souza, M.; Braga, R.; Brito, R.; Lira Rabelo, R. e Neto, P. S. **Athena: A Visual Tool to Support the Development of Computational Intelligence Systems**. 26th IEEE International Conference on Tools with Artificial Intelligence (ICTAI). 2014.

- Braga R.; Oliveira, P. A.; Souza, M.; Neto, P. S.; Rabêlo, R. e Britto, B. **Ferramentas para Desenvolvimento de Sistemas Baseados em Inteligência Computacional: Um Mapeamento Sistemático**. XII Simpósio Brasileiro de Automação Inteligente (SBAI). 2015.

Estrutura do Trabalho

O restante deste trabalho está estruturado da seguinte forma: o Capítulo 1 apresenta os conceitos fundamentais que permeiam a Inteligência Computacional. O Capítulo 2 discute os trabalhos relacionados identificados durante a realização do mapeamento sistemático e durante a análise de algumas ferramentas comerciais.

O Capítulo 3 expõe a arquitetura proposta neste trabalho. São detalhadas as diretrizes que guiaram seu desenvolvimento e os artefatos gerados após a conclusão. O Capítulo 4 apresenta detalhes da implementação da ferramenta e exemplos de utilização.

O Capítulo 5 apresenta a avaliação experimental. Por fim, o Capítulo 6 apresenta as conclusões, desafios e limitações, e as perspectivas para continuação da pesquisa.

Parte II

Fundamentação teórica

1 Inteligência Computacional

Em 1956, John McCarthy reuniu 10 pesquisadores no *Dartmouth College* para realizar um seminário sobre uma nova área na computação: a Inteligência Artificial. Esse evento é considerado como o nascimento da IA, pois nele ocorreu o primeiro uso oficial desse termo (RUSSELL; NORVIG, 2013). Entretanto, outros pesquisadores já tinham interesse em desenvolver máquinas com a capacidade de mimetizar o comportamento inteligente percebido nos seres humanos. Por exemplo, McCulloch e Pitts (1943) com a proposição do primeiro modelo matemático de um neurônio artificial (MCCULLOCH; PITTS, 1943), Donald Hebb (1949) com a criação da regra de atualização para modificar a intensidade de conexão entre neurônios artificiais (HEBB, 1949) e os próprios estudos de Alan Turing no artigo intitulado *Intelligent Machinery* (TURING, 1948).

O entusiamo inicial provocado pela possibilidade de criar máquinas com a capacidade de pensar, aprender e criar - cenário comum em filmes de ficção científica - motivou o crescimento dos trabalhos nessa linha de pesquisa e até hoje tem impulsionado avanços na IA. Um desses avanços foi a criação da subárea chamada Inteligência Computacional.

O termo IC foi cunhado por Lotfi Zadeh para designar as técnicas que possuem capacidade de serem tolerantes a imprecisão, incerteza e verdades parciais (KONAR, 2005). Entretanto, ainda hoje não há um consenso a respeito da definição de IC (EBERHART, 2007). Neste trabalho, adotou-se a definição e taxonomia propostas por Engelbrecht. Para ele, a IC concentra seus estudos em mecanismos adaptativos que possibilitam o comportamento inteligente em ambientes complexos e dinâmicos. Essa área subdivide-se em cinco subáreas: Redes Neurais Artificiais, Computação Evolutiva, Inteligência de Enxames, Sistemas Imunológicos Artificiais e Sistemas *Fuzzy* (Figura 2) (ENGELBRECHT, 2007).

Figura 2 – Taxonomia da Inteligência Computacional proposta por Engelbrecht.

1.1 Redes Neurais Artificiais

Redes Neurais Artificiais são modelos computacionais inspirados no sistema nervoso dos seres vivos (HAYKIN et al., 2009). As técnicas pertencentes a essa subárea possuem a capacidade de aquisição e manutenção do conhecimento e podem ser definidas como um conjunto de unidades de processamento, denominados neurônios artificiais, que são interligadas por um grande número de interconexões (SILVA et al., 2013). As principais características relacionadas à aplicação de RNA são a adaptação por experiência, capacidade de aprendizado e tolerância a falhas.

A Figura 3 apresenta os elementos fundamentais de um neurônio artificial: sinais de entrada $(x_1, x_2, ..., x_n)$, pesos sinápticos $(w_1, w_2, ..., w_n)$, combinador linear $(\sum)$, limiar de ativação (θ), potencial de ativação (u), função de ativação (g) e sinal de saída (y) (MCCULLOCH; PITTS, 1943).

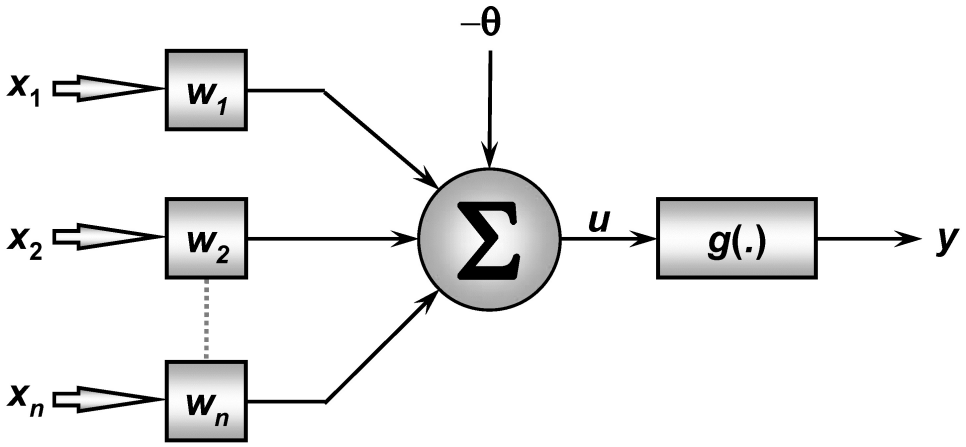


Figura 3 – Modelo de um neurônio artificial.

Os sinais de entrada $(x_1, x_2, ..., x_n)$ representam os valores assumidos pelas variáveis de uma aplicação específica. Esses valores costumam ser normalizados para facilitar o processo de aprendizagem. Os pesos sinápticos $(w_1, w_2, ..., w_n)$ são os valores que servem para ponderar cada uma das variáveis de entrada da rede. Dessa forma, é possível determinar a relevância de cada entrada em relação ao funcionamento do neurônio. O combinador linear $(\sum)$ é responsável por aplicar uma função para agregar todos os sinais de entrada ponderados pelos respectivos pesos sinápticos. O limiar de ativação (θ) é uma variável que especifica o valor necessário para que o neurônio seja ativado. A ativação do neurônio gera o potencial de ativação (u) a ser aplicado na função de ativação (g) com o objetivo de obter o valor final (y) produzido pelo neurônio (SILVA et al., 2013).

Em uma RNA, os neurônios podem ser arranjados ou dispostos de diversas formas. Esse arranjo define a arquitetura da Rede Neural Artificial. As principais arquiteturas de RNA, considerando a disposição de seus neurônios, são: redes *feedforward* de camada simples, redes *feedforward* de múltiplas camadas, redes recorrentes e redes reticuladas.

Considerando essas arquiteturas, os principais tipos de redes são o Perceptron (ROSENBLATT, 1958), o Adaline (WIDROW; HOFF, 1960), o Perceptron Multicamada (MLP, do inglês *Multilayer Perceptron*) (RUMELHART et al., 1988), as redes de base radial (RBF, do inglês *Radial basis function*) (HAYKIN, 2001), a rede de Hopfield e a rede de Kohonen (mapas auto-organizáveis de Kohonen) (KOHONEN, 1982).

As Redes Neurais Artificiais podem ser empregadas em diversos problemas relacionados às engenharias e ciências (SILVA et al., 2013). As potenciais áreas de aplicação são: aproximação de funções, controle de processos (ZHANG; STANLEY, 1999), reconhecimento/classificação de padrões (KHAN et al., 2001), agrupamento de dados (DU, 2010), sistemas de previsão (COAKLEY; BROWN, 2000), otimização de sistemas (VICENTE; CEZARE; SILVA, 2007) e memórias associativas (MICHEL; FARRELL, 1990).

As principais vantagens para a aplicação de RNA são: a possibilidade de construir o conhecimento a partir de uma base de exemplos, grande poder de generalização, não requer um modelo matemático do problema e são tolerantes a falhas (SILVA et al., 2013). No entanto, existem algumas dificuldades que podem atrapalhar sua utilização, por exemplo, ajuste dos parâmetros (*underfitting* e *overfitting*), dificuldade de explicitar os conhecimentos adquiridos pela rede, dentre outros.

Por fim, devido à grande diversidade de algoritmos pertencentes à esta subárea, existem diversas características que podem variar entre diferentes implementações da mesma técnica. Por exemplo, formato dos dados de entrada, condições de parada, organização dos resultados, etc. Em geral, as ferramentas de IC exigem dos usuários a configuração de características como: arquitetura e topologia da RNA, conjunto de treinamento, taxa de aprendizado, precisão requerida e número máximo de iterações. A condição de parada é definida com base na precisão requerida e no número máximo de iterações. Os dados de entrada e saída geralmente são organizados em formato de tabelas.

1.2 Computação Evolutiva

A Computação Evolutiva é a subárea da Inteligência Computacional que reúne uma série de técnicas que inspiradas nos conceitos associados à genética e à teoria da evolução natural proposta por Charles Darwin (FRASER, 1957). As técnicas pertencentes a esta subárea são denominados algoritmos evolutivos e, de forma análoga à evolução natural, esses algoritmos utilizam uma população de indivíduos, na qual cada um é representado por um cromossomo, o qual é composto por um conjunto de genes. Esses genes definem as características dos indivíduos na população. A cada geração, os indivíduos competem pelo direito à reprodução. Aqueles que possuem melhor capacidade de sobrevivência têm maior probabilidade de reproduzir. Os novos indivíduos são gerados a partir da combinação do cromossomo do pai e do cromossomo da mãe. Esse processo é denominado *crossover*. Além

do *crossover*, o cromossomo pode ser alterado por meio de uma mutação genética. Por fim, o poder de sobrevivência de um indivíduo é medido utilizando a função *fitness*, que reflete os objetivos e as restrições do problema a ser resolvido (ENGELBRECHT, 2007).

As principais classes de algoritmos evolutivos são: algoritmo genéticos, programação genética, programação evolutiva, estratégias de evolução, evolução diferencial, evolução cultural e coevolução. Todos os algoritmos evolutivos incorporam o processo básico da teoria da evolução proposta por Charles Darwin (DARWIN, 1978), entretanto os algoritmos genéticos são aqueles que modelam a evolução genética (ENGELBRECHT, 2007). Esse método foi proposto por diversos pesquisadores (*e.g.*, ANDERSON (1953), FRIEDBERG (1958), FRASER (1960), BREMERMANN (1962)), mas a versão mais aceita foi apresentada por HOLLAND (1975) (CASTRO, 2006).

A programação genética, proposta por CRAMER (1985), consiste em um conjunto de técnicas baseadas nos algoritmos genéticos. Entretanto, na programação genética os indivíduos são programas de computadores (representados utilizando a estrutura de árvore) e a avaliação das soluções envolve a execução desses programas. Na programação evolutiva, utiliza-se o conceito de comportamento adaptativo (evolução fenotípica). Dessa forma, a programação evolutiva difere dos algoritmos genéticos e da programação genética devido ao fato de que os indivíduos possuem modelos comportamentais e não modelos genéticos (FOGEL; OWENS; WALSH, 1966).

As estratégias evolutivas representam uma classe de algoritmos evolutivos utilizados principalmente para resolver problemas de otimização de parâmetros (PETERS; KORALEWSKI; ZERBST, 1989; KASPER, 1992; MAGELE et al., 1993). Nas estratégias evolutivas o desenvolvimento dos indivíduos considera tanto as características genéticas quanto as fenotípicas (SCHWEFEL, 1975; CASTRO, 2006). A evolução diferencial também compartilha dos conceitos evolutivos presentes nas demais classes de algoritmos evolutivos. Entretanto, ela se diferencia das demais técnicas no mecanismo de reprodução e na utilização de informações relacionadas à distância e direção da população para guiar o processo de busca (ENGELBRECHT, 2007). Esse método tem sido aplicado principalmente na otimização de funções definidas em espaços contínuos (STORN, 1996; STORN; PRICE, 1997).

A evolução cultural reúne os algoritmos inspirados na evolução da cultura de uma população e como essa cultura influencia no ganho genético e fenotípico dos indivíduos. Isso possibilita uma adaptação mais rápida do que a evolução biológica (ENGELBRECHT, 2007). Esses conceitos vêm sendo aplicados em diversos trabalhos (REYNOLDS; AL-SHEHRI, 1998; OSTROWSKI; REYNOLDS, 1999; JIN; REYNOLDS, 2000; FRANKLIN; BERGERMAN, 2000). Por fim, a Coevolução abriga as técnicas que aplicam o conceito da evolução complementar entre duas espécies diferentes. Por exemplo, em um ambiente que contenha um predador e uma presa existe uma relação inversa entre a aptidão dessas duas

espécies. A vitória de uma espécie significa a derrota da outra. Dessa forma, para sobreviver a espécie "perdedora" adapta-se para combater a espécie "vencedora" com o objetivo de superá-la. Durante esse processo a complexidade de ambas aumenta (ENGELBRECHT, 2007). Outro exemplo de um processo de coevolução é a simbiose, na qual duas espécies diferentes cooperam entre si. A coevolução tem sido aplicada para resolver problemas em diversas áreas (ROSIN, 1997; CHELLAPILLA; FOGEL, 1999; JOHNSON; SUGISAKA, 2000; KEWLEY; EMBRECHTS, 2002).

Além das aplicações que já foram citadas, os algoritmos evolutivos são utilizados em problemas industriais (BACK; FOGEL; MICHALEWICZ, 1997; MORA; SQUILLERO, 2015), por exemplo *hardware* mutável (IBA; IWATA; HIGUCHI, 1997), alocação de equipes de desenvolvimento (BRITTO et al., 2012), otimização do consumo de energia em redes de sensores sem fio (LANZA-GUTIERREZ et al., 2012), dentre outros.

Quanto à implementação dessas técnicas, em geral as ferramenta de IC solicitam ao usuário a definição dos mecanismos e probabilidades de *crossover* e mutação, estratégia de seleção, função *fitness* e o tamanho inicial da população. As condições de paradas comuns definem que o algoritmo deve ser finalizado quando não houver melhorias após determinado número de gerações consecutivas, quando uma solução aceitável for encontrada ou quando o algoritmo alcançar um número máximo de iterações. As entradas e saídas costumam ser representadas como vetores numéricos ou em formato de árvore.

1.3 Inteligência de Enxames

A Inteligência de Enxames reúne os algoritmos inteligentes que têm como inspiração o comportamento social observado em determinadas colônias de indivíduos. Esses algoritmos caracterizam-se pela presença de um enxame de agentes computacionais que possuem a capacidade de perceber e modificar o ambiente de maneira local. Essa capacidade torna possível a troca de informações entre os indivíduos, viabilizando a coordenação de ações para solucionar determinado problema. É importante ressaltar que não há uma coordenação centralizada de ações e nem um modelo explícito do ambiente. Dessa forma, o conjunto das ações individuais leva ao surgimento de um comportamento global capaz de solucionar determinado problema (ENGELBRECHT, 2007).

As técnicas mais conhecidas dessa linha de pesquisa são: otimização por colônia de formigas (ACO, do inglês *Ant Colony Optimization*) (DORIGO; MANIEZZO; COLORNI, 1996), que utiliza o comportamento da coleta de alimentos pelas formigas; otimização por enxame de partículas (PSO, do inglês *Particle Swarm Optimization*) (KENNEDY; EBERHART, 1995), baseado no comportamento social de pássaros em um bando; e os algoritmos de colônia de abelhas (KARABOGA; BASTURK, 2007) aptos a solucionar problemas de otimização por meio de mecanismos inspirados no comportamento das

abelhas.

Esses algoritmos são aplicados em problemas de classificação de imagens (OMRAN; SALMAN; ENGELBRECHT, 2002), controle de sistemas elétricos (FUKUYAMA; YOSHIDA, 2001; ANNALURU; DAS; PAHWA, 2004), roteamento em redes de comunicações (CARO; DORIGO, 1998), definição de rota em missões de vigilância (SECREST, 2001), problemas de atribuição de tarefas (SALMAN; AHMAD; AL-MADANI, 2002), mineração de dados (SOUSA; NEVES; SILVA, 2003), bioinformática (MEKSANGSOUY; CHAIYARATANA, 2003; NEETHLING; ENGELBRECHT, 2006), dentre outros.

Quanto à implementação dessas técnicas, a entrada costuma ser representada em formato de grafo para problemas de otimização discreta ou na forma de uma equação para problemas de otimização contínua. As configurações, em geral, definem o tamanho da população, a função *fitness* e a heurística utilizada para guiar o processo de busca. Por fim, as condições de parada comuns definem que o algoritmo deve ser finalizado quando não houver melhorias após determinado número de gerações consecutivas (estagnação) ou quando um número máximo de iterações for alcançado.

1.4 Sistemas Imunológicos Artificiais

O sistema imunológico dos vertebrados é composto por um conjunto de células, moléculas e organismos que juntos desempenham a importante tarefa de manter um estado de equilíbrio interno no corpo desses seres vivos (CASTRO, 2006). Esse sistema possui a capacidade de reconhecer células invasoras (antígeno); controlar a ação dos órgãos de defesa do corpo; influenciar o comportamento de outros sistemas; e aprender como combater os agentes causadores de doenças (ENGELBRECHT, 2007). Esse mecanismo serve de inspiração para as técnicas de IC pertencentes à subárea conhecida pelo termo Sistemas Imunológicos Artificiais (SIA) (CASTRO; TIMMIS, 2002).

Quando o corpo é invadido por um antígeno, o sistema imunológico natural aciona as células de defesa do organismo. Cada uma dessas células possui a capacidade de reconhecer e combater determinado invasor. Esse processo de reconhecimento estimula a reprodução das células que possuem maior afinidade com o invasor, ou seja, maior capacidade de reconhecê-lo. Nesse caso, a reprodução é assexuada e as novas células podem sofrer mutações com o objetivo de aperfeiçoar seu poder de reconhecimento. A probabilidade de mutação é inversamente proporcional ao nível de afinidade com o antígeno. Isso sugere que as células com baixa afinidade terão maior probabilidade de sofrerem mutações. Após o fim do combate ao antígeno, algumas células de defesa são recrutadas para compor o banco de memória do sistema imunológico. Assim, em uma invasão futura, o organismo poderá reagir de forma mais eficiente e eficaz (CASTRO, 2006).

O número de aplicações e algoritmos presentes na literatura sobre SIA é vasto,

entretanto apenas algumas poucas ideias têm sido amplamente exploradas: seleção clonal e maturação de afinidade, seleção negativa (mecanismo presente no órgão de defesa conhecido como Timo) e redes imunológicas. Os algoritmos mais conhecidos dessa linha de pesquisa são: CLONALG (CASTRO; ZUBEN, 2000), algoritmo de seleção negativa (FORREST et al., 1994), SIA multicamadas (KNIGHT; TIMMIS, 2002), AINet (CASTRO; ZUBEN, 2001a), teoria do perigo (MATZINGER, 1994; AICKELIN; CAYZER, 2008) e algoritmo de células dendríticas (GREENSMITH; AICKELIN; CAYZER, 2005).

Sistemas Imunológicos Artificiais estão sendo aplicados com sucesso em problemas de vários domínios, por exemplo, detecção de anomalias e invasões em redes de computadores (DASGUPTA; FORREST, 1999; GONZALEZ; DASGUPTA; KOZMA, 2002), classificação de modelos (PRAMANIK; KOZMA; DASGUPTA, 2002), detecção de vírus de computador (FORREST et al., 1994), agrupamento de dados (CASTO; ZUBEN, 2000), robótica (JUN; LEE; SIM, 1999), reconhecimento de padrões e mineração de dados (HUNT; COOKE, 1996; TIMMIS; NEAL; HUNT, 1999; TIMMIS, 2000) e sistemas de recomendação (CAYZER; AICKELIN, 2005). Além disso, é possível utilizar técnicas de SIA para a inicialização dos pesos em RNA do tipo *feedfoward* (CASTRO; ZUBEN, 2001c), inicialização dos centros das redes RBF (CASTRO; ZUBEN, 2001b) e otimização de funções multi-modal (CASTRO; TIMMIS, 2002).

Quanto à implementação das técnicas, em geral os dados de entrada são representados como vetores numéricos ou cadeias de caracteres. As configurações divergem muito entre as técnicas, entretanto, devido ao processo de treinamento supervisionado adotado pela maioria dos algoritmos, é necessário informar um conjunto de treinamento. A principal condição de parada é o número máximo de iterações.

1.5 Sistemas *Fuzzy*

O termo *fuzzy* pode ter diversos significados, de acordo com o contexto. Entretanto, o conceito básico desse adjetivo passa sempre pelo vago, indistinto e incerto (REZENDE, 2003). Dessa forma, a subárea da Inteligência Computacional denominada Sistemas *Fuzzy* contempla modelos computacionais que lidam com informações imprecisas, incompletas, vagas ou incertas (ENGELBRECHT, 2007). Essas técnicas são inspiradas no raciocínio dos seres humanos, os quais são capazes de lidar com processos complexos, baseados em informações imprecisas ou aproximadas (TANSCHEIT, 2004). Em geral, essas informações são expressas por meio de termos linguísticos ao invés de números exatos. Por exemplo, nas frases “A sala está pouco iluminada” e “Ele é muito alto” as palavras *pouco* e *muito* são termos linguísticos que descrevem a magnitude das variáveis *fuzzy iluminação* e *altura* (ENGELBRECHT, 2007).

Os Sistemas *Fuzzy* estão fundamentados em dois grandes pilares: a teoria de

conjuntos *fuzzy* (ZADEH, 1965) e a lógica *fuzzy* (ZADEH, 1975). A teoria dos conjuntos *fuzzy* pode ser vista como uma extensão da teoria clássica de conjuntos e foi criada para tratar graus de pertinência intermediários entre a pertinência total e a não-pertinência de elementos de um universo de discurso (REZENDE, 2003). Assim, um conjunto *fuzzy* é uma generalização da noção básica de um conjunto clássico (*crip*) (ROSS, 2009). A lógica *fuzzy*, por sua vez, utiliza a fundamentação matemática descrita na teoria de conjuntos *fuzzy* para atuar como lógica aplicada ao raciocínio aproximado. Na lógica clássica, lida-se com proposições que podem ser verdadeiras ou falsas. Já na lógica *fuzzy*, as proposições possuem um grau de pertinência que está associado ao seu grau de verdade (ENGELBRECHT, 2007).

As técnicas mais conhecidas dessa linha de pesquisa utilizam regras de produção da forma "*se* <*antecedente*> *então* <*consequente*>" para armazenar informações em uma base de conhecimento com o objetivo de criar modelos de inferência *fuzzy*. Esses modelos possuem a capacidade de processar informações imprecisas por meio do raciocínio aproximado. Três etapas definem o funcionamento básico dos sistemas de inferência *fuzzy*: *fuzzificação*, processo de inferência e *defuzzificação*. Durante a *fuzzificação*, as variáveis de entrada (numéricas) são convertidas em variáveis *fuzzy*, considerando as definições dos conjuntos *fuzzy* de entrada. Depois o mecanismo de inferência é acionado para identificar quais regras foram ativadas e qual será o valor *fuzzy* resultante. Por fim, o processo de *defuzzificação* associa o resultado *fuzzy* a um valor numérico final. A Figura 4 apresenta um modelo genérico de um sistema de inferência *fuzzy*.

As principais técnicas são o sistema de inferência *fuzzy* de Mamdani (MAMDANI, 1977) e o sistema de inferência *fuzzy* de Takagi-Sugeno-Kang (TAKAGI; SUGENO, 1985; SUGENO; KANG, 1988). Entretanto, além desses modelos, existem outros algoritmos que utilizam conceitos da lógica *fuzzy* para resolver diversos tipos de problemas, por exemplo, o *Fuzzy* C-Means (BEZDEK; EHRLICH; FULL, 1984) para problemas de agrupamento de dados.

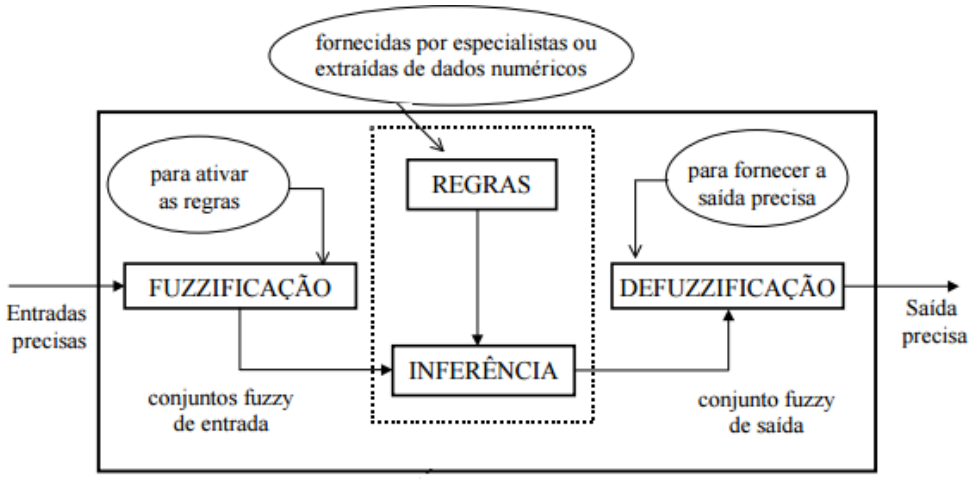


Figura 4 – Modelo de um sistema de inferência *fuzzy* (TANSCHEIT, 2004).

Esses algoritmos vêm sendo aplicados na resolução de problemas que envolvem aproximação de funções (KOSKO, 1992), controle de processos (EBERT, 1993; FAVILLA; MACHION; GOMIDE, 1993; RABELO; CARNEIRO; BRAGA, 2009), classificação e agrupamento de dados (BRIDGES; VAUGHN, 2000; CHEN; JUANG, 2013), no desenvolvimento de sistemas de apoio à decisão (NGAI; WAT, 2005) e Sistemas Híbridos (JEONG; OH, 1999; SHI; EBERHART, 2001), e em outras linhas de pesquisa, por exemplo, bioinformática (PENA-REYES; SIPPER, 2000; LAHSASNA et al., 2012), desenvolvimento de jogos (CAMELO; RABELO; PASSOS, 2014), dentre outros.

Quanto à implementação dessas técnicas, em geral, as ferramentas de IC que trabalham com sistemas de inferência *fuzzy* exigem a definição de um arquivo de configuração chamado FCL (do inglês, *Fuzzy Control Language*). Esse arquivo contém o detalhamento do controle *fuzzy*: variáveis de entrada e saída, funções de pertinência, base de regras, etc. As entradas e saídas costumam ser representadas por valores numéricos. Por fim, como geralmente esses sistemas não exigem treinamento, o algoritmo avalia as entradas, retorna as saídas e finaliza sua execução.

1.6 Abordagens Híbridas

Desde 1994, já estava claro para Lotfi Zadeh que os Sistemas Inteligentes Híbridos seriam uma linha de pesquisa interessante no futuro (ZADEH, 1994). Essa classe de aplicações combina duas ou mais técnicas distintas para solucionar determinado problema, explorando o melhor de cada algoritmo. A principal motivação para o desenvolvimento de Sistemas Inteligentes Híbridos refere-se ao fato de que uma única técnica, em razão de suas limitações e/ou deficiências, pode não ser capaz de resolver um dado problema. Dessa forma, a combinação de várias técnicas pode levar a uma solução mais robusta, entretanto não há garantias de que haverá uma melhora no desempenho do sistema como um todo (REZENDE, 2003).

Em geral, a maioria das pesquisas tem combinado técnicas baseadas em dados (*e.g.*, RNA) com técnicas que utilizam conhecimento (*e.g.*, sistemas *fuzzy*). A seguir, apresenta-se exemplos de combinações entre diferentes métodos: regras *fuzzy* podem ser utilizadas para inicializar a estrutura de uma RNA a fim de acelerar o treinamento e melhorar o processo de generalização; RNAs podem ser utilizadas para construir ou aprimorar uma base de regras *fuzzy*; algoritmos de busca local podem ser integrados com meta-heurísticas para melhorar o processo de otimização; algoritmos evolutivos podem ser usados para ajustar a topologia e os pesos de uma RNA; dentre outros.

Os Sistemas Híbridos podem ser classificados em três grupos: substituição de função (tipo I), híbridos intercomunicativos (tipo II) e híbridos polimórficos (tipo III) (GOONATILAKE; KHEBBAL, 1994). Os Sistemas Híbridos do tipo I utilizam uma técnica

para implementar determinada função de outra técnica. Por exemplo, o uso de árvores de decisão para indexar a base de casos de um sistema baseado em casos (CARDIE, 1993). Nos sistemas do tipo II as técnicas são utilizadas para resolver problemas complexos que possam ser subdivididos em tarefas independentes. Dessa forma, cada algoritmo fica responsável por uma das subtarefas. Por exemplo, o sistema EITHER que utiliza módulos independentes de raciocínio indutivo, dedutivo e abdutivo para revisar cláusulas de Horn (MOONEY; OURSTON, 1994). Cada tipo de raciocínio foi implementado com uma técnica. Por fim, os Sistemas Híbridos polimórficos definem que uma técnica deve ser adaptada para realizar uma tarefa inerente a uma outra técnica, com o objetivo de descobrir novas funcionalidades de uma técnica e entender como diferentes técnicas podem se relacionar (REZENDE, 2003). Por exemplo, aplicação de RNA para manipulação de regras (GOONATILAKE; KHEBBAL, 1994).

Além dessas três classes de Sistemas Híbridos, pode-se notar o surgimento de outro grupo, no qual o problema é subdividido em tarefas interdependentes e cada tarefa é solucionada por uma técnica específica. Esse novo grupo de sistemas é bastante semelhante aos sistemas do tipo II, entretanto as técnicas são interconectadas e a execução de uma tem influência no resultado da outra (*e.g.*, Sistema Híbrido para alocação de equipes de desenvolvimento ágil (SANTOS-NETO et al., 2012)).

Atualmente, existem várias evidências do sucesso que as abordagens híbridas vêm obtendo na solução de diversos problemas (TALBI, 2002). Entretanto, a integração das técnicas ainda é complicada e pouco transparente (PAREJO et al., 2012).

2 Trabalhos Relacionados

Na literatura são encontrados alguns trabalhos que têm o objetivo de auxiliar a utilização de técnicas de Inteligência Computacional. Este capítulo discute os trabalhos identificados por meio de um Mapeamento Sistemático (KEELE, 2007), além de algumas ferramentas relevantes que não foram incluídas no mapeamento. Por fim, são apresentadas considerações que ajudam a posicionar este trabalho dentro dessa linha da pesquisa.

2.1 Mapeamento Sistemático de Estudos

Apesar da ampla gama de aplicações possíveis para as técnicas de IC e da quantidade razoável de ferramentas que se propõem a auxiliar o desenvolvimento de SBIC, não há um trabalho sistemático (Revisão Sistemática de Literatura ou Mapeamento Sistemático de Estudos) que reúna evidências sobre tais ferramentas, apresentando suas principais características e oferecendo um mapa completo dessa linha de pesquisa.

Esta seção apresenta os resultados de um Mapeamento Sistemático de Estudos (MSE) (HIGGINS; GREEN et al., 2008) realizado com o objetivo de identificar as ferramentas existentes que oferecem apoio à construção de Sistemas Baseados em Inteligência Computacional.

2.1.1 Metodologia

Um MSE é um método de pesquisa secundário que permite identificar de forma sistemática a literatura em determinada área de conhecimento, apresentando, na maioria das vezes, um resumo visual (mapa) dos resultados encontrados (PETERSEN et al., 2008).

Os resultados do MSE apresentados nesse trabalho seguem um conjunto de diretrizes definidas por KITCHENHAM; PICKARD; PFLEEGER (1995) e implementa o processo proposto por PETERSEN et al. (2008), com o objetivo de identificar e sumarizar as ferramentas existentes que oferecem suporte ao desenvolvimento de SBIC. A Figura 5 apresenta as etapas seguidas na execução do processo de Mapeamento Sistemático realizado. A seguir discute-se o protocolo com as diretrizes que nortearam esse estudo.

2.1.2 Questões de Pesquisa

Buscando direcionar esforços durante o levantamento e compreensão dos Estudos Primários (EP) sobre ferramentas para apoio ao desenvolvimento de Sistemas Baseados em IC, a seguinte questão de pesquisa guiou este trabalho:

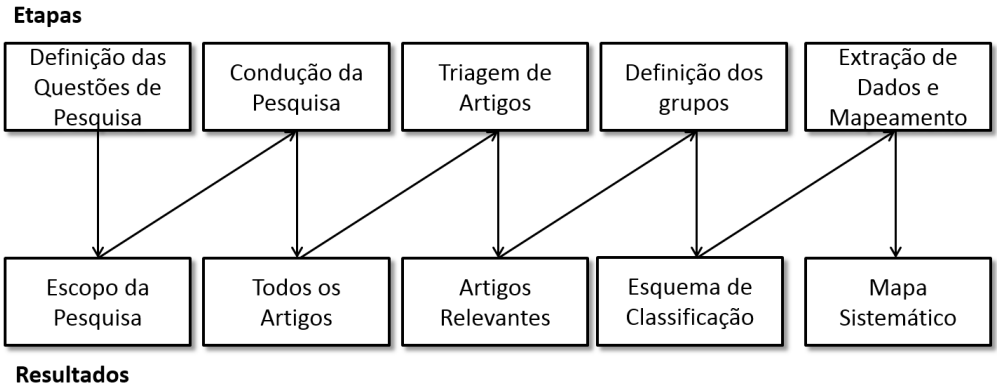


Figura 5 – Etapas do Mapeamento Sistemático (adaptada de (PETERSEN et al., 2008)).

QP1: Quais ferramentas apoiam pesquisadores e profissionais no design/implementação de aplicações baseadas em Inteligência Computacional? Essa questão motivou a elaboração das seguintes subquestões:

a) Quais técnicas de IC são suportadas?
b) Em quais plataformas está disponível?
c) Quais são os modelos de licenciamento?
d) É possível a construção de Sistemas Híbridos?
e) Possui interface gráfica para o usuário (GUI)?
f) Permite execução paralela e distribuída?
g) Possui API para conexão com outros sistemas?

Com a resolução da QP1 pretende-se entender a literatura disponível em relação às ferramentas que auxiliam os profissionais e pesquisadores no desenvolvimento de Sistemas Baseados em Inteligência Computacional, extraindo as informações necessárias para responder cada uma das subquestões motivadas por essa questão de pesquisa.

Essa Questão de Pesquisa foi obtida a partir da adaptação do modelo PICOC (*Population*, *Intervention*, *Comparison*, *Outcomes*, *Context*) (PETTICREW; ROBERTS, 2008), que define que o pesquisador deve considerar os seguintes componentes na elaboração da QP: população, intervenção, comparação, resultados e contexto. Entretanto, o critério de comparação não foi utilizado, pois o foco do estudo não é comparar intervenções. Foram utilizados os critérios população, intervenção, resultados e contexto (PIOC):

- **População:** pesquisadores e profissionais que atuam na área de IC;
- **Intervenção:** ferramentas que auxiliam o desenvolvimento dos projetos de IC;
- **Resultados:** ferramentas de IC, classificadas de acordo com as subquestões;
- **Contexto:** projetos que utilizam técnicas de IC na academia ou na indústria.

2.1.3 Estratégia de Busca

A estratégia de busca consiste na formação da *string* de busca a partir dos termos extraídos das questões de pesquisa e na sua aplicação nos mecanismos de busca das bibliotecas digitais selecionadas. As etapas realizadas nesse MSE são discutidas a seguir.

2.1.3.1 String de Pesquisa

O processo de construção da *string* de busca foi realizado em 3 etapas: (E1) inicialmente foram identificados os termos chaves contidos nos critérios população e intervenção do modelo PIOC; (E2) foram identificados sinônimos ou palavras alternativas para cada termo encontrado em E1 e utilizou-se o conectivo *OR* para ligar os termos; (E3) após a realização de E2, realizou-se uma combinação simples entre os termos que definem a população e a intervenção, e os termos resultantes foram ligados por meio do conectivo *OR*. A Tabela 1 apresenta o resultados de cada uma dessas etapas.

Tabela 1 – Etapas da construção da *string* de busca.

Etapa	**Resultado**
E1	População: Computational Intelligence Intervenção: tool
E2	População: "Computational Intelligence" *OR* "Artificial Intelligence" *OR* "Metaheuristic" *OR* "Soft Computing" Intervenção: "tool" *OR* "framework" *OR* "service" *OR* "library" *OR* "API"
E3	"Computational Intelligence Framework" *OR* "Artificial Intelligence Framework" *OR* "Metaheuristic Framework" *OR* "Soft Computing Framework" *OR* "Computational Intelligence tool" *OR* "Artificial Intelligence tool" *OR* "Metaheuristic tool" *OR* "Soft Computing tool" *OR* "Computational Intelligence service" *OR* "Artificial Intelligence service" *OR* "Metaheuristic service" *OR* "Soft Computing service" *OR* "Computational Intelligence library" *OR* "Artificial Intelligence library" *OR* "Metaheuristic library" *OR* "Soft Computing library" *OR* "Computational Intelligence api" *OR* "Artificial Intelligence api" *OR* "Metaheuristic api" *OR* "Soft Computing api"

Realizou-se uma pesquisa piloto nas bibliotecas digitais *Scopus*, *Compendex* e *Web of Science* com o objetivo de verificar se a pesquisa abrangeria trabalhos relevantes da área. Com isso, observamos que novos termos poderiam ser incluídos na *string* de pesquisa e assim tornar a pesquisa mais abrangente. A *string* final utilizada na busca e sua versão adaptada para cada biblioteca digital estão disponíveis no Apêndice A e pelo link https://goo.gl/bKVqOR.

2.1.3.2 Processo de Busca

O processo de busca por EP foi realizado aplicando a *string* de pesquisa em diferentes bibliotecas digitais utilizando os próprios mecanismos de busca avançada de cada uma delas. As bases de dados utilizadas na busca foram: *Scopus*, *Compendex* e *Web of Science*, pois elas cobrem as bases de dados mais importantes no que diz respeito às Engenharias e às Ciências Exatas/Aplicadas: *IEEE Xplore*, *ACM Digital Library*, *Elsevier* e *Springer*. O período de tempo foi limitado entre os anos de 2000 a 2015.

Como resultado dessa etapa de busca foram encontrados 907 trabalhos. A ferramenta *StArt* (HERNANDES et al., 2012) foi utilizada para facilitar a remoção dos trabalhos duplicados e, com isso, obteve-se um total de 571 candidatos a Estudos Primários. O resultado da pesquisa em cada biblioteca digital pode ser observado na Tabela 2.

Tabela 2 – Resultados da busca nas bibliotecas digitais.

Biblioteca Digital	**Resultado**
Compendex	178
Scopus	564
Web of Science	165
Total	**907**
Total sem duplicações	**571**

2.1.4 Critérios de Seleção dos Estudos

Os critérios para inclusão e exclusão de EP foram definidos em conformidade com os objetivos e questões de pesquisa.

2.1.4.1 Critérios de Inclusão

Decidiu-se que seriam incluídos na pesquisa os EPs que:

a) Fossem escritos em Inglês *AND*;

b) Estivessem publicados em workshop, conferência, revista ou jornal entre os anos de 2000 e 2015 *AND*;

c) Apresentassem uma ferramenta que auxilie o desenvolvimento de Sistemas Baseados em Inteligência Computacional *AND*;

d) Apresentassem as técnicas de Inteligência Computacional suportadas pela ferramenta.

2.1.4.2 Critérios de Exclusão

Decidiu-se que seriam excluídos da pesquisa os Estudos Primários que:

a) Não fossem escritos em Inglês *OR*;

b) Não estivessem publicados em workshop, conferência, revista ou jornal entre os anos de 2000 e 2015 *OR*;

c) Não apresentassem uma ferramenta que auxilie o desenvolvimento de Sistemas Baseados em Inteligência Computacional *OR*;

d) Estivessem disponíveis somente em forma de resumos ou apresentações.

2.1.5 Processo de Seleção dos Estudos

O processo de seleção dos estudos foi realizado em três etapas e com a participação de quatro pesquisadores: (A) o autor desta dissertação, (B) um aluno de Iniciação Científica do curso de Ciência da Computação da UFPI, (C) o orientador e (D) o coorientador do trabalho. A justificativa para essa composição é discutida na Subseção 2.1.8. As etapas da seleção foram as seguintes:

a) **Leitura de títulos e resumos:** nessa etapa, dois pesquisadores (A e B) aplicaram manualmente os critérios de inclusão e exclusão no título e resumo de todos os 571 trabalhos candidatos a EP identificados durante a pesquisa. Desses 571 trabalhos, 50 foram aceitos pelos dois pesquisadores e em 8 trabalhos houve discordância entre os pesquisadores quanto a seleção. Esses foram reavaliados por outros dois pesquisadores (C e D) de modo que decidiu-se pela exclusão dos trabalhos;

b) **Leitura completa dos trabalhos:** nessa etapa, dos 50 trabalhos aceitos, 13 foram removidos por estarem disponíveis somente na forma de *abstracts* ou resumos expandidos e 21 foram removidos após a leitura das seções de introdução e conclusão, resultando em 16 trabalhos aceitos;

c) **Análise de referências:** realizou-se uma inspeção nas referências bibliográficas dos trabalhos aceitos e decidiu-se pela inclusão de 8 artigos. Essa técnica é bastante utilizada em MSE e esses foram submetidos ao mesmo processo de seleção dos demais artigos.

2.1.6 Processo de Extração dos Dados

A extração das informações relevantes foi realizada com o auxílio de planilhas eletrônicas contendo campos gerais (título, autores, ano, etc.) e campos específicos para cada questão de pesquisa (técnicas de IC suportadas, plataforma, licenciamento, etc.). Os dados foram sintetizados para cada questão de pesquisa.

2.1.7 Resultados e Discussões

Esta seção apresenta os resultados do Mapeamento Sistemático de Estudos. A partir de um conjunto inicial de 571 trabalhos, foram identificados 24 estudos primários (Tabelas 3 e 4) que respondem as questões de pesquisa deste mapeamento.

A fim de ilustrar potenciais pontos de futura investigação, apresenta-se um mapa da área (Figura 6) com gráficos de bolhas bidimensionais, no qual o diâmetro da bolha é determinado pela quantidade de publicações correspondentes às coordenadas (X, Y). Por exemplo, a Figura 6 indica que apenas 2 estudos (8,3% do total de trabalhos) são ferramentas de IC adaptadas à plataforma *Web*.

Tabela 3 – Estudos Primários selecionados - Informações Gerais.

Estudos Primários	**Grupo de Pesquisa**	**País**	**Evento**	**Ano***	**Qualis****
Athena(OLIVEIRA et al., 2014)	Lab. EaSII - Universidade Federal do Piauí	Brasil	IEEE ICTAI	2014	A2
jMetalCpp(LÓPEZ-CAMACHO et al., 2013)	*University of Málaga*	Espanha	*Bioinformatics (Oxford)*	2013	A1
Weyland(WEYLAND; MONTEMANNI; GAMBARDELLA, 2013)	*Lab. IDSIA*	Suíça	*Journal of Parallel and Distributed Computing*	2013	A2
Optimus(SREEPATHI; MAHINTHAKUMAR, 2012)	*Oak Ridge National Lab.*	EUA	IEEE SCC	2012	-
PMF(LEE; TAHERI; ZOMAYA, 2012)	*University of Sydney*	Austrália	*Journal of Foundations of Computer Science*	2012	B2
jMetal(DURILLO; NEBRO, 2011)	*University of Málaga*	Espanha	*Advances in Engineering Software*	2011	B1
EvA2(KRONFELD; PLANATSCHER; ZELL, 2010)	*Universität Tübingen*	Alemanha	*Learning and Intelligent Optimization*	2010	B4
I-om(MOREIRA; REIS; SOUSA, 2010)	Inst. Politécnico de Viana do Castelo	Portugal	IEEE CISTI	2010	B4
Nyx(LACKOVIĆ, 2010)	*University of Zagreb*	Croácia	*Journal of Information and Organizational Sciences*	2010	-
Rosemberg(ROSENBERG et al., 2010)	*CSIRO ICT Centre*	Austrália	IEEE SCC	2010	B1
CIlib(PAMPARA; ENGELBRECHT; CLOETE, 2008)	*University of Pretoria*	África do Sul	IEEE IJCNN	2008	A2
IT2FLS(CASTILLO; MELIN, 2008)	*Tijuana Institute of Technology*	México	IEEE IJCNN	2008	A2
AFRANCI(REINALDO et al., 2006)	UTFPR - Paraná	Brasil	International Conference on Computers	2006	-
HeuristicLab(WAGNER; AFFENZELLER, 2005)	*University of Applied Sciences*	Áustria	*Adaptive and Natural Comp. Algorithms*	2005	-
Ellen(ELLEN; CAMPBELL, 2005)	*Queensland University of Technology*	Austrália	*Inter. Conf. on Comp. Intelligence*	2005	-
MAGMA(MILANO; ROLI, 2004)	*University of Bologna*	Itália	*Systems, Man, and Cybernetics*	2004	-
ParadisEO(CAHON; MELAB; TALBI, 2004)	*Lille University*	França	Journal of Heuristics	2004	A2
FOM(PAREJO et al., 2003)	*University of Sevilla*	Espanha	Springer ICCS	2003	A2
Zha(ZHA, 2003)	*Nanyang Technological University*	Singapura	*Soft Computing*	2003	A2
HOTFRAME(FINK; VOSS, 2002)	*University of Hamburg*	Alemanha	*Optimization Software Class Libraries*	2002	-
MALLBA(ALBA et al., 2002)	*University of Málaga*	Espanha	*Euro-Par Parallel Processing*	2002	B2
HSF(DORNE; VOUDOURIS, 2001)	*Intelligent Complex Systems Research Group*	Reino Unido	*Metaheuristics International Conference*	2001	-
iOpt(VOUDOURIS et al., 2001)	*University of Angers*	França	*Principles and Practice of Constraint Programming*	2001	A2
MAFRA(KRASNOGOR; SMITH, 2000)	*University of the West of England*	Reino Unido	-	2000	-

Nota: * Os trabalhos foram ordenados de acordo com o ano de publicação. ** Considerando a última classificação de periódicos emitida pela CAPES.

Tabela 4 – Estudos Primários selecionados - Informações relacionadas à Questão de Pesquisa (QP1).

Estudos Primários*	**Contribuição**	**Técnicas**	**Plataforma**	**Hibridização**	**Interface Gráfica**	**Execução Paralela e Distribuída**	**API**
AFRANCI	Ferramenta	RNA	Desktop	Sim	Sim	Sim	Sim
Athena	Ferramenta	CE; IE; SIA; SF	Web	Sim	Sim	Sim	Sim
HeuristicLab	Ferramenta	CE; IE	Desktop	Sim	Sim	-	Sim
I-om	Ferramenta	CE	Web	-	-	Sim	Sim
IT2FLS	Ferramenta	RNA; SF	Desktop	Não	Sim	Não	Não
Optimus	Ferramenta	IE	Desktop	Não	-	Sim	Não
CIlib	Framework	CE; RNA; IE; SIA; SF	Desktop	Sim	Não	Não	Não
Ellen	Framework	SF; RNA	Desktop	Sim	-	Sim	Não
EvA2	Framework	CE; IE	Desktop	Não	Sim	Não	Não
FOM	Framework	CE	Desktop	Sim	-	Não	Não
HSF	Framework	CE	Desktop	Sim	Não	-	Não
HOTFRAME	Framework	CE	Desktop	Sim	Não	-	Não
iOpt	Framework	CE	Desktop	Sim	Sim	Sim	Não
jMetal	Framework	CE; IE	Desktop	Não	Não	Não	Não
jMetalCpp	Framework	CE; IE	Desktop	Sim	Não	Sim	Não
MAFRA	Framework	CE	Desktop	Não	Não	Não	Não
MAGMA	Framework	IE	Desktop	Sim	Sim	Não	Não
MALLBA	Framework	CE	Desktop	Sim	Não	Sim	Não
Nyx	Framework	CE	-	-	Não	Não	Não
ParadisEO	Framework	CE; IE	Desktop	Sim	Não	Sim	-
PMF	Framework	CE; SIA	Desktop	Sim	-	Sim	-
Rosemberg	Framework	CE	Desktop	Sim	Não	Sim	-
Weyland	Framework	CE; IE	-	-	Não	Sim	Não
Zha	Framework	RNA; SF	Desktop	Sim	Sim	Não	Não

Nota: * Nessa tabela os estudos primários foram ordenados segundo dois critérios: tipo de contribuição e ordem alfabética considerando o nome ferramenta/*framework*.

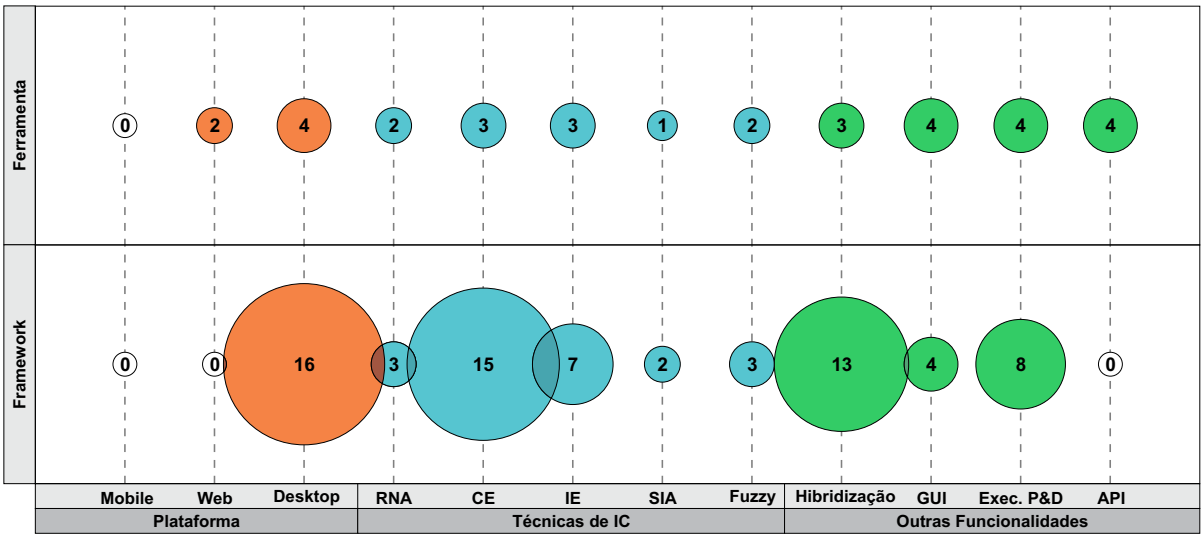


Figura 6 – Mapa de resultados da combinação das subquestões de pesquisa.

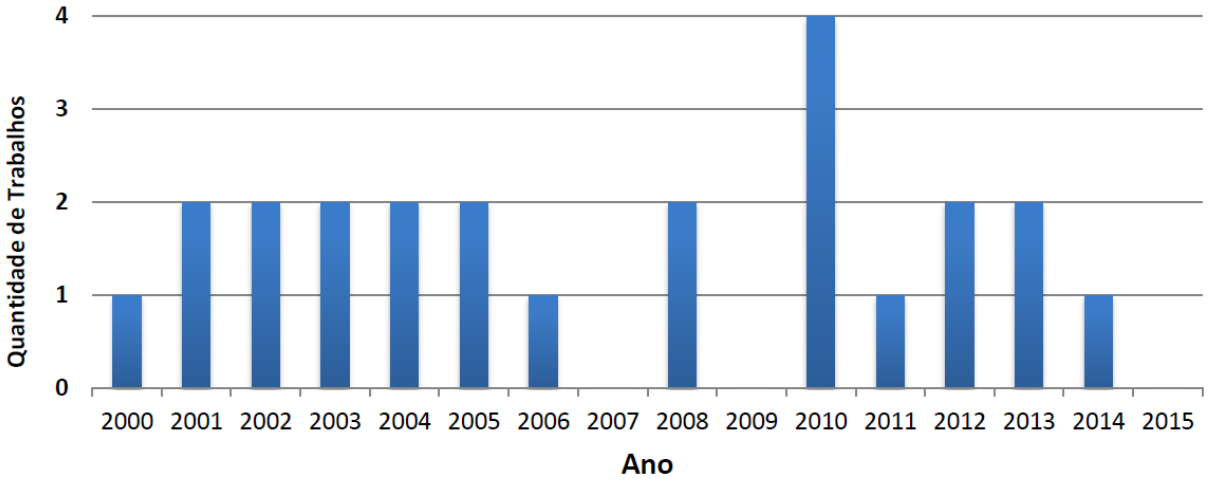


Figura 7 – Distribuição das publicações identificadas entre 2000 e 2015.

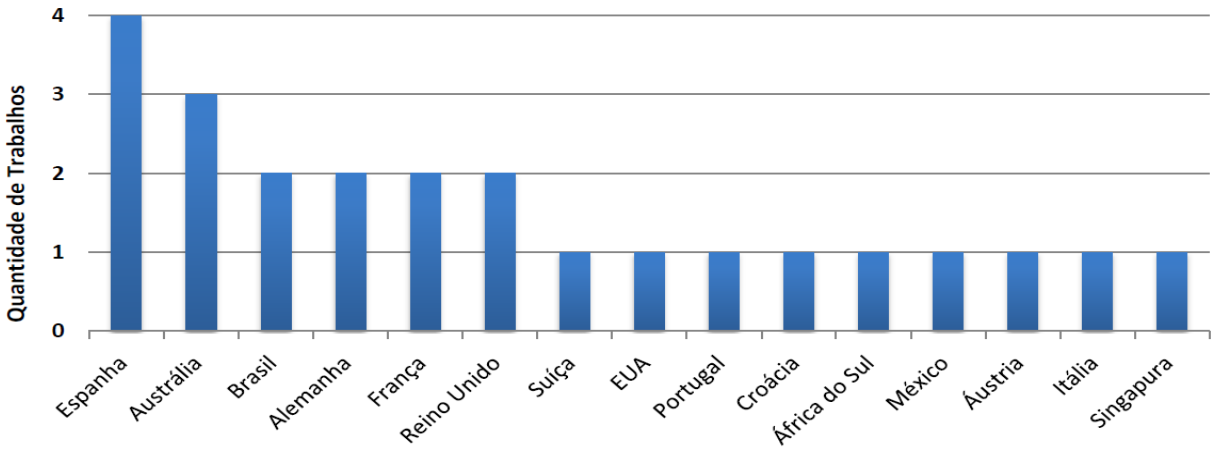


Figura 8 – Quantidade de trabalhos identificados por país.

As Figuras 7 e 8 apresentam os resultados de duas outras avaliações qualitativas realizadas com o objetivo de identificar a evolução das publicações no decorrer dos anos e os países com maior tradição nessa linha de pesquisa. A distribuição das publicações entre 2000 e 2015 manteve-se estável, pois, com exceção dos anos 2007, 2009 e 2015, nos demais sempre houve pelo menos uma publicação por ano. Dentre os países, os de maior destaque, considerando os trabalhos identificados nesse estudo, são: Espanha, Austrália, Brasil, Alemanha, França e Reino Unido. Quanto aos grupos de pesquisa, apenas a *University of Málaga* se destacou dos demais. Por fim, ressalta-se que a Tabela 3 traz informações que podem auxiliar na identificação de eventos de alto impacto para publicação de ferramentas que auxiliam o desenvolvimento de Sistemas Inteligentes, por exemplo, o IJCNN e o ICTAI, ambos eventos do IEEE (*Institute of Electrical and Electronics Engineers*).

2.1.7.1 Esquema de Classificação

Com o objetivo de classificar cada um dos Estudos Primários selecionados em conformidade com as características da pesquisa realizada, elaborou-se um esquema de classificação. O esquema consiste em dois eixos:

- **Tipo de Pesquisa**: esse eixo é utilizado para classificar os trabalhos entre os diferentes tipos de pesquisa, baseando-se nas características abstraídas a partir da metodologia da pesquisa. Os tipos foram adaptados de (WIERINGA et al., 2006) e todos os trabalhos selecionados foram classificados como *Solução Proposta* em virtude de proporem soluções para o desenvolvimento de Sistemas Baseados em Inteligência Computacional;

Tabela 5 – Esquema de classificação

	Categoria	Descrição
Tipo de Pequisa	Pesquisa de Avaliação	A metodologia é implementada na prática e uma avaliação é conduzida para avaliar as consequências da implementação em termos de vantagens e desvantagens.
	Solução Proposta	Uma solução para um problema é proposta. A solução pode ser nova ou uma extensão significativa de uma metodologia existente.
	Artigos Filosóficos	Esses estudos esboçam uma nova maneira de olhar as coisas existentes.
	Artigos de Opinião	Estes documentos expressam uma opinião pessoal de alguém sobre algo ou como as coisas deveriam ter sido feito.
	Artigos de Experiência	Explicam o que e como algo foi feito na prática. Deve ser a experiência pessoal do autor.
Tipo de Contribuição	Modelo	Representação de uma realidade observada.
	Teoria	Construção de relacionamentos causa-efeito de determinados resultados.
	Framework/Método	Modelos relacionados com a construção de *software* ou gerenciamento de processos de desenvolvimento.
	Guidelines	Lista de conselhos, síntese dos resultados da investigação obtidos.
	Lições aprendidas	Conjunto de resultados analisados por meio de um método de investigação.
	Ferramenta	Tecnologia, programa ou aplicação usada para criar, depurar, manter ou apoiar processos de desenvolvimento.
	Conselhos/Implicações	Recomendação discursiva considerada a partir de opiniões pessoais.

- **Tipo de Contribuição**: esse eixo, adaptado de (SHAW, 2003), descreve o tipo de contribuição proporcionada em cada Estudo Primário analisado. A Tabela 5 apresenta os tipos de pesquisa e contribuição, de acordo com os autores adotados neste Mapeamento Sistemático de Estudos.

2.1.7.2 Discussão da Questão de Pesquisa QP1

Nos trabalhos analisados foram identificadas 24 ferramentas que auxiliam profissionais e pesquisadores no design/implementação de aplicações baseadas em Inteligência Computacional. Cada trabalho foi analisado para responder as subquestões de pesquisa propostas. Assim, os resultados (Tabelas 3 e 4) fornecem uma visão geral das ferramentas encontradas, caracterizando-as em relação à(s):

- **Técnicas de IC**: para representar as subáreas da Inteligência Computacional apoiadas pela ferramenta. Os estudos primários podem ser classificados em cinco categorias definidas em (ENGELBRECHT, 2007). O percentual de publicações identificadas em cada subárea foram as seguintes: 5 trabalhos em Redes Neurais Artificiais (20% do total de publicações), 18 em Computação Evolutiva (75% do total de publicações), 10 em Inteligência de Enxames (41,6% do total de publicações), 3 em Sistemas Imunológicos Artificiais (12,5% do total de publicações) e 5 em Sistemas de Inferência *Fuzzy* (20% do total de publicações);

- **Plataforma**: para representar o tipo de plataforma na qual a ferramenta está disponível. Os trabalhos foram classificados em três categorias sendo elas: *Desktop*, *Web* ou *Mobile*. Foram identificadas apenas 2 ferramentas adaptadas à plataforma *Web* (8,3% do total de publicações), 20 ferramentas para ambientes *Desktop* (83,3% do total de publicações) e em 2 trabalhos a plataforma não foi identificada (8,3% do total de publicações);

- **Interface Gráfica**: para verificar se a ferramenta analisada disponibiliza uma interface gráfica de usuário (GUI, do inglês *Graphical User Interface*). Os trabalhos foram classificados em apenas duas categorias: Sim, caso apresente GUI, ou Não, caso não disponha de uma interface gráfica de usuário. Após a análise dos trabalhos aceitos, observou-se que 8 disponibilizam GUI (33% do total de publicações);

- **Hibridização**: para verificar se a ferramenta permite a construção de Sistemas Híbridos, ou seja, que combinam duas ou mais técnicas de IC para solucionar determinado problema. Os estudos podem ser classificados em: Sim, caso permita a hibridização, ou Não, caso contrário. Foram identificadas 16 ferramentas (66% do total de publicações) que possibilitam ao usuário a construção de Sistemas Inteligentes que combinem duas ou mais técnicas de Inteligência Computacional;

- **Computação paralela e/ou distribuída**: para verificar se a ferramenta permite a execução dos sistemas de forma paralela e/ou distribuída. Os estudos podem ser classificados em Sim, caso permita a computação paralela e distribuída, em Não, caso não permita computação paralela e distribuída. Observou-se que 12 trabalhos (50% do total de publicações) disponibilizam a funcionalidade de computação paralela e/ou distribuída;

- **API (*Application Programming Interface*)**: para verificar se a ferramenta disponibiliza uma API pública de modo a possibilitar diferentes formas de interação com o usuário ou com outros sistemas. Esse tipo de funcionalidade garante interoperabilidade ao sistema e não prende o usuário à interface gráfica. Os estudos podem ser classificados em Sim, caso disponibilize API, ou Não, caso contrário. Observou-se que apenas 4 trabalhos (16% do total de publicações) disponibilizam uma API para conexão com outros sistemas.

Figura 9 – Técnica de IC

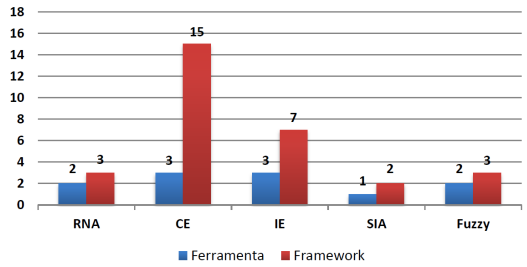


Figura 10 – Plataforma

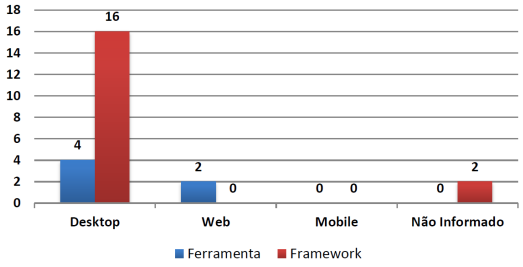


Figura 11 – Interface Gráfica

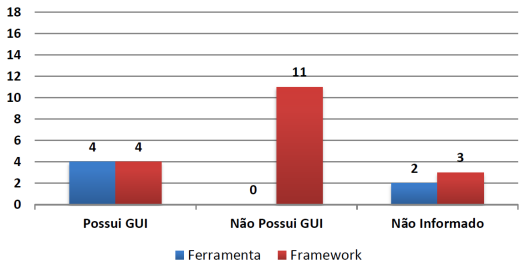


Figura 12 – Hibridização

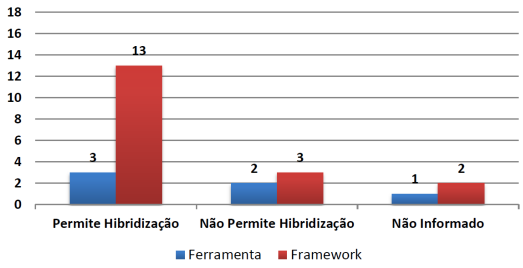


Figura 13 – Exec. Paral. e/ou Dist.

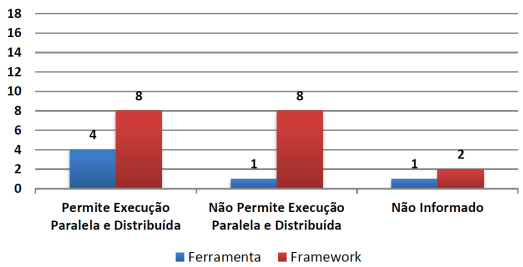


Figura 14 – API

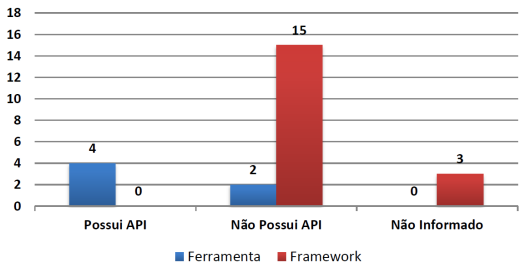


Figura 15 – Sumarização gráfica dos resultados do mapeamento.

A Figura 15 apresenta diversas combinações do mapa de resultados obtidos para as subquestões de pesquisa. Esses resultados indicam que há carência de pesquisas no desenvolvimento de:

- Ferramentas que auxiliem o desenvolvimento de sistemas de IC adaptadas às plataformas *Web* e *Mobile*. A maioria dos trabalhos identificados nesse estudo refere-se a ferramentas adaptadas apenas para ambientes *Desktop* (83,3%) e essa característica não favorece a alta disponibilidade e mobilidade das aplicações;
- Ferramentas que ofereçam suporte às técnicas baseadas em Sistemas Imunológicos Artificiais. Os algoritmos presentes nessa área são bastante utilizados para detecção de falhas de segurança, devido a alta capacidade de reconhecimento de padrões (CASTRO, 2006). Dessa forma, essa carência pode representar uma oportunidade de trabalhos futuros;
- API's públicas que auxiliem os profissionais e pesquisadores no desenvolvimento de novos produtos. Essa hipótese pode ser explicada devido a baixa quantidade de estudos encontrados referentes à ferramentas *Web*;
- Ferramentas que ofereçam interface gráfica de usuário que facilite o desenvolvimento de sistemas de Inteligência Computacional. Segundo FERNANDEZ; INSFRAN; ABRAHÃO (2011) e CHEN; YEN; HWANG (2012), a facilidade de uso pode definir o sucesso ou fracasso de uma aplicação, portanto, desenvolver ferramentas com GUI mostra-se promissor.

Poucos trabalhos forneceram informações sobre a forma de licenciamento da ferramenta proposta. Portanto, não foram obtidos resultados satisfatórios para classificação dos trabalhos quanto a esse quesito.

2.1.7.3 Discussões

Esta seção apresentou um MSE que teve por objetivo principal identificar ferramentas que apoiam pesquisadores e profissionais no desenvolvimento de aplicações inteligentes. Tais ferramentas foram classificadas em conformidade com as características extraídas de cada trabalho e, com isso, foi possível fornecer uma visão geral dessa linha de pesquisa, identificar lacunas e oportunidades de futura investigação, além de facilitar a escolha de uma ferramenta por parte de um profissional que necessita resolver determinado problema.

Foram aceitos 24 trabalhos como EP e observou-se que apenas 12,5% das ferramentas encontradas oferecem suporte ao desenvolvimento de aplicações baseadas em Sistemas Imunológicos Artificiais, 12% permitem hibridização de técnicas de IC e somente 16% disponibilizam uma API pública para conexão com outros sistemas. Além disso, a maioria dos trabalhos está restrita a plataforma *Desktop*.

Os resultados obtidos durante a análise dos estudos mostraram que a pesquisa no desenvolvimento de ferramentas adaptadas às plataformas *Web* e *Mobile* ainda tem sido pouco investigada e existe uma carência de ferramentas que apoiem pesquisadores e profissionais na subárea da IC referente à Sistemas Imunológicos Artificiais. Dessa forma, acredita-se que o desenvolvimento de ferramentas com tais características é um potencial ponto de investigação futura.

Finalmente, ressalta-se que o principal objetivo de um mapeamento é apresentar uma visão geral de determinada linha de pesquisa, no intuito de identificar lacunas e oportunidades de novos trabalhos. Dessa forma, diferente de uma revisão sistemática, um MSE tem caráter quantitativo e, em geral, não apresenta comparações qualitativas dos estudos selecionados (KEELE, 2007). Esse deficit é suprido pelas seções 2.3 e 2.4.

2.1.8 Ameaças à Validade

Existem muitos fatores que podem ameaçar a validade de um MSE, portanto identificou-se alguns desses fatores com o objetivo de mitigar suas influências nos resultados desse estudo.

A primeira ameaça refere-se à identificação dos EP, pois a utilização do mecanismo de busca automatizada não garante a inclusão de todos os trabalhos relevantes e ainda há o problema da indexação de trabalhos recentes. Essa ameaça foi mitigada com a realização de uma pesquisa piloto para verificar a consistência dos resultados obtidos nas bibliotecas digitais. Além disso, utilizou-se mais duas estratégias para evitar que trabalhos relevantes não fossem incluídos: i) análise das referências dos artigos selecionados; ii) inclusão de trabalhos considerados importantes, mas que foram publicados recentemente.

A formação da *string* de pesquisa também pode ser considerada como uma ameaça à validade deste trabalho. Há uma vasta quantidade de termos que poderiam ser incluídos para caracterizar ferramentas de IC, por exemplo, *Machine Learning*, *Data Mining*, *Fuzzy Systems*, *Artificial Neural Networks*, etc. Entretanto, ao realizar uma pesquisa piloto nas bibliotecas digitais, verificou-se que o resultado tornaria a pesquisa inviável devido a grande quantidade de trabalhos retornados. Dessa forma, decidiu-se por termos centrados em "*Computational Intelligence*". Essa decisão pode ter influenciado a não inclusão de ferramentas conhecidas (por exemplo, WEKA ou MatLab), entretanto foi possível mapear as ferramentas descritas usando o termo Inteligência Computacional.

Outro fator que pode ameaçar a validade dessa pesquisa diz respeito a seleção dos estudos e a extração dos dados. Essas etapas podem ser enviesadas pela opinião dos pesquisadores que estão executando o processo, portanto definiu-se um rígido protocolo para conduzir toda a pesquisa. Além disso, todos os trabalhos foram analisados separadamente por dois pesquisadores e as divergências foram solucionadas por outros dois pesquisadores.

2.2 Trabalhos Não Incluídos no Mapeamento

Apesar dos bons resultados obtidos pelo MSE, alguns trabalhos relevantes à área de IC não foram incluídos, pois não atendem aos critérios de inclusão definidos no mapeamento ou tratam-se de ferramentas comerciais, que não possuem artigos acadêmicos indexados pelas bases selecionadas. Esta seção discute algumas dessas ferramentas.

LEON; MIRANDA; SEGURA (2009) desenvolveram um *framework* para resolver problemas de otimização multiobjetivo que utiliza a arquitetura baseada em *plugins* para minimizar o esforço necessário ao incorporar novos problemas e algoritmos evolucionários. O *framework* é chamado METCO (*Metaheuristics-based Extensible Tool for Cooperative Optimization*) e fornece uma série de esqueletos algorítmicos. De acordo com os autores, esses esqueletos proporcionam uma vantagem importante em comparação com uma implementação direta a partir do zero, não só em termos de capacidade de reutilização, mas também no que se refere à metodologia e clareza de código (LEON; MIRANDA; SEGURA, 2009).

B-course (MYLLYMÄKI et al., 2002) é uma ferramenta *Web* gratuita que permite a análise de dados para verificação de dependências probabilísticas multivariadas por meio da utilização de redes Bayesianas. Essa ferramenta foi implementada como um *Application Service Provider*(ASP). Não há a necessidade de *download* ou instalação de *software*. B-Course pode ser utilizada na maioria dos navegadores *Web* e requer apenas que os dados de entrada sejam enviados em um arquivo de texto.

A *Waikato Environment for Knowledge Analysis* (HALL et al., 2009), mais conhecida como WEKA, tem como objetivo fornecer um conjunto abrangente de algoritmos de aprendizado de máquina e ferramentas de pré-processamento de dados para pesquisadores e profissionais. Para melhorar a experiência do usuário, o WEKA possui uma interface gráfica do usuário e permite que tal ambiente seja estendido de várias formas sem a necessidade de modificar seu núcleo. Essas funcionalidades tornam o WEKA um dos *software* mais bem conhecidos para solucionar problemas de aprendizado de máquina e mineração de dados.

RapidMiner é uma plataforma analítica moderna que acelera significativamente a produtividade - desde a preparação de dados até a descoberta de conhecimento - com modelos pré-construídos (MIERSWA et al., 2006). A RapidMiner é provavelmente uma das ferramentas mais semelhantes à Athena, porque fornece um editor de prototipagem para tarefas de descoberta de conhecimento. Como pode ser visto na Figura 16, a RapidMiner lida com os algoritmos como módulos com entradas, saídas e configurações. Essa abstração é muito semelhante à abordagem proposta neste trabalho, mas a Athena não é restrita à área de descoberta de conhecimento e a RapidMiner não é um serviço *Web*, ou seja, é necessário realizar o *download* e instalação da ferramenta em um computador local.

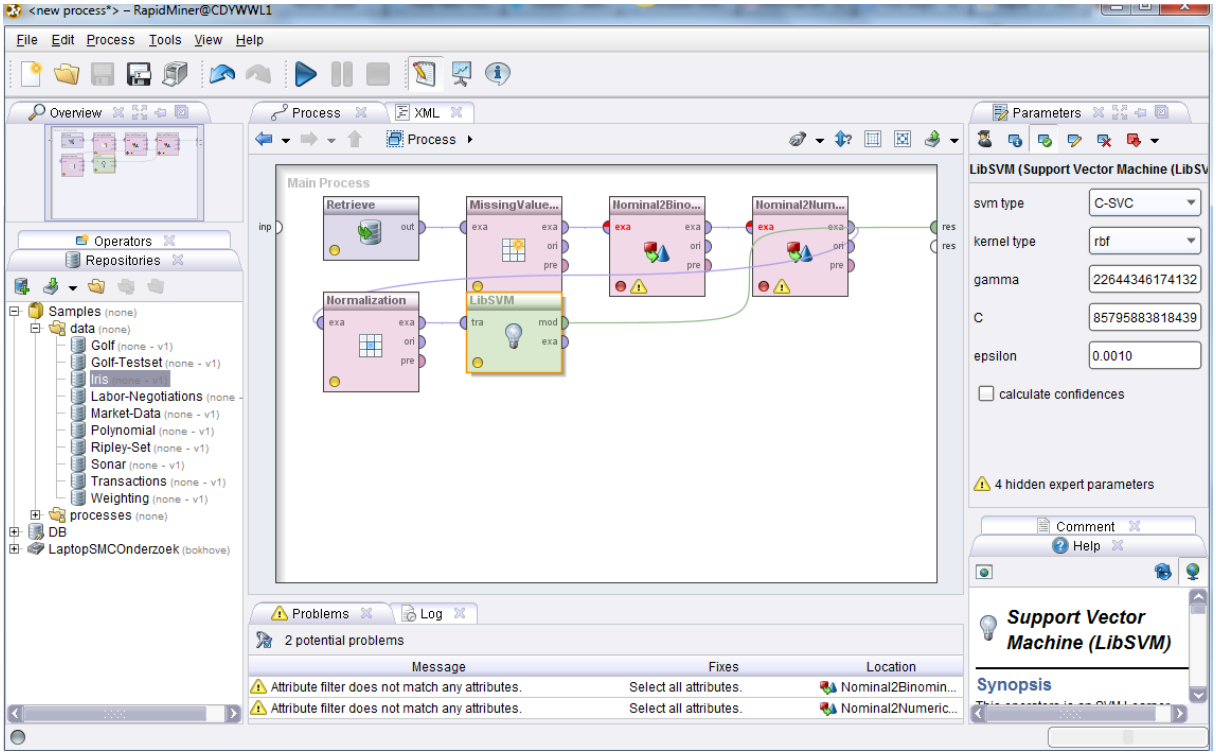


Figura 16 – Interface de usuário da ferramenta Rapidminer.

Outra ferramenta que merece um comentário especial é a *Google Prediction API*[1]. Ela oferece o poder dos algoritmos de Aprendizado de Máquina do Google por meio de uma API *RESTFul*. Como usuário, é possível fazer o *upload* dos dados de treinamento e, uma vez analisado, começar a classificar novas observações com base na análise dos dados de treinamento. Além dessa ferramenta, a Google também lançou recentemente outra aplicação *open source* para lidar com a AM, chamada TensorFlow[2]. A TensorFlow utiliza o modelo de grafos para realizar a computação de dados numéricos. Os nós do grafo representam operações matemáticas, enquanto que as arestas são canais de comunicação de dados multidimensionais. Essa ferramenta foi inicialmente desenvolvida por pesquisadores e engenheiros da equipe *Google Brain Team* para promover pesquisas na área de redes neurais profundas (*Deep Neural Networks*) e Aprendizado de Máquina em geral.

Finalmente, a ferramenta que está mais relacionada com a Athena é o *Azure Machine Learning Studio*[3], exceto pelo fato de que ela foi projetada para funcionar apenas com técnicas de aprendizagem de máquina. A *Azure Machine Learning Studio* foi lançada em 18 de fevereiro de 2015 na *Strata Big Data Conference*[4] e tem um visual de desenvolvimento colaborativo que permite ao usuário construir, testar e implantar soluções preditivas, tais como, sistemas de recomendação, análise de sentimento, detecção de fraudes e previsão de falhas, sem a necessidade de instalação pois funciona por meio de um navegador *Web*.

1 *Google Prediction API* - http://cloud.google.com/prediction/.
2 *TensorFlow* - http://www.tensorflow.org/.
3 *Azure ML Studio* - http://azure.microsoft.com/en-us/services/machine-learning/.
4 Strata website - http://strataconf.com/.

2.3 Principais Trabalhos

Este capítulo discutiu 24 trabalhos acadêmicos identificados pelo Mapeamento Sistemático e 7 ferramentas que não foram incluídas no mapeamento, mas que possuem reconhecida importância para a linha de pesquisa. Entretanto, apesar dos diversos trabalhos relacionados à abordagem proposta, alguns são destacáveis por possuírem métodos semelhantes ao proposto nesta dissertação. Dentre eles, *HeuristicLab*, CILib, Ellen, WEKA, RapidMiner e o *Azure Machine Learning Studio* merecem destaque.

A ferramenta *HeuristicLab* (WAGNER; AFFENZELLER, 2005) oferece um ambiente de otimização genérico, extensível e independente de paradigma. Ela foi concebida com um objetivo semelhante ao proposto neste trabalho que é o de permitir o reuso de implementações de algoritmos para resolver diversos tipos de problemas. Além disso, a ferramenta dispõe de uma interface gráfica para o usuário, o que facilita a configuração e execução dos algoritmos, bem como a obtenção e análise dos resultados. Outro diferencial é a diversidade de técnicas pré-embutidas. Atualmente, a *HeuristicLab* possui algoritmos de Aprendizado de Máquina, algoritmos baseados em população, algoritmos de busca baseados em trajetória e algoritmos multiobjetivo. Essa ferramenta oferece diversas vantagens para seus usuários, entretanto ainda está restrita ao ambiente *Desktop*. Diferente da Athena, para utilizar esse aplicativo o pesquisador deve fazer o *download* e instalar o programa em um computador pessoal. Outro ponto negativo refere-se a inexistência de um editor gráfico capaz de interconectar técnicas de diferentes áreas de forma simples e eficaz. Essa característica foi incorporada à proposta deste trabalho e é um diferencial das ferramentas WEKA, RapidMiner e *Azure Machine Learning Studio*.

A CILib (PAMPARA; ENGELBRECHT; CLOETE, 2008) é um projeto *open source* que tem por objetivo prover a construção de algoritmos de IC com o mínimo de esforço, utilizando componentes reutilizáveis. Essa ferramenta foi desenvolvida pelo grupo de pesquisa CIRG (*Computational Intelligence Research Group*) da Universidade de Pretória (Africa do Sul) e acabou inspirando o projeto dos componentes internos da Athena. Sua abordagem é semelhante à proposta deste trabalho, entretanto a CILib não dispõe de uma interface gráfica para a configuração dos sistemas computacionais. Esse processo é realizado utilizando arquivos no formato XML, o que dificulta sua utilização por usuário leigos em programação.

O trabalho de Ellen e Campbell (ELLEN; CAMPBELL, 2005) descreve um *framework* para desenvolver algoritmos IC em sistemas distribuídos. A arquitetura dos componentes foi definida com base no padrão de objetos com portas (STEWART; VOLPE; KHOSLA, 1997). Este modelo permite um baixo acoplamento entre os módulos e torna o sistema mais flexível. Essa ideia também foi utilizada na Athena, no entanto, ao invés de controles de entrada/saída, a Athena utiliza módulos com configurações ajustáveis em tempo de execução.

Os três trabalhos que estão mais próximos da proposta defendida pela Athena são as ferramentas WEKA (HALL et al., 2009), RapidMiner (MIERSWA et al., 2006) e *Azure Machine Learning Studio*. Essas ferramentas possuem como características em comum a adequação a problemas que envolvem a Aprendizagem de Máquina e a existência de editores gráficos que facilitam a construção dos sistemas. Essas aplicações já estão consolidadas na área de IC e apresentam uma quantidade considerável de usuários ativos. Entretanto, elas não foram projetadas para funcionar com técnicas de outras subáreas da IC, por exemplo, meta-heurísticas e algoritmos de busca. A WEKA e a RapidMiner não foram mapeadas porque o termo *Machine Learning* não foi incluído na busca e não foram encontrados estudos científicos que abordassem a *Azure Machine Learning Studio*.

2.4 Avaliação Comparativa

As ferramentas supracitadas apresentam diversas características e cada uma possui sua particularidade. Dessa forma, a apresentação de uma comparação objetiva entre esses trabalhos, mostra-se como uma informação relevante a pesquisadores que desejam entender um pouco mais sobre essa linha de pesquisa. Entretanto, a tarefa de definir critérios eficazes para avaliar tais ferramentas é complexa. Portanto, decidiu-se utilizar uma adaptação do método proposto por PAREJO et al. (2012) para qualificar essas ferramentas.

2.4.1 Método de Avaliação

Avaliar um *software* geralmente implica no equilíbrio de diversos interesses, muitas vezes conflitantes, em relação à nova tecnologia (PAREJO et al., 2012). Nesse sentido, os critérios comparativos propostos por PAREJO et al. (2012) abrangem seis áreas de interesse, que, por sua vez, são subdivididas em 35 características. Neste trabalho, decidiu-se pela simplificação desse método visando sua adequação aos conceitos propostos por ENGELBRECHT (2007).

A Tabela 6 apresenta as áreas de interesse, características e funcionalidade utilizadas na avaliação dos trabalhos relacionados. As características pertencentes às áreas A1 e A2 representam requisitos funcionais (PRESSMAN, 2011) e, para cada característica, foi identificada uma série de funcionalidades relevantes para ferramentas que trabalham com Inteligência Artificial (nesse caso, utilizamos o termo IA porque alguns trabalhos relacionados não estão restritos à área da IC). Essas funcionalidades são avaliadas de modo objetivo utilizando um valor binário (0 ou 1).

Esse método de avaliação busca obter, de forma objetiva, conhecimento sobre a real capacidade das ferramentas identificadas anteriormente. Dessa forma, a nota final de cada ferramenta é calculada de acordo com as Equações 2.1 e 2.2, sujeitas às restrições apresentadas na Equação 2.3.

Tabela 6 – Critérios para avaliação dos trabalhos relacionados (requisitos funcionais).

Área de Interesse	**Características**	**Funcionalidade**
A1. Técnicas de Inteligência Artificial	A1.1 Redes Neurais Artificiais	Perceptron/Adaline
		Perceptron Multicamadas
		Redes de funções de base radial
		Redes auto-organizáveis
	A1.2 Computação Evolutiva	Algoritmo genético
		Estratégia evolutiva
		Programação genética
		Programação evolutiva
		Evolução diferencial
		Evolução cultural
		Coevolução
	A1.3 Inteligência de Enxames	PSO
		PSO discreto
		Ant System
		Ant Colony System
		Max-Min Ant System
		Ant System Rank
	A1.4 Sistemas Imunológicos Artificiais	CLONALG
		optIA
		aiNet
	A1.5 Sistemas de Inferência Fuzzy	Sistemas *Fuzzy* baseados em regras
		Fuzzy Tipo-2
		Fuzzy C-Means
		Algoritmos *Neuro-Fuzzy*
	A1.6 Algoritmos de Busca	*Steepest descent/hill climbing*
		Simulated annealing
		Tabu search
		Scatter search
		GRASP
		Variable neighborhood search (VNS)
	A1.7 Algoritmos de Aprendizado de Máquina	*Apriori*
		Tertius
		HotSpot
		Redes de Bayes
		C4.5
		KStar
		NaiveBayes
		RandomForest
		RandomTree
		Cobweb
		K-Means
	A1.8 Meta-heurísticas Multiobjetivo	PGA
		MOGA
		NSGA
		NSGA-II
		NPGA
		SPEA
		SPEA-II
		PAES
		PESA
		PESA-II
		MOMGA
		ARMOGA
		Multobjective Ant System
		Multiobjective PSO
		POSA
		MOSA
A2. Suporte ao Processo de Otimização	A2.1 Adaptação ao problema	Diferentes classes de problemas
		Pré-processamento dos dados de entrada
		Codificação da solução
		Definição da estrutura de vizinhança
		Construção de heurísticas baseadas em população
		Estratégias para seleção de soluções
		Especificação de funções objetivo
		Definição e controle de restrições
	A2.2 Características Avançadas	Hibridização
		Hiper-heurísticas
		Execução paralela e distribuída
	A2.3 Condições de Parada	Número máximo de iterações
		Valor específico obtido pela função objetivo
		Tempo de execução
		Combinação logica entre condições de parada
	A2.4 Execuções em Massa (*Batch execution*)	Repetição automática de tarefas simples
		Automatização de tarefas variando os parâmetros de execução
		Repetição automática de diferentes tarefas
		Automatização de diferentes tarefas em diferentes problemas
	A2.5 Projeto de Experimentos	Definição das Hipóteses
		Modelagem do experimento (especificação das variáveis)
		Desenho do experimento (Fatorial, quadrado latino, etc.)
		Auxílio na execução do experimento
		Geração do plano de execução
		Importar/Exportar experimentos
	A2.6 Análises Estatísticas	T-Student
		ANOVA
		Wilcoxon
		Mann-Whitnet
		Kolmogorov-Smirnov
		Importar/Exportar análises estatísticas
	A2.7 GUI e Relatórios Gráficos	Design
		Configuração das técnicas de IA
		Modelagem do problema
		Auxílio no planejamento de tarefas de otimização
		Suporte a definição de diferentes projetos
		Representação gráfica dos resultados
	A2.8 Interoperabilidade	Importar dados
		Exportar dados
		Web Service
		Manipulação de projetos por meio de XML ou JSON

$$Nota = \sum_{i=1}^{m} A_i \tag{2.1}$$

$$A_i = \sum_{j=1}^{n} \left(\sum_{k=1}^{o} w_{jk} \times x_{jk} \right) \times \lambda_j \tag{2.2}$$

$$Restrições : \begin{cases} \sum_{k=1}^{o} w_{jk} & = & 1 \\ \sum_{j=1}^{n} \lambda_j & = & 1 \\ x_{jk} & \in & \{0, 1\} \\ i & \in & \{1, 2\} \end{cases} \tag{2.3}$$

A Equação 2.1 computa a nota final para cada ferramenta por meio do somatório dos valores obtidos em cada área de interesse A_i. O valor obtido nas áreas é calculado utilizando a Equação 2.2, na qual o índice j representa a área e o k simboliza a funcionalidade. A variável x_{jk} guarda a informação que define se a funcionalidade está presente na ferramenta em análise (0, caso não contenha a funcionalidade, ou 1, caso contrário) e a λ_j representa o peso de cada característica em relação à área de interesse. Esses pesos w_{ij} e λ_i foram mantidos de acordo com a proposta de PAREJO et al.. Por fim, as Restrições 2.3 definem que o somatório dos pesos atribuídos às funcionalidades de determinada característica e o somatório dos pesos das características de uma área devem ser igual a 1.

É importante ressaltar que a definição dos pesos expressa a relevância de cada funcionalidade para o contexto no qual a avaliação é realizada. Desse modo, em outros cenários - industrial, por exemplo - os pesos podem variar para refletir a exata importância das funcionalidades, características e áreas de interesse.

2.4.2 Resultados e Discussões

A Tabela 7 apresenta a compilação dos resultados para as áreas de interesse A1 e A2. Os trabalhos estão ordenados de acordo com a nota final obtida pelas ferramentas e para cada característica o maior valor foi destacado com negrito. O Apêndice B exibe os resultados de forma detalhada.

A principal funcionalidade de qualquer ferramenta que auxilia o desenvolvimento de Sistemas Inteligentes é o conjunto de técnicas suportadas. Nessa área (A1), avaliou-se a existência de algoritmos pertencentes a oito diferentes subáreas da IA.

A1.1 Redes Neurais Artificiais (RNA): esse campo da IA abriga as técnicas inspiradas na estrutura e no funcionamento do sistema nervoso dos seres vivos. Tais técnicas são aplicadas para resolver problemas de classificação, identificação de padrões, otimização, controle de processo, dentre outros. Nesse quesito a ferramenta que obteve maior nota foi a Athena.

Tabela 7 – Resultado final para as áreas de interesse A1 e A2.

	Técnicas de IA								**Suporte ao processo de otimização**								
	A1.1 RNA	A1.2 CE	A1.3 IE	A1.4 SIA	A1.5 SIF	A1.6 Alg. de Busca	A1.7 Alg. de AM	A1.8 MH Multi.	A2.1 Adapt. ao Problema	A2.2 Carac. avançadas	A2.3 Condições de parada	A2.4 Exec. em massa	A2.5 Experimentos	A2.6 Análises estatísticas	A2.7 GUI e relatórios	A2.8 Interoperabilidade	Nota Final
Athena	**0,125**	0,036	0,031	**0,042**	**0,083**	0,000	0,090	0,016	0,078	0,084	0,063	**0,063**	0,000	0,000	0,104	**0,094**	**0,908**
HeuristicLab	0,031	0,054	0,047	0,000	0,000	**0,104**	0,028	0,008	0,094	0,042	0,063	**0,063**	0,021	0,000	**0,125**	**0,094**	0,772
EvA2	0,000	**0,072**	0,031	0,000	0,000	0,063	0,000	**0,054**	0,078	0,042	**0,125**	0,031	0,000	0,000	0,104	0,063	0,663
CILib	0,031	0,036	**0,078**	**0,042**	0,063	0,000	0,021	0,000	0,094	0,042	**0,125**	**0,063**	0,000	0,000	0,021	0,031	0,646
FOM	0,000	0,018	0,000	0,000	0,000	**0,104**	0,000	0,000	0,063	0,084	**0,125**	0,031	**0,104**	**0,038**	0,063	0,000	0,629
ParadisEO	0,000	0,054	0,063	0,000	0,000	0,084	0,000	0,023	0,094	0,084	**0,125**	0,000	0,000	0,000	0,000	0,063	0,588
WEKA	0,094	0,000	0,000	0,000	0,000	0,000	**0,125**	0,000	0,031	0,042	0,021	**0,063**	0,000	0,000	0,104	0,063	0,542
Azure ML	0,063	0,000	0,000	0,000	0,000	0,000	0,076	0,000	0,031	0,084	0,000	**0,063**	0,000	0,000	0,104	0,063	0,483
RapidMiner	0,063	0,000	0,031	0,000	0,000	0,000	0,062	0,000	0,031	0,042	0,021	**0,063**	0,000	0,000	0,104	0,063	0,479
AFRANCI	0,063	0,000	0,000	0,000	0,000	0,000	0,007	0,000	0,031	0,084	0,021	0,000	0,000	0,000	0,084	0,063	0,351
METCO	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,031	**0,109**	**0,125**	0,042	0,000	0,000	0,000	0,042	0,000	0,349
MALLBA	0,000	0,036	0,031	0,000	0,000	0,042	0,000	0,008	0,063	0,084	0,063	0,000	0,000	0,000	0,021	0,000	0,346
TensorFlow	0,094	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,031	0,084	0,021	0,000	0,000	0,000	0,000	**0,094**	0,323
jMetal	0,000	0,054	0,063	0,000	0,000	0,021	0,000	0,039	0,094	0,000	0,042	0,000	0,000	0,000	0,000	0,000	0,311
PMF	0,000	0,018	0,000	**0,042**	0,000	0,042	0,000	0,000	0,094	0,042	0,063	0,000	0,000	0,000	0,000	0,000	0,300
HSF	0,000	0,018	0,000	0,000	0,000	0,063	0,000	0,000	**0,109**	0,042	0,063	0,000	0,000	0,000	0,000	0,000	0,294
Rosemberg	0,000	0,018	0,000	0,000	0,000	0,042	0,000	0,000	**0,109**	0,084	0,042	0,000	0,000	0,000	0,000	0,000	0,294
Zha	0,063	0,000	0,000	0,000	**0,083**	0,000	0,000	0,000	0,016	0,042	0,021	0,000	0,000	0,000	0,063	0,000	0,287
jMetalCpp	0,000	0,054	0,031	0,000	0,000	0,000	0,000	0,023	0,094	0,042	0,042	0,000	0,000	0,000	0,000	0,000	0,285
HOTFRAME	0,000	0,018	0,000	0,000	0,000	0,042	0,000	0,000	**0,109**	0,000	0,063	0,000	0,000	0,000	0,042	0,000	0,273
iOpt	0,000	0,018	0,000	0,000	0,000	0,042	0,000	0,000	**0,109**	0,042	0,021	0,000	0,000	0,000	0,042	0,000	0,273
I-om	0,000	0,018	0,000	0,000	0,000	0,042	0,000	0,000	0,094	0,042	0,042	0,000	0,000	0,000	0,000	0,031	0,268
Ellen	0,031	0,000	0,000	0,000	**0,083**	0,000	0,000	0,000	0,016	0,084	0,000	0,000	0,000	0,000	0,021	0,000	0,235
MAGMA	0,000	0,000	0,016	0,000	0,000	0,021	0,000	0,000	0,078	0,042	0,042	0,000	0,000	0,000	0,021	0,000	0,219
IT2FLS	0,031	0,000	0,000	0,000	0,021	0,000	0,000	0,000	0,031	0,000	0,000	0,000	0,000	0,000	0,063	0,031	0,177
Nyx	0,000	0,018	0,000	0,000	0,000	0,000	0,000	0,000	0,094	0,000	0,063	0,000	0,000	0,000	0,000	0,000	0,174
Prediction API	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,031	0,042	0,000	0,000	0,000	0,000	0,000	**0,094**	0,167
Weyland	0,000	0,018	0,047	0,000	0,000	0,042	0,000	0,000	0,016	0,042	0,000	0,000	0,000	0,000	0,000	0,000	0,164
MAFRA	0,000	0,018	0,000	0,000	0,000	0,000	0,000	0,000	0,094	0,000	0,042	0,000	0,000	0,000	0,000	0,000	0,153
B-Course	0,000	0,000	0,000	0,000	0,000	0,000	0,007	0,000	0,016	0,000	0,000	0,000	0,000	0,000	0,063	0,031	0,116
Optimus	0,000	0,000	0,031	0,000	0,000	0,000	0,000	0,000	0,016	0,042	0,021	0,000	0,000	0,000	0,000	0,000	0,110

A1.2 Computação Evolutiva (CE): as técnicas pertencentes a essa subárea são inspiradas nos princípios da evolução biológica. Elas são adequadas para resolver problemas cujo espaço de busca é descontinuo, não-diferenciável e multimodal. A ferramenta que apresentou melhor nota para essa característica foi a EvA2.

A1.3 Inteligência de Enxames (IE): essa subárea originou-se a partir do estudo de colônias e enxames de organismos sociais, por exemplo, abelhas e formigas. Esses algoritmos são utilizados para resolver problemas de otimização e agrupamento de dados. Nesse critério a CILib obteve o melhor desempenho.

A1.4 Sistemas Imunológicos Artificiais (SIA): inspiradas no funcionamento do sistema imunológico natural, as técnicas dessa linha de pesquisa são muito aplicadas em problemas que exigem uma alta capacidade no reconhecimento de padrões, por exemplo, detecção de fraudes e identificação de vírus de computador. Nesse quesito apenas 9,6% das ferramentas pontuaram: Athena, CILib e PMF.

A1.5 Sistemas de Inferência Fuzzy (SIF): a principal característica das técnicas pertencentes a essa subárea refere-se ao raciocínio aproximado, inspirado na forma como o ser humano lida com informações imprecisas. Nesse quesito foram consideradas 4 funcionalidades: SIF tradicionais, *Fuzzy* Tipo-2, *Fuzzy* C-Means e algoritmos *Neuro-Fuzzy*. Três ferramentas empataram nessa característica: Athena, Zha e Ellen.

A1.6 Algoritmos de Busca: os algoritmos de busca tradicionais são projetados para explorar espaços de busca com o objetivo de encontrar o melhor ponto de acordo com uma função objetivo. Para avaliar essa característica considerou-se os algoritmos Subida da Encosta (*Hill Climbing*), Têmpera Simulada (*Simulated Annealing*), Busca Tabu (*Tabu Search*), Busca Dispersa (*Scatter Search*), GRASP e Busca em Vizinhança Variável (*Variable Neighborhood Search*). As ferramentas que obtiveram as melhores representações nesse critério foram a HeuristicLab e FOM.

A1.7 Algoritmos de Aprendizado de Máquina: a aprendizagem refere-se à capacidade de melhorar o desempenho nas tarefas futuras após a realização de observações sobre o problema. Para avaliar essa característica analisou a presença de algoritmos de regra de associação, aprendizagem probabilística, árvores de decisão e algoritmos de agrupamento. Nesse quesito a WEKA apresentou o melhor resultado.

A1.8 Meta-heurísticas Multiobjetivo: essas técnicas são utilizadas para resolver problemas com dois ou mais objetivos. As funcionalidades consideradas para avaliar essa característica foram definidas de acordo com PAREJO et al. (2012). A ferramenta que obteve a melhor representação nesse critério foi a EvA2.

A área de interesse A2 avalia o suporte que cada ferramenta oferece ao processo de otimização. Nessa área, foram avaliadas oito características que vão desde a capacidade de adaptação ao problema até questões relacionadas a interface gráfica do usuário e a

interoperabilidade das ferramentas.

A2.1 Adaptação ao Problema e A2.2 Características Avançadas: esses dois critérios avaliam a capacidade de adaptação ao problema, além da existência de funções avançadas como a hibridização, suporte a hiper-heurísticas e possibilidade de execução paralela ou distribuída. Quanto à adaptação ao problema, 100% das ferramentas apresentam funções relacionadas a essa característica e cinco ferramentas empataram com a melhor avaliação: HSF, Rosemberg, iOpt, HOTFRAME e METCO. Quanto às características avançadas, a METCO obteve a melhor nota e apenas 6 trabalhos não pontuaram.

A2.3 Condições de Parada e A2.4 Execuções em Massa: alguns algoritmos de IA não possuem critérios bem definidos quanto às condições de parada. Dessa forma, é desejável que as ferramentas permitam a especificação dessas condições, bem como a execução programada de determinadas tarefas. Considerando esses critérios, as ferramentas que se destacam são a CILib, FOM, EvA2, ParadisEO, Athena, HeuristicLab, WEKA, RapidMiner e Azure ML.

A2.5 Projeto de Experimentos e A2.6 Análises Estatísticas: estudos experimentais bem projetados são essenciais para a obtenção de resultados confiáveis, entretanto 93% das ferramentas não oferecem suporte ao projeto de experimentos e apenas uma ferramenta dispõe de mecanismos que auxiliam a realização de análises estatísticas. FOM e HeuristicLab são as duas ferramentas que apresentaram funções relacionadas a essas características.

A2.7 GUI e Relatórios Gráficos: a usabilidade de uma aplicação depende diretamente da forma como a interface gráfica do usuário foi projetada. Essa característica é fundamental para usuários com pouco ou nenhum conhecimento em programação, pois facilita a configuração e execução dos algoritmos, bem como a interpretação dos resultados por meio dos relatórios gráficos. Nesse quesito, 58% das ferramentas apresentam GUI e a maior nota foi obtida pela ferramenta HeuristicLab.

A2.8 Interoperabilidade: essa característica define a capacidade que um sistema tem de trocar informações com outros de forma transparente. Isso permite a execução de técnicas de IA de forma independente do cliente que solicitou o serviço, além da importação e exportação de dados/resultados. Nesse quesito, 14 das 31 ferramentas analisadas possuem funcionalidades relacionadas à interoperabilidade. Dentre essas 14, quatro merecem destaque: HeuristicLab, Athena, TensorFlow e *Google Prediction API*.

A Figura 17 apresenta a relação entre a quantidade de ferramentas que não apresentam nenhuma funcionalidade relacionada a determinada característica (em vermelho e hachurado) e a quantidade de ferramentas que dispõem de pelo menos uma funcionalidade em relação às características analisadas (em azul). Com isso, pode-se perceber que as características A2.1, A2.2 e A2.3 estão presentes na maioria dos trabalhos e há poucas ferramentas com as características A2.5 e A2.6.

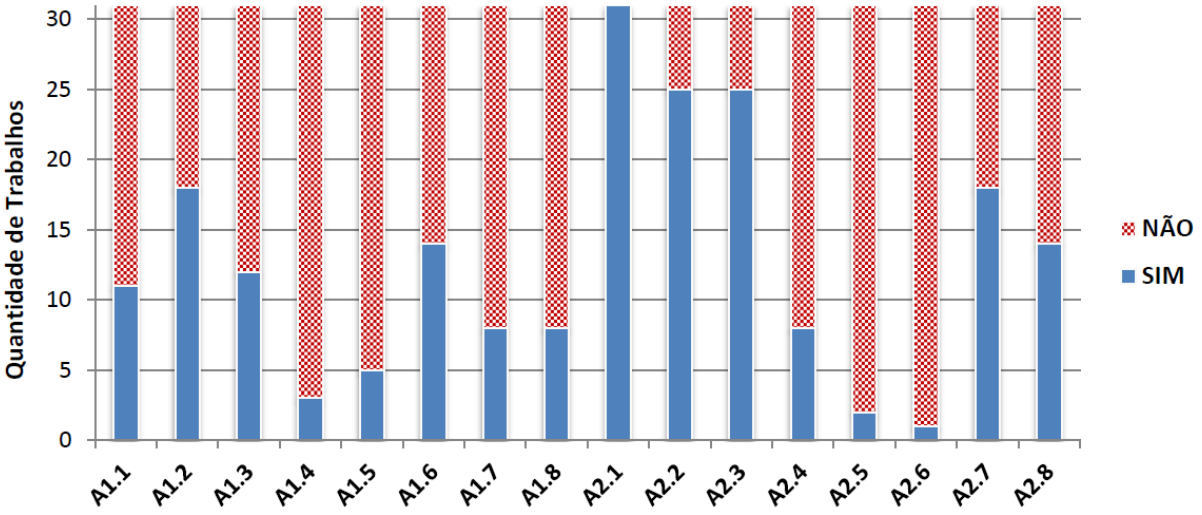


Figura 17 – Relação entre a quantidade de ferramentas que apresentam ou não determinada característica.

Por fim, ressalta-se que apesar da Athena ter obtido a melhor nota final, não há uma ferramenta dominante em todas as características e áreas de interesse. Esse fato já foi discutido pelo teorema *No Free Lunch Theorems for Optimization* (NFL) (WOLPERT; MACREADY, 1997) e reforça a justificativa deste trabalho, uma vez que ainda há espaço para novas investigações dentro dessa linha de pesquisa.

2.5 Considerações Finais

Este capítulo apresentou o amplo processo de identificação, triagem, sumarização e discussão das principais ferramentas desenvolvidas com o objetivo de auxiliar o desenvolvimento de Sistemas Baseados em Inteligência Computacional. Esse estudo fornece uma visão geral dessa linha de pesquisa, agrupando os resultados e identificando lacunas passíveis de investigação. Isso atende ao segundo objetivo desta dissertação (ver Seção I).

Parte III

Proposta

3 Arquitetura da Solução

O termo arquitetura de *software* possui diversos conceitos (CLEMENTS et al., 2002). Para a Engenharia de *Software*, a definição de uma arquitetura envolve a descrição de elementos arquiteturais (componentes) dos quais os sistemas serão constituídos, interações entre esses elementos, padrões que guiam suas composições e restrições sobre esses padrões (PRESSMAN; JAWADEKAR, 1987; GARLAN, 2000). Em geral, a definição da arquitetura de uma solução acontece após o levantamento dos requisitos e é fundamental para que um projeto tenha êxito, pois facilita a comunicação com as partes interessadas (*stakeholders*), permite o gerenciamento dos riscos e a diminuição de custos, além de ser a base para um desenvolvimento seguro e com qualidade (GAMMA et al., 1994).

Este capítulo discute a arquitetura proposta para auxiliar o desenvolvimento de Sistemas Baseados em Inteligência Computacional (SBIC). São apresentados as motivações, os requisitos, as estruturas fundamentais, as definições de comportamento e as tecnologias adotadas. Para facilitar o entendimento, dividiu-se o capítulo em duas grandes seções: Modelo Conceitual e Modelo Tecnológico.

3.1 Modelo Conceitual

Esta seção apresenta os conceitos relacionadas à arquitetura sem considerar quais tecnologias devem ser adotadas para cada componente. Dessa forma, o modelo conceitual da Arquitetura Proposta (AP) não está vinculado diretamente a nenhuma tecnologia específica, o que permite que outros pesquisadores utilizem-na como base para trabalhos futuros. Esse modelo atende ao terceiro objetivo específico deste trabalho (ver Seção I - *Objetivos*).

3.1.1 Princípios Arquiteturais

Atualmente, quando um pesquisador ou profissional deseja empregar uma técnica de Inteligência Computacional para resolver determinado problema, destacam-se quatro tarefas importantes: i) modelar o problema-alvo; ii) desenvolver uma aplicação utilizando ferramentas especializadas; iii) realizar experimentos para avaliar os resultados obtidos; e iv) integrar a aplicação com outros sistemas.

A primeira tarefa requer conhecimentos sobre o problema-alvo e sobre as técnicas escolhidas para solucioná-lo. O problema deve ser modelado de tal forma que a técnica selecionada seja capaz de encontrar uma solução adequada. A segunda tarefa exige conhecimento técnico na ferramenta utilizada para apoiar o desenvolvimento. Atualmente,

a maioria dessas ferramentas disponibiliza apenas o esqueleto dos algoritmos, o que implica na necessidade do conhecimento em linguagens de programação para viabilizar a construção da aplicação. A terceira tarefa é importante para ajustar os parâmetros das técnicas, possibilitando a obtenção de melhores resultados. Por fim, a quarta tarefa é responsável por integrar o que foi desenvolvido com outros sistemas, objetivando a exploração máxima das vantagens da aplicação do Sistema Inteligente em ambientes reais.

A realização dessas tarefas envolve a superação de uma série de dificuldades, como o alto custo de desenvolvimento, a difícil reutilização das implementações, ferramentas inadequadas, dentre outras (ver Seção I - *Definição do Problema*). Considerando esse contexto, foi definida uma arquitetura capaz de facilitar a construção, manutenção e aplicação de Sistemas Baseados em Inteligência Computacional. Essa definição foi guiada por um conjunto de princípios levantados a partir da análise cuidadosa das dificuldades enfrentadas no desenvolvimento de SBIC e das lacunas presentes nessa linha de pesquisa. A seguir são apresentados tais princípios:

- **Simplicidade**: um dos maiores obstáculos a respeito da aplicação de algoritmos IC é a complexidade por trás da modelagem e codificação das soluções. Dessa forma, a arquitetura foi definida com o intuito de simplificar o desenvolvimento de SBIC. Decidiu-se por uma arquitetura modular para que fosse possível encapsular a complexidade interna das técnicas, deixando exposto apenas o que interessa ao usuário final: entradas, configurações e saída. Além disso, foi realizada uma adaptação da abstração de circuitos lógicos digitais para permitir a interconexão dos módulos. Com isso, o conhecimento técnico deixa de ser primordial na construção dos SBIC;

- **Extensibilidade**: essa característica é fundamental para facilitar a integração de novos módulos. A padronização das técnicas de IC aliada à incorporação do padrão modular permite que essa integração aconteça em tempo real (*on-the-fly*), ou seja, os usuários podem incluir novos módulos sem a necessidade de reiniciar a aplicação. Entretanto, essa característica acabou gerando um problema de segurança que é discutido no Apêndice D;

- **Software como Serviço**: em geral, a maioria das ferramentas disponíveis requer dos usuários um processo de instalação, o que pode acabar dificultando sua utilização. A arquitetura proposta foi construída utilizando o modelo *Software* como Serviço (SaaS), no qual o serviço é distribuído por meio da Internet. A adoção desse modelo conferiu à arquitetura características como serviço sob demanda, alta disponibilidade e mobilidade (acesso aos dados de qualquer computador com a acesso à Internet);

- **Alta Performance**: dentre as quatro atividades mencionadas anteriormente, uma das mais importantes é a realização dos experimentos para verificar a adequabilidade da solução proposta, além da realização do ajuste fino dos parâmetros para obter

melhores resultados. Essa tarefa pode requerer uma grande quantidade de recursos (memória e processamento) e, dependendo da situação, sua realização é inviável em infraestrutura local (pouco robusta). A fim de atenuar esses problemas, esse princípio prescreve que o usuário tenha a possibilidade de ajustar o recurso computacional de acordo com a sua necessidade (elasticidade) e que a implementação das técnicas considere questões relacionadas à eficiência na utilização dos recursos computacionais;

- **Colaboração**: a AP também foi desenvolvida visando a colaboração entre pesquisadores da comunidade de IC. Os usuários podem compartilhar seus resultados, discutir sobre a solução proposta para determinado problema e propor melhorias para os módulos, facilitando a troca de conhecimento entre grupos de todo o mundo.

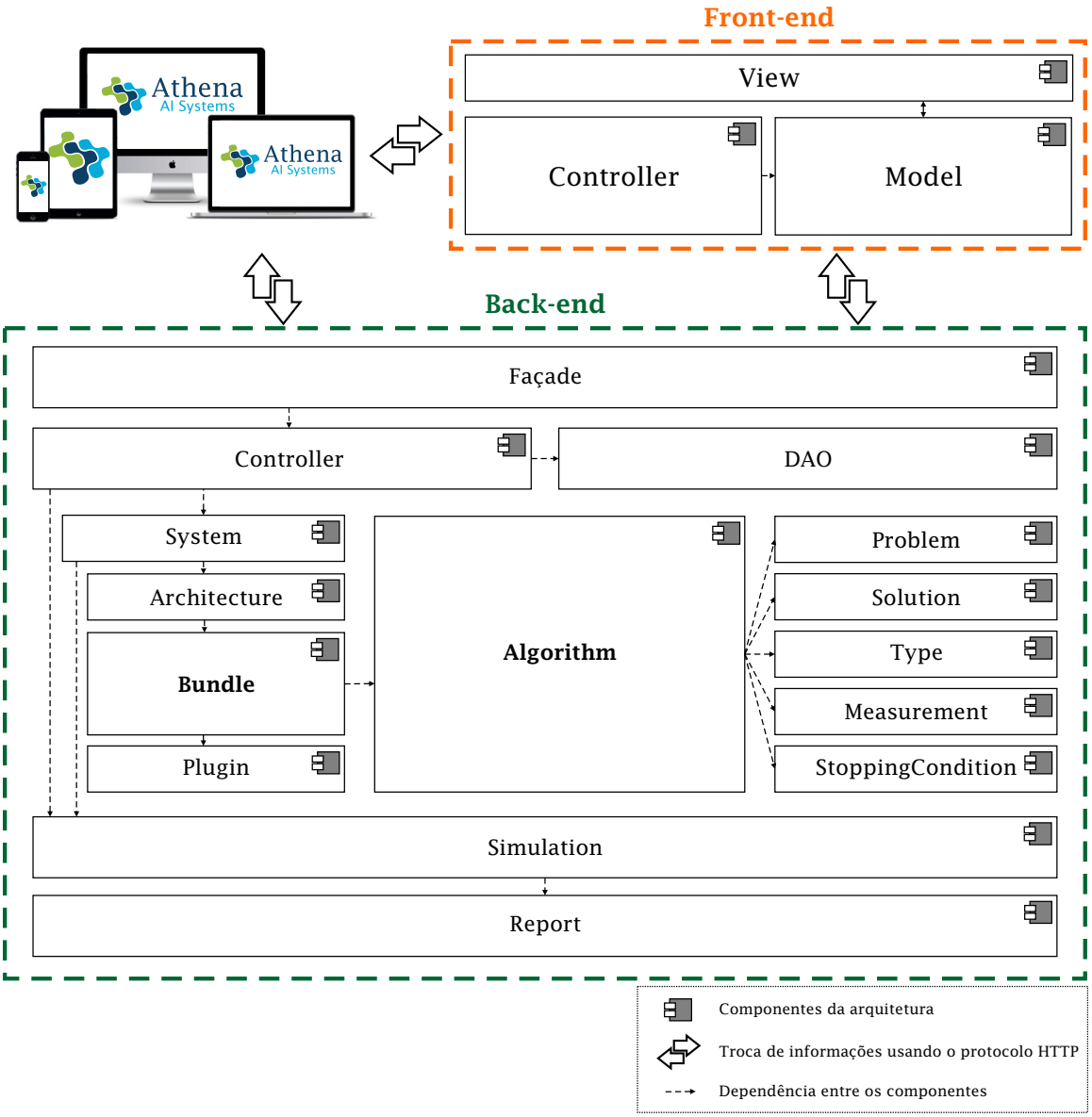


Figura 18 – Visão geral da Arquitetura Proposta (AP).

3.1.2 Estruturas Fundamentais

A Figura 18 apresenta uma visão geral da Arquitetura Proposta. Observa-se que o *Front-end* (parte do sistema que interage diretamente com o usuário) está separado fisicamente do *Back-end* (responsável pelo processamento das requisições, execução dos algoritmos e devolução dos resultados). Essa decisão foi tomada para fortalecer a interoperabilidade da arquitetura, visto que suas funcionalidades podem ser acessadas independente da Interface Gráfica do Usuário (GUI, do inglês *Graphical User Interface*).

A apresentação dos componentes segue uma abordagem *bottom-up*, ou seja, partindo das estruturas mais internas para as estruturas mais próximas dos usuários. Além disso, os componentes são apresentados desvinculados das tecnologias utilizadas no desenvolvimento da ferramenta.

3.1.3 Back-end

Conforme discutido anteriormente, a AP prescreve a existência de dois grandes subsistemas: *Back-end* e *Front-end*. O *Back-end* é responsável pelo processamento das requisições advindas do cliente, execução dos algoritmos, além da geração e armazenamento dos resultados. Análogo ao motor de um carro, o *Back-end* realiza todo o trabalho pesado para suportar as exigências do usuário.

Esse subsistema possui em seu núcleo um componente chamado *Algorithm* (apresentado em destaque na Figura 18). Esse componente é responsável por todas as definições necessárias à execução dos algoritmos de Inteligência Computacional e está presente em quase todos os trabalhos relacionados. Ele dependente diretamente dos componentes *Problem*, *Solution*, *Type*, *Measurement* e *StoppingCondition*.

O componente *Problem* é responsável por lidar com as questões relacionadas ao problema que está sendo atacado. Esse componente prescreve uma *interface* que define um comportamento padrão para todos os tipos de problema. Cada problema deve ter no mínimo um nome, o tipo do objetivo (maximização, minimização ou multiobjetivo) e uma função para avaliar a qualidade das soluções. O componente *Solution* padroniza as soluções encontradas pelos algoritmos. O componente *Type* padroniza todos os tipos de dados que trafegam dentro do *Back-end* (por exemplo, Real, Inteiro, *String*, *File*), dessa forma é possível realizar a troca de informações entre algoritmos. O componente *Measurement* cuida das medições realizadas durante as execuções (*e.g.*, tempo de execução, número de iterações) e, por fim, o componente *StoppingCondition* fica encarregado de definir as condições de parada a serem aplicadas nos algoritmos.

Em resumo, os algoritmos possuem um conjunto de medições e condições de parada. Eles são responsáveis por resolver determinado problema e retornar as soluções encontradas. Além disso, todos os dados que são utilizados pelos algoritmos são padronizados para

permitir a transferência de informações entre os algoritmos. A Figura 19 apresenta as *interfaces* presentes nesses componentes. Vale ressaltar que essa estrutura foi inspirada no modelo proposto por PAMPARA; ENGELBRECHT; CLOETE (2008).

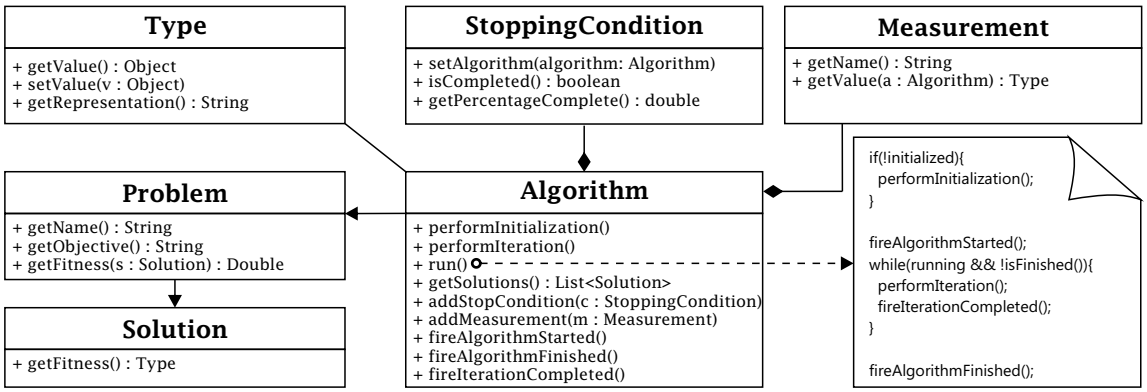


Figura 19 – Diagrama com as principais interfaces exigidas pelos componentes *Algorithm*, *Problem*, *Solution*, *Measurement*, *StoppingCondition* e *Type*.

A Figura 19 apresenta também o pseudocódigo (parte inferior direita da imagem) que define o funcionamento básico dos algoritmos: inicialização e execução das iterações até que determinada condição de parada seja atendida.

O segundo componente em escala de importância chama-se *Bundle*. Esse componente é responsável por um conceito chave dentro da Athena: o módulo. Considerando o princípio de simplicidade, a AP prescreve que todos os algoritmos de IC devem ser encapsulados dentro de um módulo que possui basicamente um conjunto de entradas (*inputs*), uma lógica interna (*logic*), configurações (*settings*) e saídas (*outputs*).

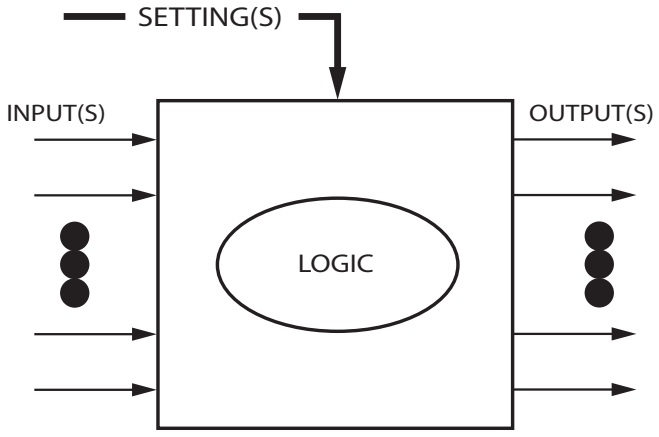


Figura 20 – Visão macro dos módulos da Athena.

A Figura 20 apresenta uma visão macro dos módulos utilizados dentro da Athena. Pode-se observar que a complexidade da lógica interna fica preservada dentro do módulo, dessa forma o usuário fica exposto apenas às entradas, configurações e às saídas resultantes. Essa decisão foi inspirada no trabalho proposto por ELLEN; CAMPBELL (2005).

Esses módulos podem ser interligados pois os dados que trafegam nas interconexões são padronizados pelo componente *Type*. Um conjunto de módulos organizados para solucionar um problema é chamado de arquitetura[1]. As arquiteturas são gerenciadas pelo componente *Architecture* e podem ser reunidas em outro conceito: o sistema. Nesse contexto, um sistema representa uma entidade capaz de aglutinar todas as informações a respeito do problema alvo. Um sistema pode conter várias arquiteturas com diferentes módulos, ou seja, para um mesmo problema o usuário pode testar e comparar a combinação de diversas técnicas. Esse modelo permite ainda o encapsulamento de arquiteturas dentro de módulos, ou seja, um módulo pode conter em sua lógica interna um arranjo de outros módulos. Essa situação é exemplificada pela Figura 21, na qual a lógica interna do Módulo A é definida pela execução dos módulos B, C, D e E.

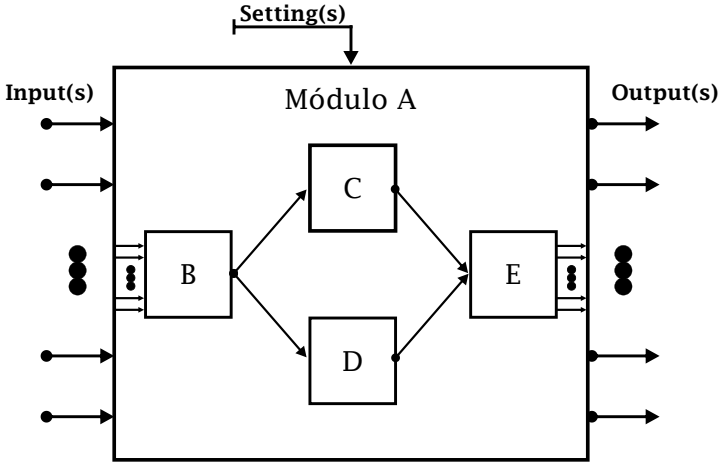


Figura 21 – Visão macro de um módulo cuja a lógica interna é definida pelo arranjo de outros módulos.

A execução dos sistemas é responsabilidade do componente intitulado *Simulation* e os relatórios de execução são organizados pelo componente *Report*. Os controles da AP ficam dentro do componente *Controller*, que por sua vez utiliza as definições do componente *DAO* para persistir todas as informações relevantes. Essas informações podem ser compartilhadas com outros pesquisadores, favorecendo o princípio da colaboração.

O componente *Façade* é responsável por ocultar a complexidade dos controles, provendo uma API de acesso simples (GAMMA et al., 1994). Esse componente recebe as requisições advindas da Internet por meio do protocolo HTTP (FIELDING et al., 1999), realiza seu processamento acionando as funções definidas no *Controller* e devolve os resultados utilizando o mesmo protocolo. Essa definição permite a adequação ao princípio de *Software* como Serviço, além de conferir independência ao *Back-end*, pois sua execução não depende de uma GUI.

[1] Nesse contexto, o termo "arquitetura" refere-se a uma reunião de módulos interconectados ou não, que tem por finalidade a resolução de um problema. Para evitar conflitos de interpretação, o termo "Arquitetura Proposta (AP)" será substituído pelo acrônimo AP. A Arquitetura Proposta refere-se a todas as definições prescritas neste estudo.

A AP também prescreve que o subsistema *Back-end* seja executado utilizando um modelo de Infraestrutura como Serviço (IaaS, do inglês *Infrastructure as a Service*) (JAMSA, 2012) que permita o ajuste dos recursos computacional (memória e processamento). Essa prescrição foi motivada pelo princípio de alta performance descrito na Seção 3.1.1.

Por fim, o princípio da extensibilidade foi obtido pela definição do componente chamado *Plugin*. Esse componente padroniza a integração dinâmica de novos módulos. Com isso, é possível adicionar novas técnicas em tempo de execução (*on-the-fly*).

3.1.3.1 Front-end

O subsistema denominado *Front-end* encontra-se mais próximo do usuário final. Portanto, possui a função de abrigar os componentes relacionados à usabilidade da AP. Nesse contexto, o termo usabilidade refere-se à eficiência, eficácia e satisfação percebida na utilização da ferramenta que implementa as definições da Arquitetura Proposta. Esse subsistema é composto por três componentes principais: *Model*, *View* e *Controller*.

O componente *Model* é responsável pela definição do modelo utilizado para materializar o *Front-end*. A AP preconiza que esse modelo seja um reflexo simplificado do modelo presente no *Back-end*, no qual um sistema possui um conjunto de arquiteturas, que são constituídas por módulos e suas ligações.

O componente *View* é responsável por tudo que o usuário final visualiza. Esse componente prescreve a criação de um editor gráfico que permita a construção e manutenção dos sistemas padronizados pelo *Back-end*. Esse editor deve seguir a abstração de módulos apresentada na Figura 20. Além disso, as execuções devem ser configuráveis e os resultados devem ser exibidos por meio de gráficos e tabelas.

Finalmente, o componente *Controller* controla as informações presentes no *Front-end*. Esse componente define o que, quando e onde as informações devem ser exibidas, além de ser responsável pela comunicação com o *Back-end*.

3.1.4 Definições de Comportamento

Nesta seção são discutidos os aspectos relacionados ao comportamento da Arquitetura Proposta. O primeiro e mais importante aspecto refere-se à prescrição do método para a construção e execução dos Sistemas Baseados em Inteligência Computacional. Esse método possui cinco etapas, são elas:

- **1ª etapa - Informações Básicas**: refere-se à criação do sistema de Inteligência Computacional informando nome, descrição e quais usuários poderão acessar suas informações. Esses dados são importantes para catalogar os sistemas criados pelos usuários;

- **2ª etapa - Arquitetura**: nessa etapa o usuário deve selecionar e conectar os módulos com o objetivo de criar uma arquitetura capaz de resolver o problema-alvo. Esse processo deve ser feito com o auxilio do editor gráfico prescrito pelo *Front-end*;

- **3ª etapa - Configuração e Execução**: nessa etapa o usuário deve configurar os módulos para habilitar a execução da arquitetura. As configurações devem ser persistidas para que seja possível comparar os resultados obtidos com diferentes parâmetros de execução;

- **4ª etapa - Análise dos Resultados**: ao final do processo de execução, o usuário pode analisar os resultados obtidos, que por sua vez serão exibidos por meio de gráficos e tabelas. Além disso, são exibidas informações adicionais coletadas durante a execução dos módulos, por exemplo, número de iterações, tempo de execução etc;

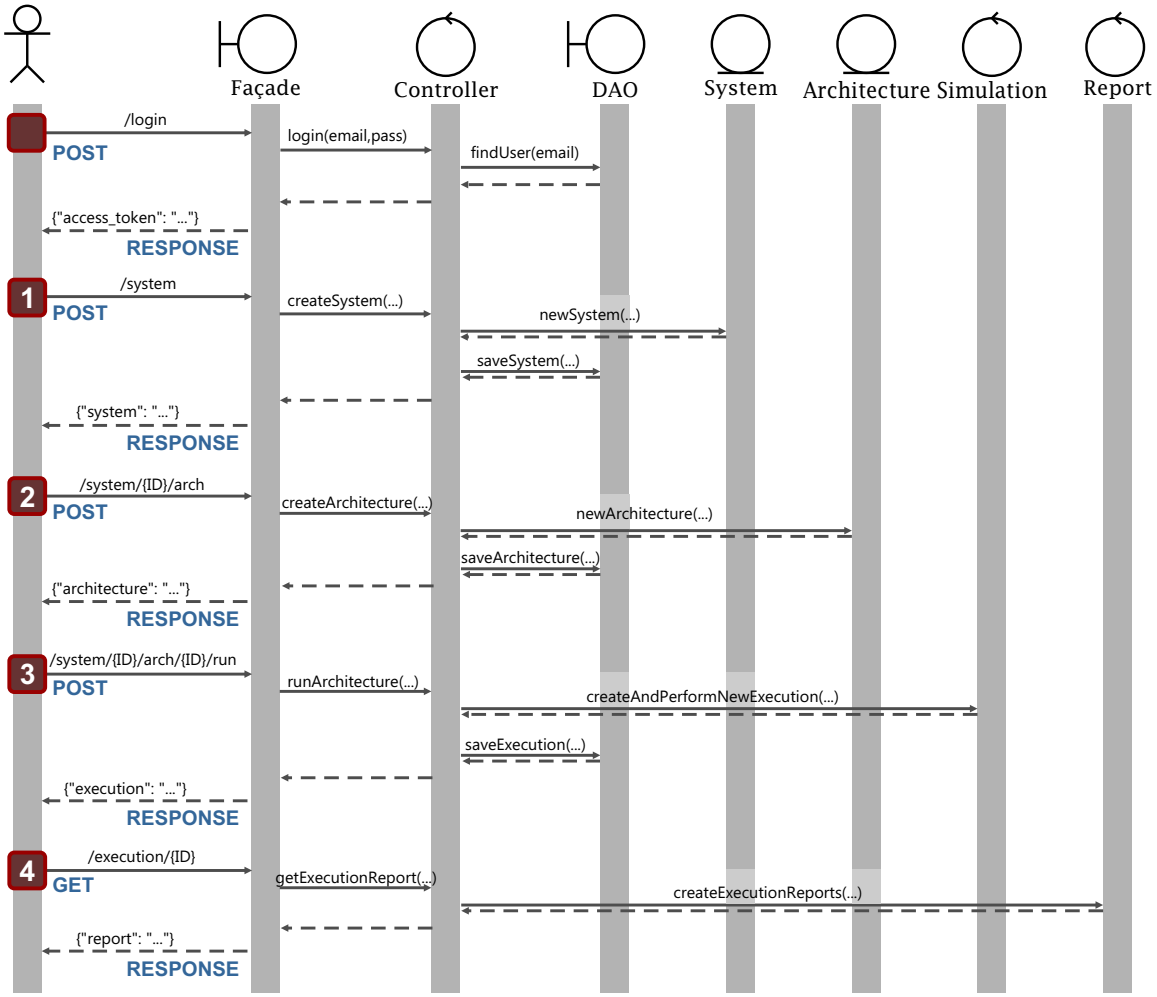


Figura 22 – Diagrama de sequência simplificado apresentando a interação ocorrida durante o processo de construção e execução dos sistemas de IC.

- **5ª etapa - Integração**: ao encontrar uma arquitetura adequada para solucionar determinado problema, o usuário pode realizar a integração entre a Athena e um sistema externo. Essa integração é feita por meio da API *RESTFul* definida pelo componente *Façade*, utilizando como insumos a chave de identificação da arquitetura a ser executada, as entradas e as configurações requeridas.

A Figura 22 apresenta um diagrama de sequência (simplificado para fins didáticos) (BOOCH; RUMBAUGH; JACOBSON, 2006) com a interação entre os componentes durante a realização das quatro primeiras etapas descritas anteriormente. Inicialmente o usuário solicita acesso à aplicação por meio da função *login*. Em seguida, é realizada a criação de um sistema de IC (1) para reunir todas as informações relacionadas ao problema que está sendo atacado. Após a construção do sistema, o usuário seleciona os módulos e define suas interconexões para criar uma arquitetura (2). Com a arquitetura definida, é necessário configurá-la para solicitar sua execução (3). Por fim, ao final da execução o usuário solicita os resultados (4), que são exibidos por meio de gráficos e tabelas. A integração com um sistema externo é exemplificada pela Figura 23 na qual há uma requisição e uma resposta HTTP para a solicitar a execução de determinada arquitetura.

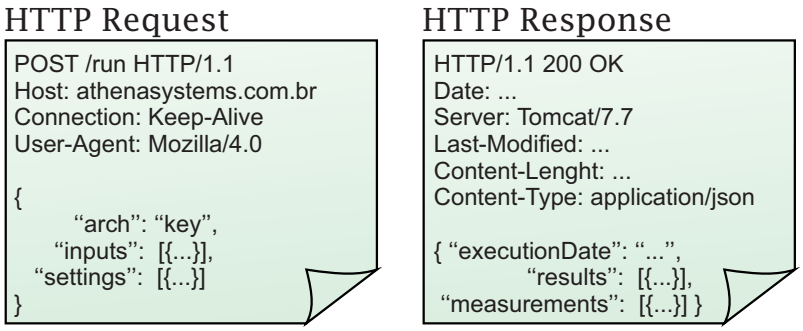


Figura 23 – Modelo de requisição e resposta HTTP utilizadas para executar determinada arquitetura a partir de um sistema externo.

Outro comportamento importante refere-se ao Algoritmo 1, o qual é responsável pela execução dos módulos de uma arquitetura. Ele recebe como insumo a lista dos módulos definida pelo usuário, os dados de entrada e as configurações de execução. É importante ressaltar que os dados de entrada só são requeridos quando o módulo possui entradas (*input*) não conectadas à saída (*output*) de outros módulos (linha 7).

Após a instanciação da estrutura que abriga os resultados (linha 4), o algoritmo itera sobre todos os módulos para realizar quatro atividades fundamentais: i) injetar os dados de entrada fornecidos pelo usuário (linhas 6-10); ii) injetar as configurações de execução (linhas 11-13); iii) realizar a execução do módulo (esse processo ativa a execução da técnica de IC embutida no módulo selecionado); e por fim, iv) a propagação das saídas para os próximos módulos (linhas 15-18). Após o término dessa iteração, o algoritmo retorna os resultados obtidos (linha 22).

Algoritmo 1 Algoritmo responsável pela execução dos módulos de uma arquitetura.

Entrada:

1: *architecture*: Lista de módulos definidos pelo usuário;
2: *inputsData*: Mapa com os dados de entrada de cada módulo;
3: *settingsData*: Mapa com as configurações de execução de cada módulo;

Saída: Mapa de resultados produzidos pela execução dos módulos;

```
4:  Resultados = novo Mapa;
5:  para cada módulo M em architecture faça
6:      para cada entrada E em M.getEntradas() faça
7:          se !E.estaConectado() então
8:              E.adicionarDados(inputsData.get(M.key));
9:          fim se
10:     fim para
11:     para cada configuracao C em M.getConfiguracoes() faça
12:         C.adicionarDados(settingsData.get(M.key));
13:     fim para
14:     M.executar();
15:     se M.getErros() está vazia então
16:         Resultados.put(M.key, M.getResultados);
17:         M.propagarSaídas();
18:     senão
19:         finalizar;
20:     fim se
21: fim para
22: retornar Resultados;
```

A Figura 24 ilustra o funcionamento do Algoritmo 1. Na imagem pode-se observar dois módulos (A e B). O módulo A possui três entradas (X_1^A, X_2^A e X_3^A) e duas saídas (Y_1^A e Y_2^A), enquanto que o módulo B possui duas entradas (X_1^B e X_2^B) e apenas uma saída (Y_1^B). Além disso, esses módulos estão conectados por duas ligações ($Y_1^A \rightarrow X_1^B$ e $Y_2^A \rightarrow X_2^B$).

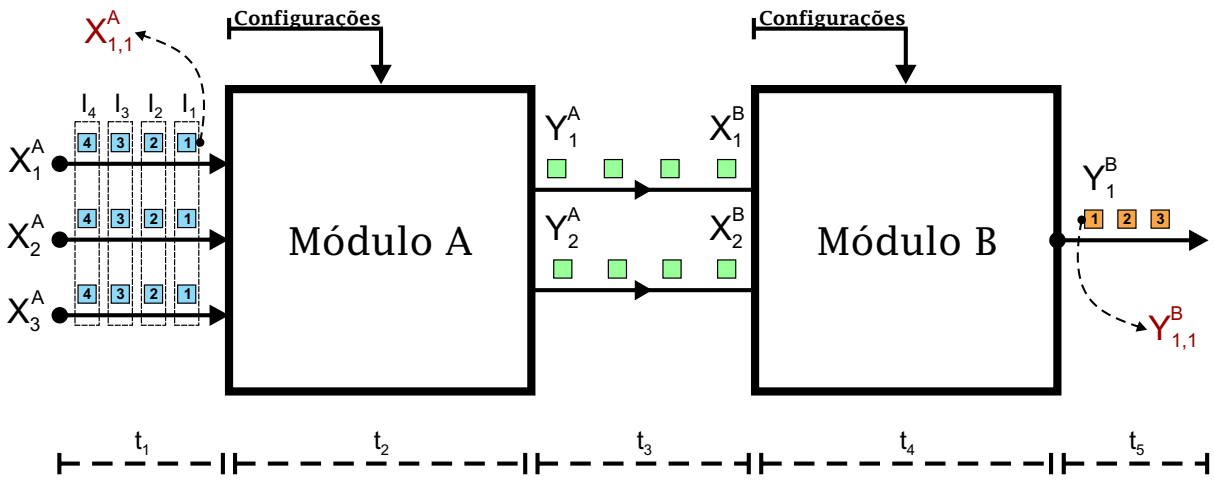


Figura 24 – Modelo de interação entre dois módulos da Athena.

Os dados que trafegam entre os módulos estão representados pelos quadrados menores e são identificados por $X^A_{i,j}$, quando se referem à variável de entrada, e por $Y^A_{i,j}$, quando se referem à variável de saída. Em ambos, i representa o índice da entrada e j representa a ordem de chegada/saída da informação ao módulo. Por exemplo, o primeiro dado que chega ao módulo A pela entrada 1 é identificado como $X^A_{1,1}$. De forma análoga, o primeiro valor que deixa o módulo B pela saída 1 é identificado como $Y^B_{1,1}$. Esses dados são transmitidos em blocos de forma síncrona e serializada. Considerando todas as variáveis de entrada, pode-se definir uma instância de entrada (I_k) como o conjunto dos dados com ordem de chegada igual a k. Essa definição é similar para as variáveis de saída.

Conhecendo todas essas variáveis é possível analisar a execução dos módulos. Em t_1, quatro instâncias de entrada chegam ao módulo A (I_1, I_2, I_3 e I_4). Esse módulo utiliza as configurações de execução para ajustar o processamento dessas informações. Após a realização desse processamento (em t_2), os resultados são propagados (em t_3) para o módulo B por meio das ligações ($Y^A_1 \rightarrow X^B_1$ e $Y^A_2 \rightarrow X^B_2$). Em t_4, o módulo B realiza sua execução e em t_5 tem-se os resultados finais. Vale ressaltar que a quantidade de instâncias de entrada nem sempre é igual à quantidade de instâncias de saída. Essa relação depende do algoritmo implementado pelo módulo e, portanto, não pode ser restringida pela Arquitetura Proposta.

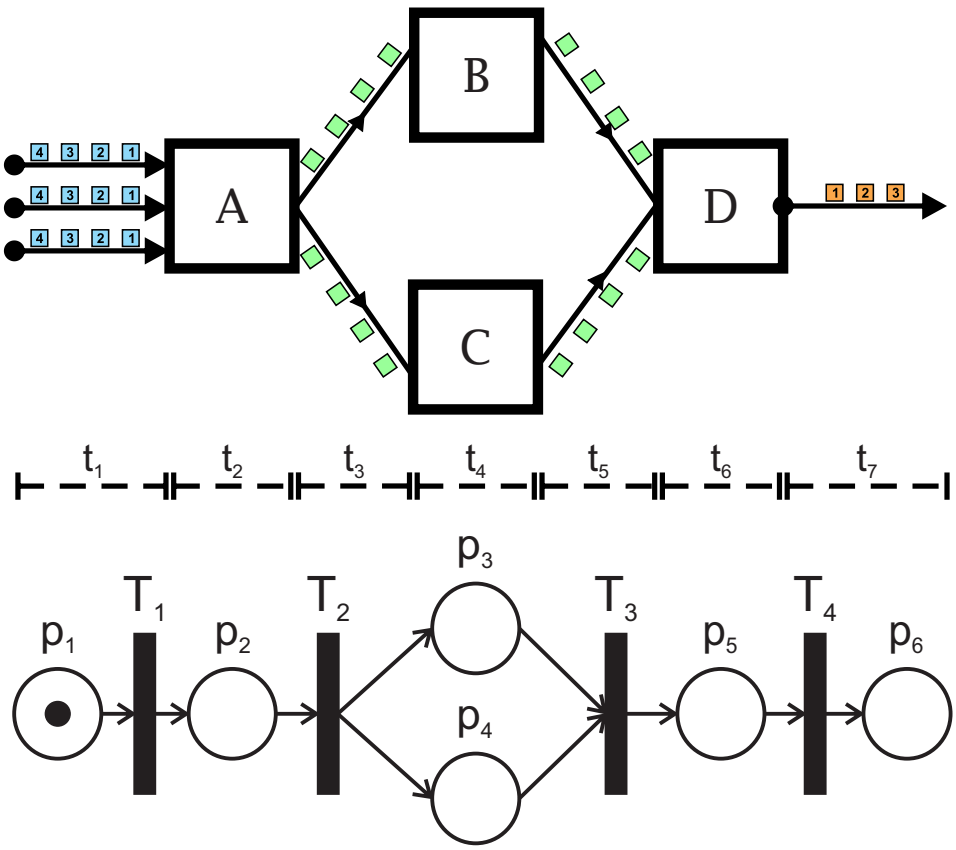


Figura 25 – Representação de uma arquitetura de acordo com o modelo de Redes de Petri.

O Algoritmo 1 exibe o passo-a-passo necessário para a execução sequencial dos módulos pertencentes a uma arquitetura. Ele foi descrito dessa forma para facilitar a compreensão das ações envolvidas nessa tarefa, entretanto a AP prescreve que a execução dos módulos deve ser realizada de forma concorrente sempre que possível. Para isso, utilizou-se uma técnica de modelagem que permite a construção de sistemas paralelos, concorrentes, assíncronos e não-determinísticos chamada Redes de Petri (PETRI, 1962).

Uma Rede de Petri (RdP) possui três elementos básicos: lugar, transição e ficha. Os lugares, representados por um círculo, podem ser interpretados como uma condição, estado parcial, uma espera ou procedimento a ser realizado. As transições, representadas por uma barra, são associadas a um evento que ocorre no sistema. As fichas, representadas por um ponto, são indicadores significando que a condição associada ao lugar é verificada. Formalmente, defini-se uma RdP por meio de uma quádrupla $R = (P, T, F, M_0)$, onde $P = \{p_1, p_2, ..., p_n\}$ é um conjunto finito não-vazio de lugares, $T = \{t_1, t_2, ..., t_n\}$ é um conjunto finito não-vazio de transições. F é o conjunto de arcos que conectam os lugares às transições e M_0 representa a marcação inicial (CARDOSO; VALETTE, 1997).

A Figura 25 expõe a representação de uma arquitetura de acordo com o modelo de Redes de Petri. A ficha encontra-se no lugar p_1, indicando que a arquitetura está pronta para ser executada. Os locais p_2, p_3, p_4 e p_5 equivalem a execução da lógica interna dos módulos A, B, C e D, respectivamente. O processo se inicia com a ativação da transição T_1 que define o evento *iniciar operação*. Após a execução do primeiro módulo, representado pelo lugar p_2, a transição T_2 é ativada iniciando a execução em paralelo dos módulos B e C. Quando os dois módulos finalizam sua execução, a transição T_3 é acionada e a lógica interna do módulo D é executada. Por fim, a transição T_4 indica o término da execução da arquitetura e o lugar p_6 define o estado *execução concluída*.

3.2 Modelo Tecnológico

Diversas tecnologias foram adotadas para materializar as exigências da Arquitetura Proposta (AP). A Figura 26 apresenta as principais tecnologias escolhidas.

O *Back-end* foi construído sobre a infraestrutura provida pelo *Amazon Elastic Compute Cloud - EC2*. O EC2[2] é um serviço fornecido pela *Amazon* com capacidade de computação redimensionável em nuvem. Ele foi projetado para facilitar a obtenção e configuração de recursos computacionais para aplicações Web. Esse serviço foi escolhido por estar de acordo com as restrições da arquitetura e por possibilitar a inclusão de novos módulos em tempo de execução ("*on-the-fly*"). Outros serviços de Computação em Nuvem possuem restrições que impedem tal comportamento (JAMSA, 2012).

[2] Amazon EC2 - disponível em http://aws.amazon.com/pt/.

Dessa forma, para facilitar essa inclusão *on-the-fly* de JARs[3], utilizou-se parte das especificações da arquitetura OSGi, que foi concebida com o objetivo de auxiliar o desenvolvimento e implantação de aplicações modulares em Java (ALLIANCE, 2003).

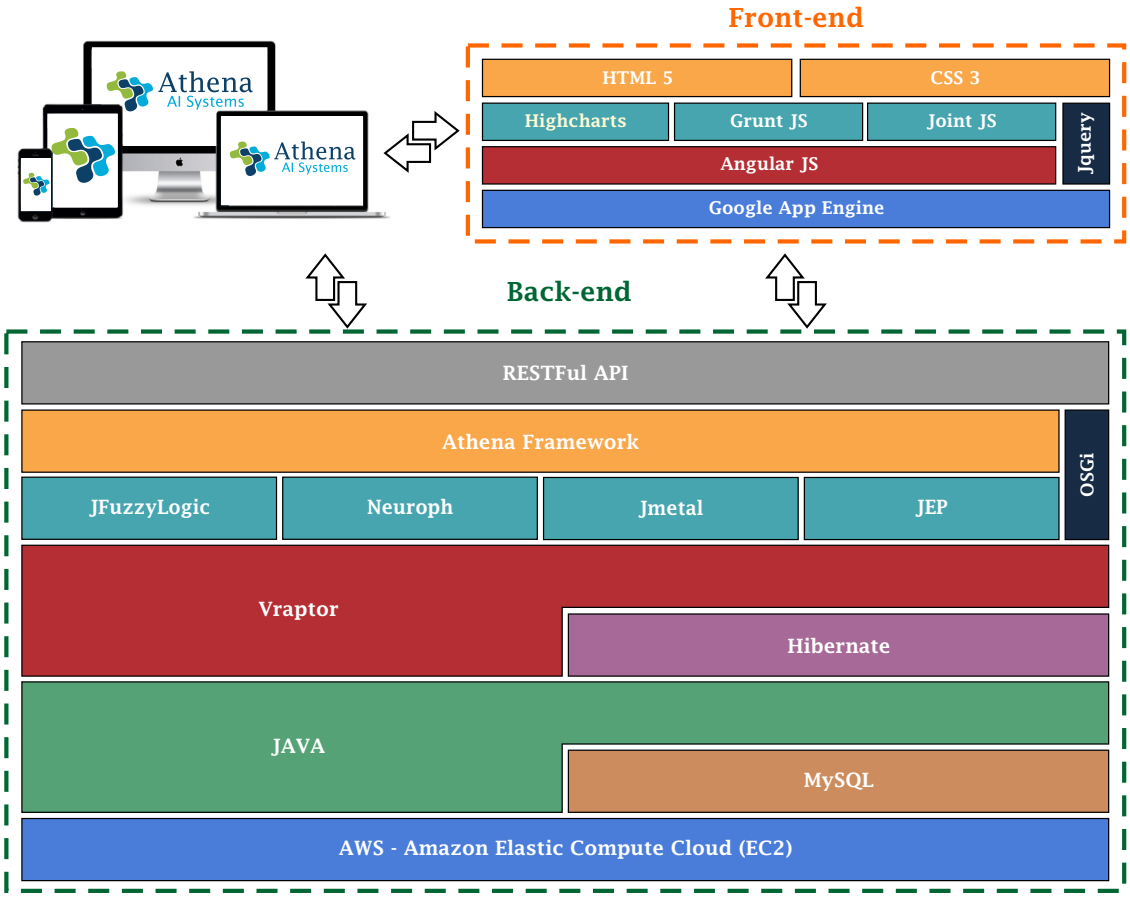


Figura 26 – Esquema com as principais tecnologias utilizadas durante o desenvolvimento da ferramenta.

Adotou-se o sistema de gerenciamento de banco de dados MySQL[4] e para facilitar o mapeamento objeto-relacional foi utilizado o *framework* Hibernate[5]. O VRaptor[6] foi escolhido como *framework* de desenvolvimento de aplicações Web por garantir alta produtividade, ter uma curva de aprendizado muito acentuada, ser adaptado ao padrão *RESTFul* e possuir documentação em português. Essa escolha implicou na linguagem de programação JAVA (GOSLING, 2000).

Para auxiliar a implementação das técnicas de IC, foram utilizadas quatro ferramentas bem conhecidas: JFuzzyLogic[7], Neuroph[8], JMetal[9] e WEKA[10]. A JFuzzyLogic é

[3] JAR é um arquivo compactado usado para distribuir uma aplicação desenvolvida em JAVA.
[4] MySQL - disponível em http://www.mysql.com.
[5] Hibernate - disponível em http://hibernate.org.
[6] VRaptor - disponível em http://www.vraptor.org.
[7] JFuzzyLogic - disponível em http://jfuzzylogic.sourceforge.net.
[8] Neuroph - disponível em http://neuroph.sourceforge.net.
[9] JMetal - disponível em http://jmetal.sourceforge.net.
[10] WEKA - disponível em http://www.cs.waikato.ac.nz/ml/weka.

especializada na construção de sistemas de inferência *fuzzy*, a Neuroph auxilia a concepção de Redes Neurais Artificiais, a JMetal facilita o desenvolvimento de algoritmos evolutivos e a WEKA dispõe de diversos algoritmos adequados para resolver problemas que envolvem Aprendizagem de Máquina. Além disso, utilizou-se a biblioteca JEP[11] para suprir a necessidade de um mecanismo capaz facilitar a definição de funções matemáticas usando linguagem natural. Essa extensão possui uma gramática de expressões matemáticas que inclui diversas variáveis, constantes e funções.

O núcleo da ferramenta Athena está apoiado em todas as tecnologias citadas até o momento e seus serviços estão disponíveis a partir de uma API *RESTFul* (RICHARDSON; RUBY, 2008), capaz de processar requisições HTTP com mensagens no formato JSON (do inglês, *Javascript Object Notation*) (CROCKFORD, 2006).

O subsistema *Front-end* foi construído sobre a plataforma provida pelo *Google App Engine - GAE*. O GAE[12] é a plataforma do Google para desenvolvimento e hospedagem de aplicativos Web. Esse serviço foi escolhido por permitir a construção de aplicativos de alto tráfego sem a necessidade de gerenciar a infraestrutura necessária para isso. Dessa forma, *Back-end* e *Front-end* estão separados fisicamente e se comunicam utilizando apenas a troca de mensagens via Internet.

Como a Arquitetura Proposta prescreve que o *Front-end* seja baseado no padrão MVC (GAMMA et al., 1994), decidiu-se pelo *framework* AngularJS[13]. Além disso, o AngularJS é adaptado ao padrão *RESTFul*, possui dupla sincronização entre a *View* e o *Model* e tem curva de aprendizado semelhante a do VRaptor (muito acentuada). A biblioteca JointJS[14] foi utilizada para a construção do editor gráfico dos sistemas de IC e a Highcharts[15] para a exibição dos resultados em formato gráfico.

A construção da aplicação (*build*) é responsabilidade do *framework* GruntJS[16]. Com o GruntJS são automatizadas tarefas como a minimização e concatenação dos arquivos (*.html, .css, .js*), otimização das imagens e empacotamento final do subsistema *Front-end*. Todos esses *frameworks* utilizam a linguagem Javascript (FLANAGAN, 2011) e possuem como dependência a biblioteca JQuery[17].

Os componentes estéticos e a identidade visual da ferramenta foram desenvolvidos utilizando HTML5 e CSS3 (FRAIN, 2012). Nas páginas iniciais adotou-se o termo AI (do inglês, *Artificial Intelligence*) por ser mais comum do que CI (do inglês, *Computacional Intelligence*), entretanto a diferença é esclarecida na documentação[18].

11 JEP - disponível em http://www.cse.msu.edu/SENS/Software/jep-2.23/doc/website/index.html.
12 Google App Engine (GAE) - disponível em http://cloud.google.com.
13 AngularJS - disponível em http://angularjs.org.
14 JointJS - disponível em http://www.jointjs.com.
15 Highcharts - disponível em http://www.highcharts.com.
16 GruntJS - disponível em http://gruntjs.com.
17 JQuery - disponível em http://jquery.com.
18 A documentação da ferramenta pode ser acessada em http://goo.gl/avgA51.

4 Ferramenta

A Athena foi desenvolvida seguindo as prescrições da arquitetura apresentada no capítulo anterior. Este capítulo apresenta a visão geral da ferramenta, os módulos disponíveis e alguns exemplos de utilização.

4.1 Visão Geral

A Athena permite a construção, manutenção e aplicação de Sistemas Híbridos de Inteligência Computacional de forma simples e escalável, utilizando um editor gráfico que propicia a programação visual das técnicas de IC. Esses sistemas podem ser compartilhados com outros pesquisadores para que seja possível discussões sobre os algoritmos selecionados e até mesmo sobre os resultados obtidos. O ambiente pode facilitar a troca de conhecimento entre grupos de pesquisa espalhados pelo mundo. A Figura 27 apresenta a página inicial da ferramenta que pode ser acessada no endereço http://athenasystems.com.br.

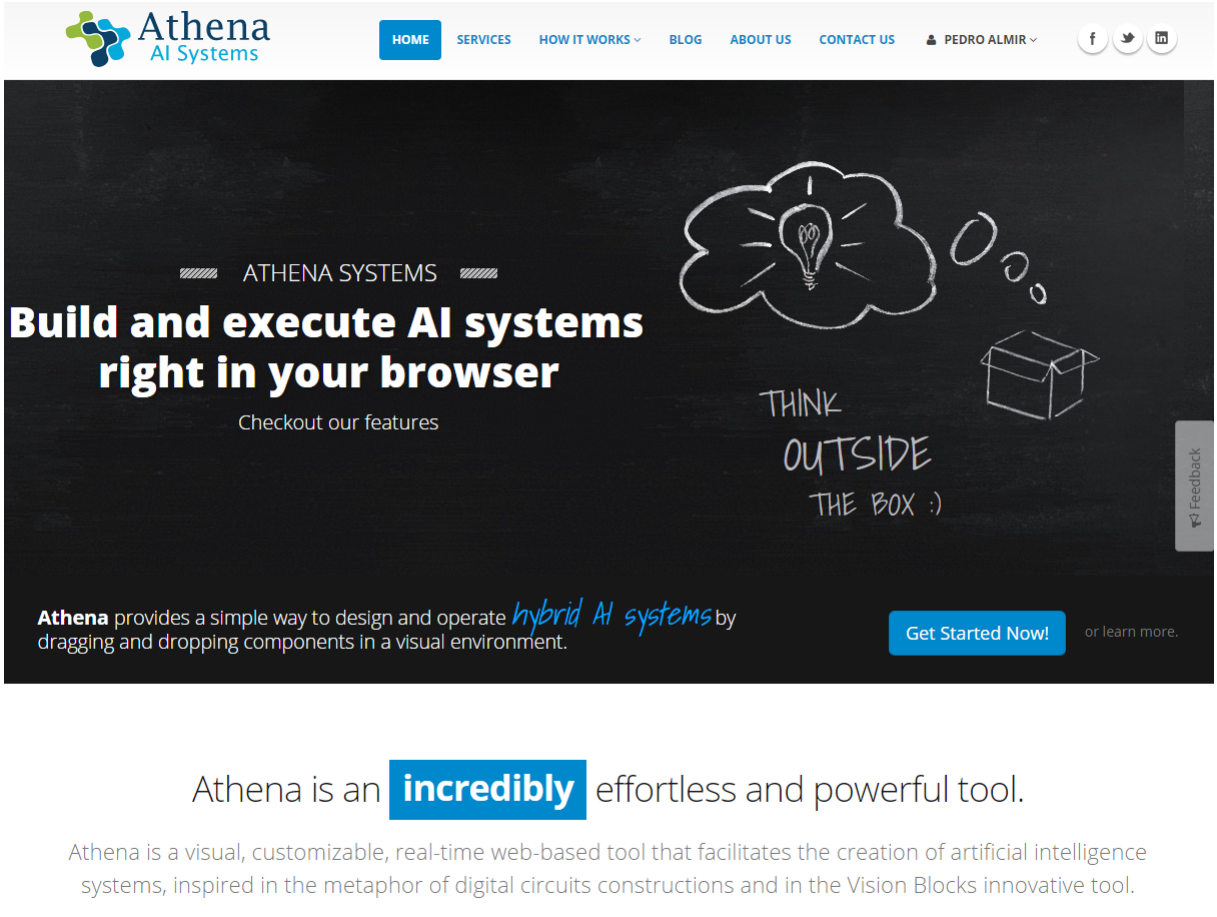


Figura 27 – Página inicial da ferramenta.

Atualmente, a Athena encontra-se em sua segunda versão e possui mais de 10.000 linhas de códigos dividas entre os subsistemas *Front-end* e *Back-end*. A ferramenta está em processo de registro no Instituto Nacional da Propriedade Industrial (INPI) - código do processo: BR 51 2015 000521 9.

A Figura 28 apresenta o editor gráfico desenvolvido no *Front-end*. Esse componente é fundamental para facilitar a construção dos sistemas de IC. Nessa figura, observa-se a palheta com os módulos disponíveis (lado esquerdo). Essa palheta possui um campo de busca e os módulos são categorizados de acordo com a sua função ou a subárea da IC ao qual pertencem. Ao clicar em um módulo, ele será automaticamente incluído na área de edição que fica no centro da imagem. Esse espaço de edição pode ser ajustado utilizando as opções exibidas no canto inferior direito da tela. O menu de contexto (lado direito), no qual é possível incluir ou remover entradas e saídas de um módulo, é acionado ao clicar nos módulos presentes na área de edição. Por fim, a barra de ações no canto superior direito permite ao usuário salvar (*Save*) e executar (*Run*) a arquitetura, reiniciar os trabalhos limpando todos os módulos presentes no espaço de edição (*Reset*), voltar (*Back*) e exibir informações sobre o funcionamento do editor (*Help*).

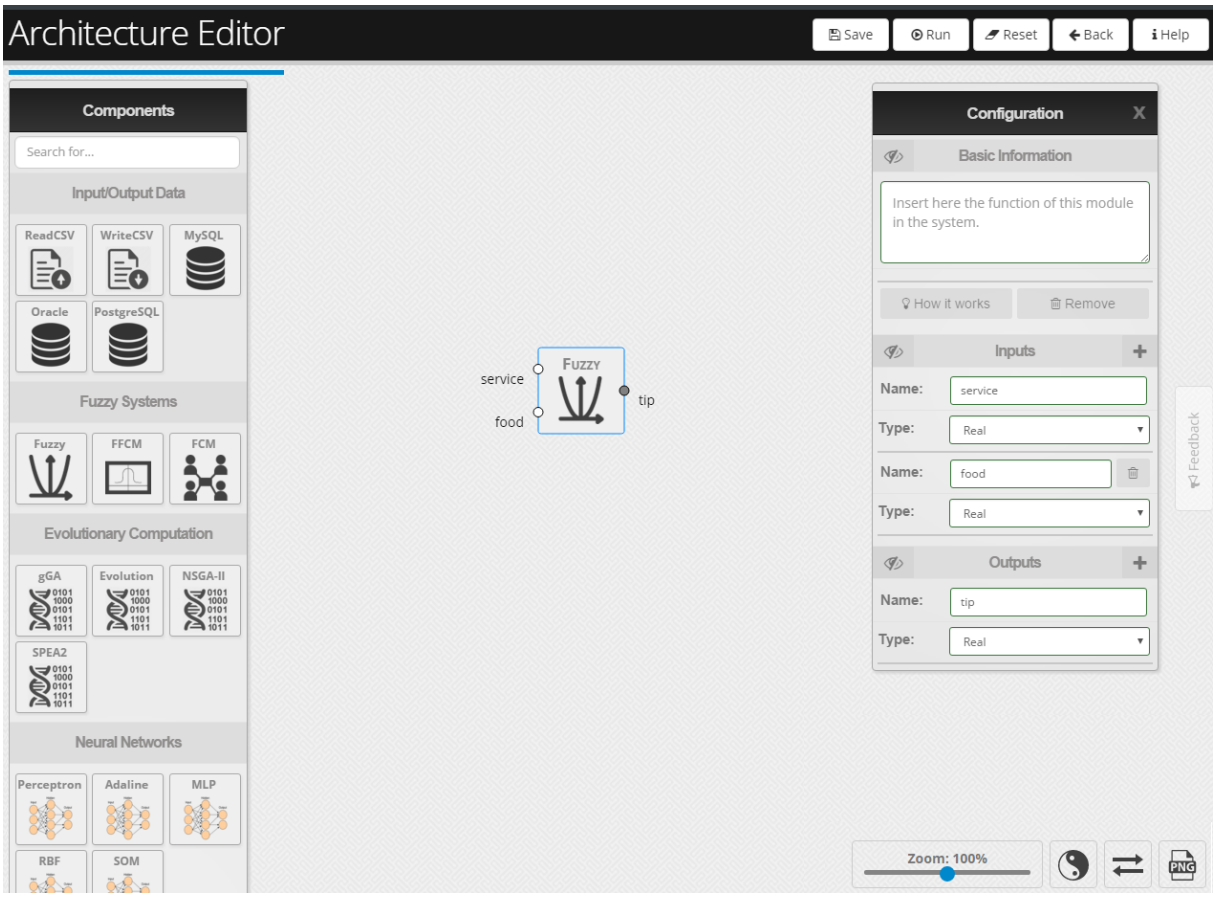


Figura 28 – Editor gráfico desenvolvido no *Front-end* da Athena.

4.2 Módulos Implementados

O processo de desenvolvimento da Athena começou em 2014 e sua primeira versão contava apenas com três algoritmos de IC (*Fuzzy*, *AntSystem* e NSGA-II) e dois módulos conversores de dados (ReadCSV e WriteCSV). Hoje, a Athena dispõe de 35 módulos subdivididos em cinco categorias:

- **Módulos de Inteligência Computacional**: esse tipo de módulo encapsula os algoritmos de IC, funcionando como motor da ferramenta. Atualmente, a Athena dispõe de pelo menos um módulo para cada subárea da IC e com a absorção dos algoritmos do WEKA foi possível disponibilizar alguns algoritmos de Aprendizagem de Máquina. Dentre os módulos mais importantes dessa categoria, destacam-se: Fuzzy, Fuzzy C-Means, algoritmo genético (gGA), NSGA-II, SPEA2, MLP, RBF, AntSystem, MMAS, PSO, CLONALG, ID3, J48, K-Means e Cobweb;

- **Módulos conversores de dados**: esse tipo de módulo permite a conversão dos dados que trafegam nas ligações do sistema. Além disso, podem ser utilizados para injetar dados nas execuções ou para extrair os resultados em formato de arquivo. Essa categoria dispõe de dois módulos pré-implementados: ReadCSV e WriteCSV para leitura e escrita em arquivos CSV, respectivamente;

- **Módulos de acesso à base de dados**: esses permitem o acesso à bancos de dados externos para extrair entradas para as execuções. Atualmente, três SGBDs são suportados: MySQL[1], PostgreSQL[2] e Oracle[3];

- **Módulos auxiliares**: essa categoria inclui os módulos que auxiliam a construção dos sistemas de IC. Por exemplo, módulo de expressões matemáticas para realização de operações entre os dados, módulo de regressão linear para análise dos dados, módulo estatístico para realização de testes estatísticos, dentre outros;

- **Módulos dos Usuários**: os usuários da Athena podem submeter novos módulos para a ferramenta. Esses módulos são analisados por uma equipe técnica antes de sua disponibilização para os demais usuários (ver Apêndice D). Atualmente, essa categoria possui apenas dois módulos: o Textfy - realiza análise de sentimento em textos escritos em inglês - e o Twitter - coleta postagens na rede social *Twitter*.

As tabelas 8, 9 e 10 apresentam de forma resumida (a fim de não tornar a leitura cansativa) as informações sobre os módulos embutidos na Athena. Para maiores detalhes, a documentação da ferramenta detalha o funcionamento de cada um dos módulos.

[1] MySQL - disponível em http://www.mysql.com.
[2] PostgreSQL - disponível em http://www.postgresql.org.
[3] Oracle - disponível em https://www.oracle.com/database.

Tabela 8 – Módulos embutidos na Athena (1 - 14).

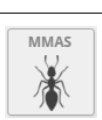
#	Módulo	Descrição
1		Implementa o Sistema de Inferência *Fuzzy* de Mamdani (MAMDANI, 1977). Nesse módulo os usuários podem definir m entradas e n saídas do tipos Inteiro, Real ou *String*. Tanto as entradas quanto as saídas devem estar de acordo com as definições do arquivo de configuração FCL (*Fuzzy Control Language*).
2		Implementa uma variação do Sistema de Inferência *Fuzzy* de Mamdani no qual as funções de pertinência das variáveis de entrada são definidas a partir da execução do algoritmo de agrupamento Fuzzy C-Means (ROSS, 2009). Nesse módulo os usuários podem definir m entradas e n saídas do tipo Real.
3		Encapsula o algoritmo de agrupamento Fuzzy C-Means (BEZDEK; EHRLICH; FULL, 1984). Esse algoritmo consiste na versão *fuzzy* do algoritmo k-Means. Os usuários podem definir m entradas do tipo Real e o resultado é uma matriz que relaciona o grau de pertinência de cada ponto em relação aos centroides identificados.
4		Esse módulo implementa um algoritmo genético geracional no qual toda a população é substituída a cada geração (EBERHART, 2007). Ele pode ser utilizado para resolver problemas que envolvem otimização discreta, problema da mochila, menor caminho e caixeiro viajante. Nesse módulo o número de entradas deve ser igual ao número de saídas.
5		Implementa um algoritmo baseado em estratégia evolutiva proposto inicialmente por RECHENBERG (1965) e aperfeiçoado por SCHWEFEL (1975). Como esse módulo é possível resolver problemas que envolvem otimização discreta, problema da mochila, menor caminho e caixeiro viajante. O número de entradas deve ser igual ao número de saídas.
6		Implementa a meta-heurística multiobjetivo NSGA-II (DEB et al., 2002) para resolver problemas combinatórios nos quais as variáveis são binárias (PARKER; RARDIN, 2014). Nesse módulo os usuários podem definir m entradas, mas apenas 2 saídas são permitidas. Essa restrição deve-se ao fato de que atualmente esse módulo trabalha apenas com dois objetivos.
7		Implementa a meta-heurística multiobjetivo intitulada SPEA2 (ZITZLER et al., 2001) para resolver problemas combinatórios nos quais as variáveis são binárias. Similar ao módulo NSGA-II, os usuários podem definir m entradas, mas apenas 2 saídas são permitidas. Essa restrição deve-se ao fato de que atualmente esse módulo trabalha apenas com dois objetivos.
8		Implementa a configuração mais simples de uma Rede Neural Artificial, a rede *Perceptron* (ROSENBLATT, 1958). Nesse módulo os usuários podem definir n entradas e apenas uma saída do tipo Inteiro ou Real. Esse módulo permite a resolução de problemas de classificação com duas classes.
9		Implementa a RNA proposta por Widrow e Hoff em 1960, conhecida como rede *Adaline* (WIDROW; HOFF, 1960). Ele se diferencia do Perceptron pela regra de aprendizado que é capaz de encontrar o hiperplano de separação ótimo se as classes forem linearmente separáveis (SILVA; SPATTI; FLAUZINO, 2010).
10		Implementa a rede Perceptron Multicamadas (RUMELHART et al., 1988) com *backpropagation*. Essa rede é caracterizada pela presença de pelo menos uma camada escondida entre a camada de entrada e a de saída. Esse módulo permite a resolução de problemas de classificação com mais de duas classes, aproximação de funções e predição de séries.
11		Implementa a RNA chamada rede de Funções de Base Radial (RBF) (HAYKIN, 2001). Nesse módulo os usuários podem definir m entradas e n saídas do tipo Inteiro ou Real. Esse módulo permite a resolução de problemas de classificação com mais de duas classes e aproximação de funções.
12		Abriga os mapas auto-organizáveis de Kohonen, também denominados de SOM (*self-organization maps*) (KOHONEN; SOMERVUO, 1998). Esse tipo de RNA possui estrutura reticulada com conexões laterais entre os neurônios. Os usuários podem definir m entradas e n saídas do tipo Real ou String para resolver problemas de agrupamento de dados.
13		Implementa a meta-heurística *Ant System* (AS) (DORIGO; MANIEZZO; COLORNI, 1996). Ela possibilita a resolução de problemas modelados como o problema da mochila binária e caixeiro viajante. As configurações requeridas são: número de formigas, máximo de iterações, as constantes *alpha*, *beta* e Q, estratégia de atualização de feromônio, a taxa de evaporação, a heurística, a função *fitness* e as restrições aplicadas ao problema.
14		Implementa a meta-heurística *Max-Min Ant System* (STÜTZLE; HOOS, 2000). Essa meta-heurística diverge do *Ant System* em três aspectos: i) apenas a melhor solução é utilizada no processo de atualização do feromônio; ii) existe um intervalo que define o máximo e mínimo do feromônio; e iii) as trilhas de feromônio são inicializadas com o valor máximo permitido.

Tabela 9 – Módulos embutidos na Athena (15 - 28).

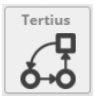
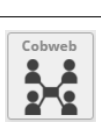
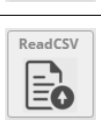
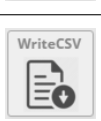
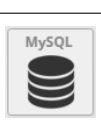
#	Módulo	Descrição
15		Implementa o algoritmo CLONALG (CASTRO; ZUBEN, 2002). Essa técnica de IC é inspirada no princípio de seleção clonal que é utilizado para explicar o funcionamento de uma resposta imune adaptativa ao estímulo de um antígeno. Ele pode ser utilizado para resolver problemas que envolvem reconhecimento de padrões.
16		Implementa a técnica de aprendizagem supervisionada intitulada *Artificial Immune Recognition System* (AIRS) (WATKINS; BOGGESS, 2002). Esse algoritmo também foi inspirado nos mecanismos adaptativos presentes nos Sistemas Imunológicos Naturais e é adequado para resolver problemas de classificação.
17		Abriga o algoritmo ID3 (MITCHELL, 1997). Essa técnica utiliza a estrutura de árvores de decisão para representar as informações relacionadas ao processo em análise. Após o modelo treinado, é possível avaliar e classificar novas instâncias (QUINLAN, 1987). Para executar esse módulo é necessário informar um arquivo de treinamento no formato ARFF.
18		Esse módulo implementa uma versão *open source* do algoritmo C4.5 (QUINLAN, 1993). O J48 utiliza o atributo com maior razão de ganho para subdividir a árvore de decisão até que as instâncias estejam separadas por classes nas folhas da árvore. Ele se difere do ID3 pela capacidade de lidar com atributos contínuos e discretos.
19		Implementa a técnica de classificação conhecida como *Random Decision Tree* (BREIMAN, 1984). Esse algoritmo escolhe aleatoriamente os atributos para cada nó da árvore de decisão. Assim como os módulos ID3 e J_48, é necessário informar um arquivo de treinamento no formato ARFF e, após o treinamento, o módulo poderá classificar novas instâncias.
20		Encapsula o algoritmo *Random Forest*. Essa técnica de AM permite a resolução de problemas classificação por meio de múltiplas árvores de decisão. Dessa forma, o *Random Forest* acaba sendo menos suscetível a ruídos e atributos que dificultam o aprendizado (BREIMAN, 2001).
21		O algoritmo Apriori é utilizado para minerar padrões frequentes em bases de dados. Em sua implementação na Athena, o Apriori requer apenas um conjunto de dados em formato ARFF para retornar as regras de associação mais frequentes (AGRAWAL; SRIKANT, 1994).
22		Essa técnica é similar ao algoritmo Apriori, entretanto o Tertius utiliza um ajuste baseado na distribuição qui-quadrado para tornar a busca de regras mais eficiente. Cada regra encontrada possui dois valores associados: a probabilidade de que a regra seja verdadeira e a frequência de contra-exemplos identificados (FLACH; LACHICHE, 1999).
23		Implementa o algoritmo de agrupamento intitulado K-Means (HARTIGAN; WONG, 1979). Essa técnica particiona um conjunto de dados criando grupos com elemento similares. Em geral, a métrica de similaridade utilizada pelo K-Means é a distância euclidiana. Por padrão, nos módulos que implementam algoritmos de agrupamento a primeira saída define o grupo ao qual a instância pertence e as demais saídas são utilizadas para propagar as entradas.
24		O Cobweb é um algoritmo de aprendizado não-supervisionado proposto por FISHER (1987). Essa técnica realiza o agrupamento incremental de uma base de dados utilizando a estrutura de uma árvore. Nesse módulo os usuários podem definir m entradas, mas por padrão a primeira saída representa o grupo e as demais saídas apenas propagam as entradas.
25		Permite a conversão de arquivos CSV em dados de entradas para a execução dos sistemas. Para cada linha ou coluna do arquivo é criada uma lista de valores que pode ser conectada às entradas e outros módulos. A definição da orientação de leitura (linha ou coluna) e se arquivo possui cabeçalho são as duas configurações exigidas.
26		Esse módulo permite a conversão dos resultados obtidos durante a execução dos sistemas em arquivos CSV (arquivo com dados em formato tabular).
27		Possibilita o acesso a bancos de dados MySQL. As configurações requeridas são: URL de acesso ao banco, nome de usuário, senha e nome da tabela a ser extraída. Esse tipo de módulo não possui entradas e as saídas devem conter o mesmo nome das colunas da tabela que está sendo requisitada.
28		Possibilita o acesso a bancos de dados PostgreSQL. As configurações requeridas são: URL de acesso ao banco, nome de usuário, senha e nome da tabela a ser extraída. Não possui entradas e as saídas devem conter o mesmo nome das colunas da tabela que está sendo requisitada.

Tabela 10 – Módulos embutidos na Athena (29 - 35).

#	Módulo	Descrição
29		Possibilita o acesso a bancos de dados Oracle. As configurações requeridas são: URL de acesso ao banco, nome de usuário, senha e nome da tabela a ser extraída. Não possui entradas e as saídas devem conter o mesmo nome das colunas da tabela que está sendo requisitada.
30		Permite a realização de operações matemáticas sobre os dados que trafegam nos sistemas de IC. As configurações exigidas são: a expressão aritmética a ser realizada e se essa operação deve ser executada considerando o tipo de dados Matriz.
31		Esse módulo permite a realização de algumas análises estatísticas, por exemplo, média, média geométrica, média quadrática, variância e desvio padrão.
32		Implementa um algoritmo de regressão linear. Esse módulo recebe como entrada duas listas de valores reais, representando as coordenadas x e y dos pontos analisados, e retorna como saída o coeficiente linear e angular da reta que aproxima a função descrita pelos pontos.
33		Esse módulo possibilita a normalização de atributos numéricos entre 0 e 1, permitindo o ajuste de escala entre diferentes atributos. Esse processo é importante para facilitar o aprendizado de determinados algoritmos, por exemplo, Perceptron Multicamadas (MLP).
34		Esse módulo pertence à categoria *Módulos dos Usuários*, pois foi implementado por outro pesquisador com o intuito de facilitar a utilização dos classificadores disponibilizados pelo site μ*Classify.com*. Com esse módulo os usuários podem realizar análise de sentimento em textos escritos em inglês.
35		Esse módulo foi desenvolvido por outro pesquisador e possibilita o resgate de publicações na rede social *Twitter*. Ele foi concebido junto com o módulo Textfy para facilitar a mineração de opiniões em publicações presentes nessa rede social.

Esse módulos oferecem suporte à resolução das seguintes classes de problemas: controle de processos por meio de sistemas de inferência *fuzzy* (*e.g.*, módulo *Fuzzy*), otimização combinatória com variáveis binárias (gGA, Evolution), problema da mochila binária mono-objetivo e multiobjetivo (AntSystem, NSGA-II), classificação (CLONALG, ID3, J48), regressão (Perceptron, Adaline) e agrupamento de dados (Cobweb), predição de séries temporais (MLP), aproximação de funções (RBF), construção de regras de associação (Apriori) e caixeiro viajante (MMAS).

4.3 Outras Funcionalidades

Além das funções básicas, como gestão dos sistemas de IC, controle de acesso, listagem e detalhamento das execuções, a Athena dispõe de outras funcionalidades interessantes para os usuários, por exemplo: submissão de novos módulos, execuções dos sistemas em *batch* (o usuário pode criar lotes de execução), execução assíncrona (o usuário solicita a execução e recebe uma notificação por e-mail após a conclusão), execução dos sistemas via API, dentre outras. Essas funcionalidades não estão descritas neste documento devido ao caráter puramente técnico dessas funções. Dessa forma, decidiu-se focar apenas nas funções com relevância acadêmica e que estão diretamente relacionadas às prescrições da Arquitetura Proposta.

4.4 Exemplos de Utilização

Nessa seção são apresentados três exemplos de utilização com o objetivo de ilustrar o funcionamento da Athena. O primeiro apresenta a resolução do problema de Seleção de Casos de Teste (SCT) utilizando a meta-heurística *Max-Min AntSystem*. O segundo discute a resolução do problema de estimação do tamanho das nanopartículas de prata utilizando o algoritmo Perceptron Multicamadas (MLP). Por fim, o terceiro retrata a resolução do problema de alocação de equipes utilizando uma abordagem híbrida que combina dois sistemas de inferência *fuzzy* e o algoritmo genético multiobjetivo NSGA-II.

O método para construção de sistemas de IC definido pela Athena possui cinco etapas (ver Seção 3.1.4). No entanto, para simplificar o detalhamento dos exemplos, a descrição da etapa de integração com outros sistemas foi omitida.

Vale ressaltar que todos os dados utilizados nesses exemplos estão disponíveis para *download* a partir do link http://goo.gl/k04954 e os resultados podem ser replicados no *website* da ferramenta http://athenasystems.com.br.

4.4.1 Seleção de Casos de Teste

O Teste de *Software* é a verificação dinâmica do comportamento de um programa, realizada geralmente com base em um conjunto de casos de teste (PRESSMAN, 2011). Essa é uma atividade cara e, quando um *software* é modificado, ele deve ser testado para verificar se as mudanças não têm efeitos indesejáveis nas funcionalidades (IEEE, 1990).

A reexecução de todos os casos de teste pode ser inviável, devido a limitações de tempo e de recursos (YOO; HARMAN, 2007). Muitas vezes é necessário selecionar um subconjunto de casos de teste a ser executado. Esse problema é difícil e o uso de técnicas convencionais da Engenharia de *Software* têm se mostrado pouco eficiente em sua solução (HARMAN et al., 2012).

Para o problema de seleção de casos de teste foi usado uma abordagem que utiliza a meta-heurística *Max-Min Ant System* (STÜTZLE; HOOS, 2000) para selecionar um subconjunto de testes que maximize a criticidade (nível de importância do teste), obedecendo a restrição de tempo imposta. O problema foi modelado como um problema da mochila binária (BAZGAN; HUGOT; VANDERPOOTEN, 2009), com a seguinte formulação matemática:

$$Maximizar : \sum_{k=1}^{t}(z_k) \times x_k \tag{4.1}$$

$$Restrição : \sum_{k=1}^{t}(w_k) \times x_k \leq KLC \tag{4.2}$$

Onde k é o índice do teste; z_k representa a criticidade (ganho) do teste k; $x_k = 0$ quanto o teste k não foi incluído na mochila; $x_k = 1$ quando o teste k foi incluído na mochila; w_k é o tempo de execução (peso) do teste k; e KLC representa a capacidade total da mochila (restrição de tempo).

A partir o entendimento do problema é possível utilizar a Athena para implementar a solução. Inicialmente, como descrito no método proposto (ver Seção 3.1.4), é necessário a acessar a ferramenta. Depois, deve-se definir as informações básicas (nome e descrição) a respeito do sistema que está sendo criado. Nesse exemplo, utilizou-se como nome "Seleção de Casos de Teste" e a descrição foi o relato do problema. A Figura 29 apresenta a tela de criação de sistemas (Figura 29a) e a tela de detalhes dos sistemas de IC (Figura 29b).

(a) Tela para criação dos sistemas.

(b) Tela que exibe os detalhes dos sistemas.

Figura 29 – Criação e detalhes de um novo sistema de IC.

Após a definição das informações básicas é possível criar uma arquitetura para o sistema. A arquitetura define os módulos selecionados e suas interconexões. Diversas arquiteturas podem ser criadas com a finalidade de comparar diferentes abordagens para resolver o mesmo problema. Nesse exemplo, definiu-se uma arquitetura com três módulos: ReadCSV, MMAS e WriteCSV. A Figura 30 apresenta o resultado final dessa arquitetura.

ReadCSV: nesse exemplo o módulo ReadCSV foi utilizado para realizar a leitura dos dados de entrada provenientes de um arquivo no formato CSV. Foram definidas uma entrada do tipo arquivo (File) e duas saídas do tipo Real. Dessa forma, o módulo realiza a conversão do arquivo CSV em duas listas de valores (criticidade e tempo).

MMAS: o módulo MMAS foi utilizado para solucionar o problema da mochila formulado pelas equações 4.1 e 4.2. Foram definidas duas entradas e duas saídas do tipo Real. O módulo utiliza a meta-heurística *Max-Min Ant System* para selecionar um subconjunto de testes que maximize a criticidade obedecendo a restrição de tempo.

WriteCSV: esse módulo foi utilizado para persistir os resultados obtidos no MMAS (*criticidadeSelecionada* e *tempoSelecionado*) em um arquivo no formato CSV. Foram definidas duas entradas do tipo Real e uma saída do tipo arquivo (File). Portanto, o módulo realiza a conversão das duas listas de valores em um arquivo CSV.

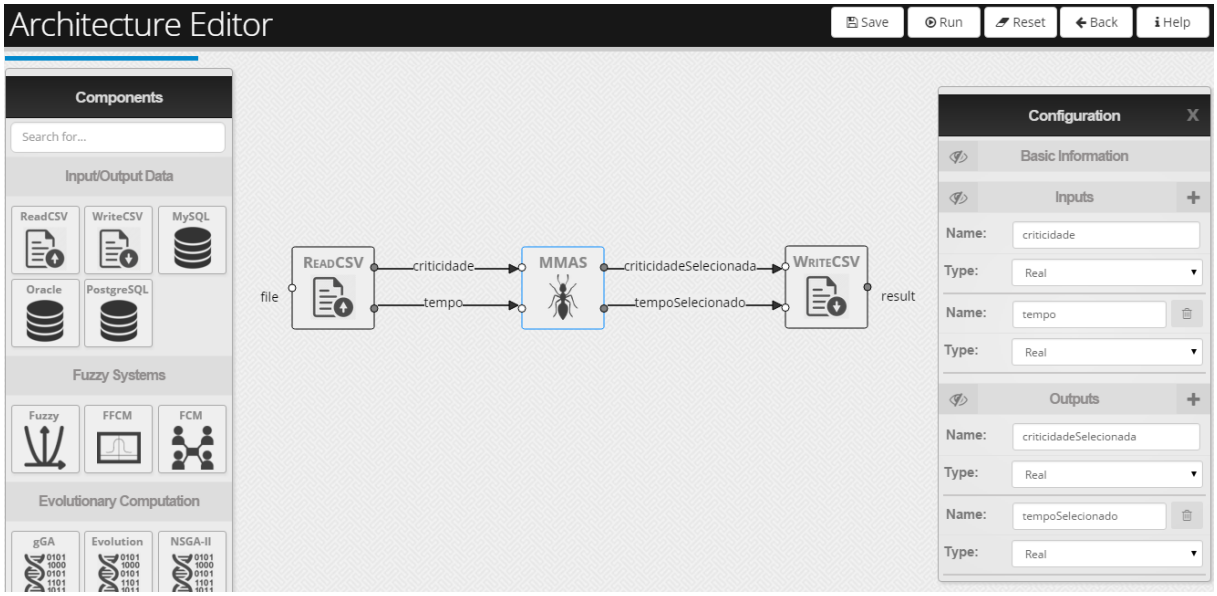


Figura 30 – Arquitetura para solucionar o problema de seleção de casos de teste utilizando a meta-heurística *Max-Min Ant System* (MMAS).

Para finalizar a construção da arquitetura, os módulos foram interconectados observando a compatibilidade de tipo, *i.e.*, saída do tipo Real com entrada do tipo Real. A Tabela 11 apresenta as conexões definidas na versão final da arquitetura (Figura 30).

Tabela 11 – Interconexões dos módulos da arquitetura proposta para o problema de Seleção de Casos de Teste.

#	**Origem**	**Destino**	**Saída**	**Entrada**
1	ReadCSV	MMAS	criticidade	criticidade
2	ReadCSV	MMAS	tempo	tempo
3	MMAS	WriteCSV	criticidadeSelecionada	criticidadeSelecionada
4	MMAS	WriteCSV-II	tempoSelecionado	tempoSelecionado

Após a definição da arquitetura é possível realizar a etapa de configuração dos módulos para posterior execução. A Figura 31 apresenta a tela na qual essa etapa é realizada. Na imagem é possível observar as informações básicas sobre o sistema (*Basic Information*), uma visualização da arquitetura (*Screenshot*) e os módulos que devem ser configurados.

ReadCSV: nesse exemplo utilizou-se como entrada um arquivo CSV com os dados apresentados na Tabela 12. A orientação de leitura foi definida com o valor "Coluna" e o arquivo possui cabeçalho.

MMAS: as entradas desse módulo são provenientes das saídas do ReadCSV. Os valores das configurações são apresentados na Tabela 13 e foram definidos de acordo com as sugestões propostas por Stützle e Hoos (STÜTZLE; HOOS, 2000).

WriteCSV: esse módulo não possui configurações e as suas entradas são provenientes das saídas do MMAS.

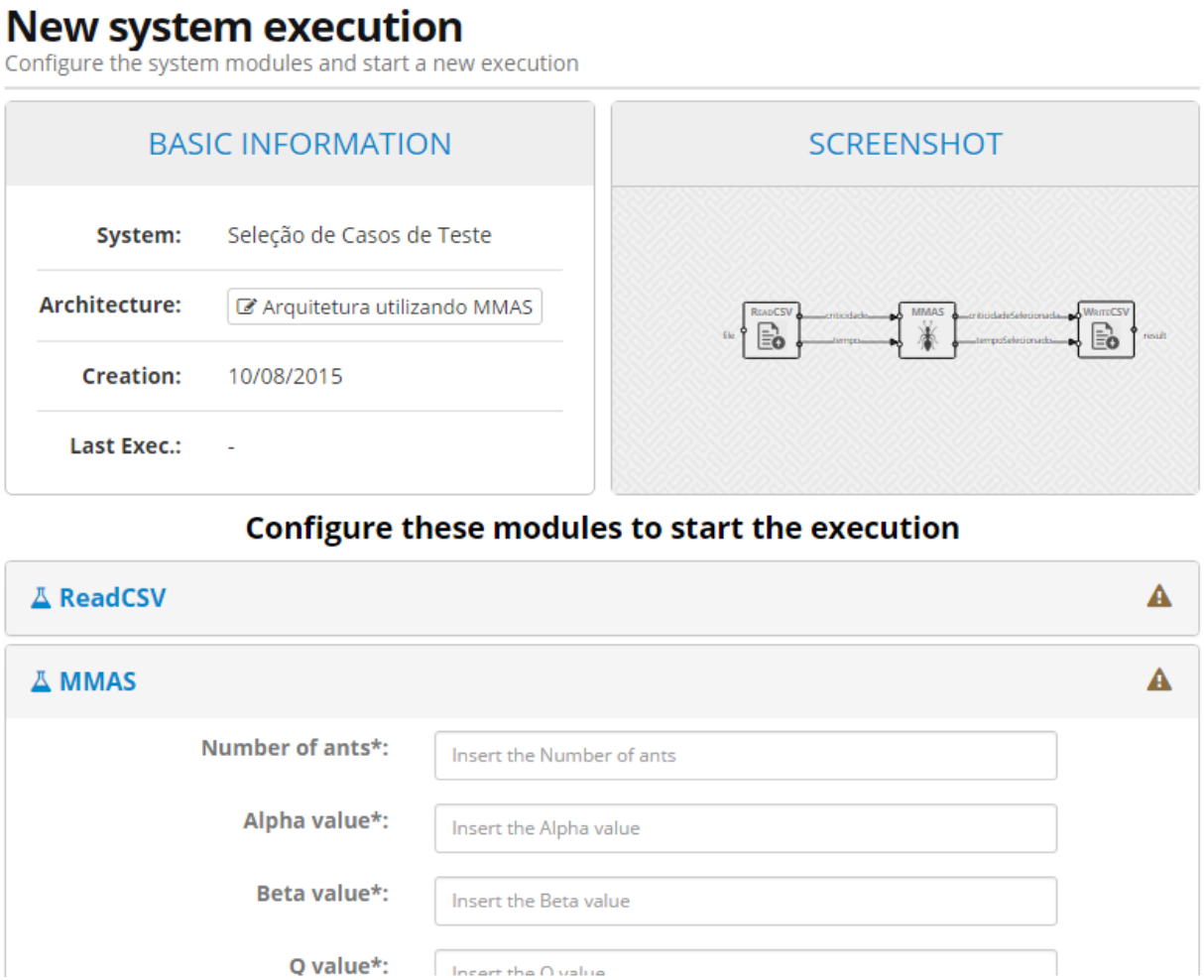


Figura 31 – Tela de configuração de execução da arquitetura proposta para solucionar o problema de seleção de casos de teste utilizando a meta-heurística MMAS.

Tabela 12 – Dados de entrada utilizados no módulo ReadCSV para o problema de Seleção de Casos de Teste.

#	**Criticidade**	**Tempo**
1	141,51	48,00
2	109,41	310,00
3	139,89	91,00
4	66,29	10,00
5	2,32	2,00
6	50,41	36,00
7	8,93	0,10
8	5,76	10,10
9	58,79	10,00
10	56,19	75,00
11	250,72	235,00
12	86,38	17,00
13	5,65	6,00
14	52,98	27,00
15	38,04	31,00

Tabela 13 – Configurações utilizadas para solucionar o problema de SCT.

#	**Módulo**	**Configuração**	**Valor**
1	ReadCSV	Orientação de Leitura	Coluna
2	ReadCSV	Cabeçalho	Sim
3	MMAS	Número de Formigas	10,00
4	MMAS	Valor *alpha*	1,00
5	MMAS	Valor *beta*	2,00
6	MMAS	Valor Q	1,00
7	MMAS	Taxa de Evaporação	0,50
8	MMAS	Valor máximo do feromônio	10,00
9	MMAS	Valor mínimo do feromônio	0,10
10	MMAS	Número máximo de iterações	10,00
11	MMAS	Estratégia de atualização do feromônio	AntCycleMax
12	MMAS	Função heurística	$(criticidade/tempo^2)$
13	MMAS	Função qualidade da solução	$sum(criticidade)$
14	MMAS	Função de restrição	$tempo \leq 150$
15	WriteCSV	Não possui configurações	-

Analisando a Tabela 13 é possível identificar a aplicação da gramática de expressões disponibilizada pela biblioteca JEP (ver Seção 4.1). As configurações 12, 13 e 14 foram definidas por meio de sentenças matemáticas e lógicas escritas em linguagem natural. Dessa forma, o usuário pode criar equações utilizando como variáveis as entradas do módulo (criticidade e tempo) e todos os operadores descritos na documentação da JEP[4].

Após a construção e configuração da arquitetura é possível acionar sua execução. Esse processo faz com que o *Front-end* envie uma requisição HTTP solicitando ao *Back-end* a execução do sistema, utilizando como insumo a arquitetura selecionada, as configurações e os dados de entrada. Ao final da operação, o *Back-end* devolve uma resposta HTTP com os resultados obtidos. Para esse exemplo, as soluções (testes selecionados) estão disponíveis na Tabela 14 e a Figura 32 apresenta um recorte da tela de exibição dos resultados.

Tabela 14 – Resultados obtidos para o problema de Seleção de Casos de Teste.

#	**Criticidade**	**Tempo**
1	141,51	48,00
4	66,29	10,00
5	2,32	2,00
7	8,93	0,10
8	5,76	10,10
9	58,79	10,00
12	86,38	17,00
13	5,65	6,00
14	52,98	27,00
Total	428,61	130,20

[4] JEP - documentação disponível em http://goo.gl/hFfPeU.

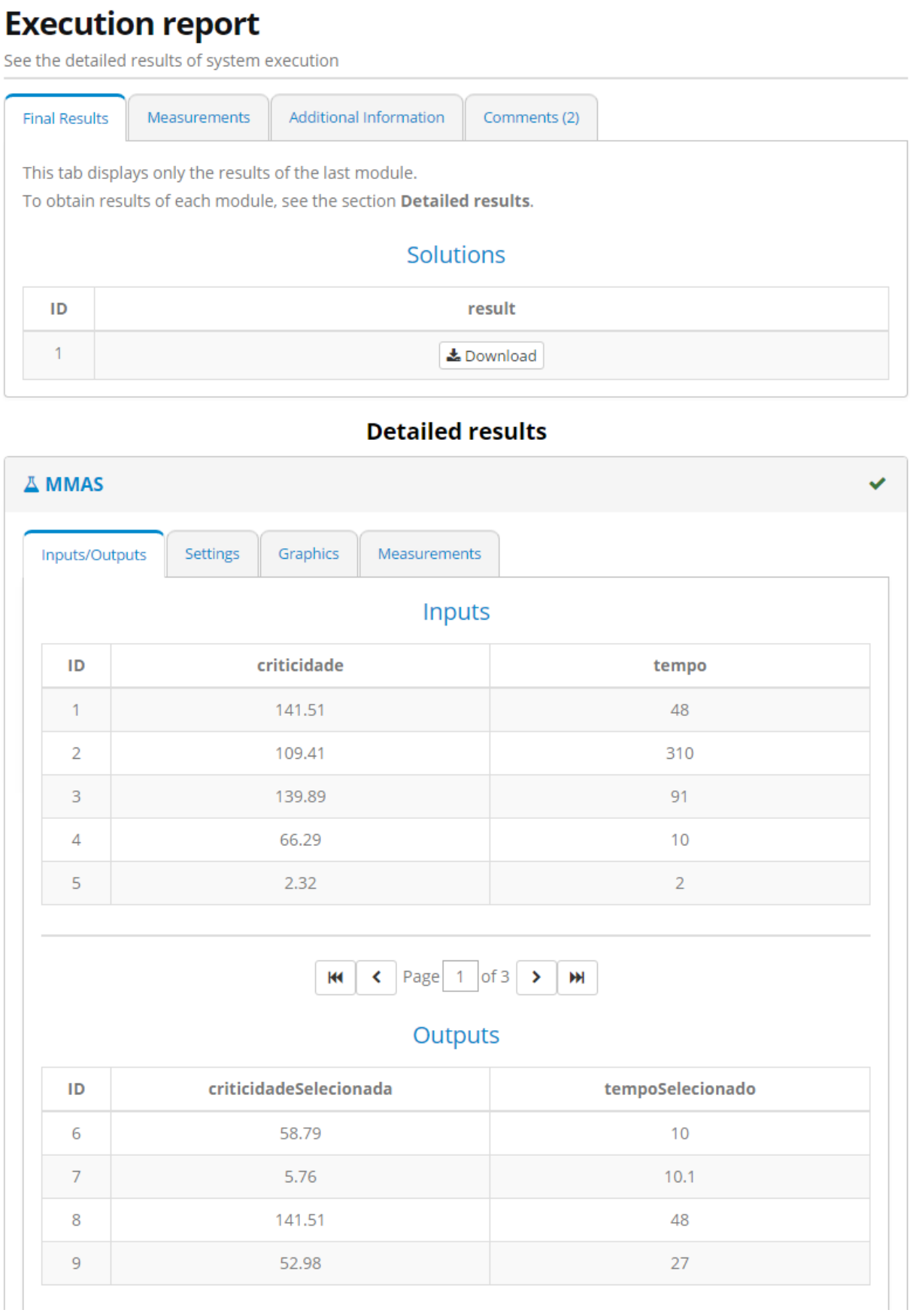


Figura 32 – Tela de exibição dos resultados da execução.

4.4.2 Estimação do Tamanho das Nanopartículas de Prata

Nanopartículas (NPs) são definidas como partículas com dimensões da ordem de 100 nm ou menos (EASON; NOBLE; SNEDDON, 1955). Dentre as NPs, merecem destaque as nanopartículas de prata AgNPs, por suas características físicas, químicas e biológicas que despertam interesse em uma ampla gama de aplicações (NAIR; LAURENCIN, 2007).

A preparação de AgNPs (Nanopartículas de Prata) pode ser considerada como um processo multivariável, pois uma série de fatores como a temperatura, o pH do meio reacional, o tempo de síntese, a concentração dos sais de prata, a concentração do agente redutor, bem como, a concentração do agente estabilizante, entre outros, desempenham papel determinante e tendem a afetar as características do produto final.

No trabalho proposto por Aragão (ARAGAO et al., 2015) e desenvolvido no campus Parnaíba da Universidade Federal do Piauí (UFPI), foi empregada uma RNA do tipo MLP com algoritmo de treinamento *backpropagation* para a predição do tamanho de AgNPs. Esse exemplo apresenta a implementação dessa abordagem utilizando a Athena.

A Figura 33 apresenta o esquema da MLP adotada para resolver a estimação do tamanho das nanopartículas de prata. De posse dessas definições, é possível utilizar a Athena para implementar um sistema de IC capaz de solucionar o problema. Conforme descrito no método proposto (ver seção 3.1.4), deve-se definir um nome e uma descrição para o sistema que está sendo criado. Nesse exemplo, utilizou-se como nome "Estimação do Tamanho das Nanopartículas de Prata" e a descrição foi o relato do problema.

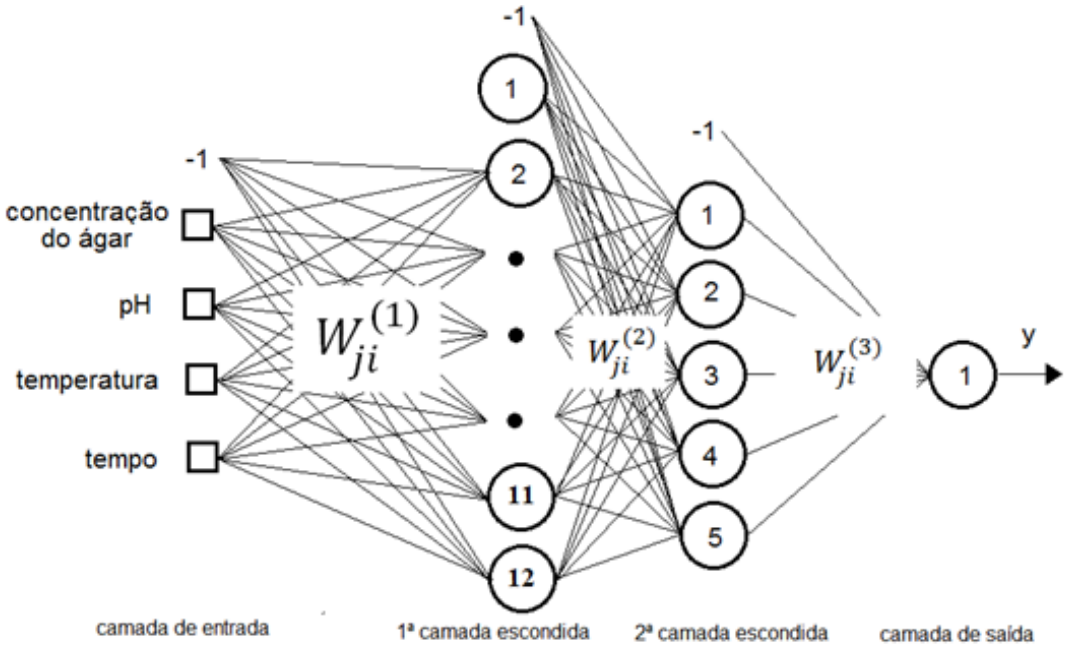


Figura 33 – Esquema da MLP adotada para resolver o problema de estimação do tamanho de Nanopartículas de Prata (AgNPs) (ARAGAO et al., 2015).

Após a definição das informações básicas, é possível criar uma arquitetura para o sistema. Nesse exemplo, definiu-se uma arquitetura com três módulos: ReadCSV, MLP e WriteCSV. A Figura 34 apresenta o resultado final dessa arquitetura.

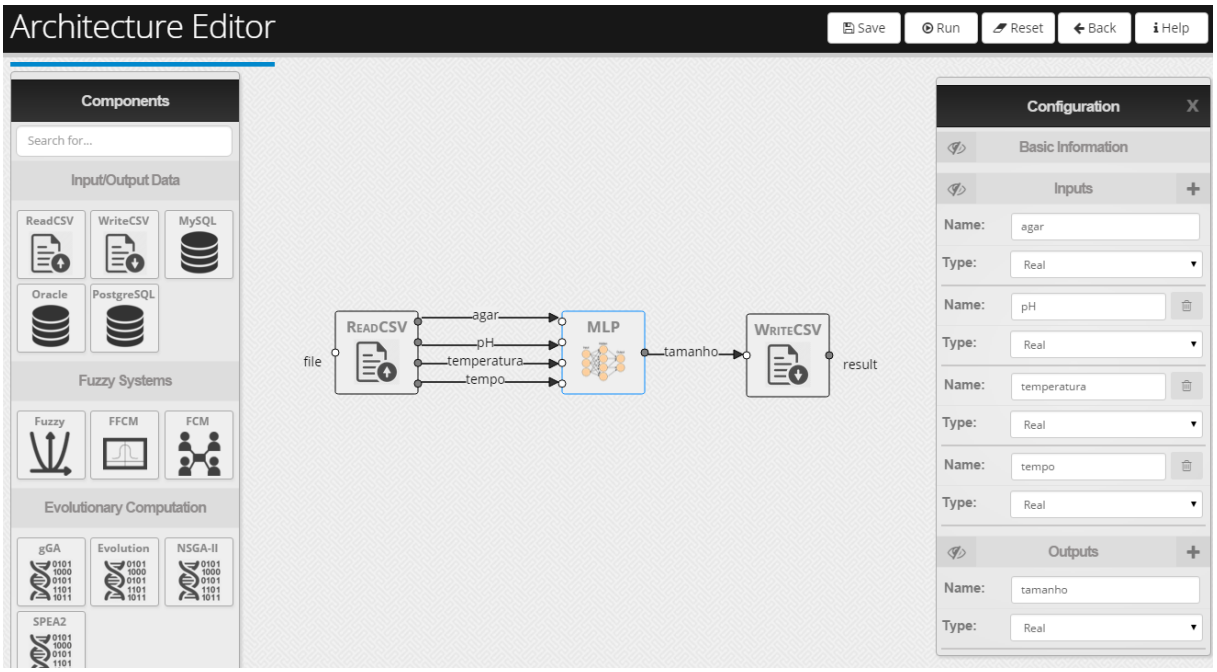


Figura 34 – Arquitetura para solucionar o problema de estimação do tamanho de Nanopartículas de Prata (AgNPs) utilizando a rede MLP.

ReadCSV: nesse exemplo o módulo ReadCSV foi utilizado para realizar a leitura dos dados de entrada provenientes de um arquivo CSV com as informações de concentração do ágar, pH do meio reacional, temperatura e tempo de reação. Foram definidas uma entrada do tipo arquivo (File) e quatro saídas do tipo Real.

MLP: o módulo MLP foi utilizado para predizer o tamanho da Nanopartícula de Prata. Foram definidas quatro entradas e uma saída, ambas do tipo Real.

WriteCSV: nesse exemplo o módulo WriteCSV foi utilizado para persistir os resultados obtidos a partir da rede MLP em um arquivo no formato CSV. Foram definidas uma entrada do tipo Real e uma saída do tipo arquivo (File).

A Tabela 15 apresenta as interconexões definidas na versão final da arquitetura.

Tabela 15 – Interconexões dos módulos para solucionar o problema de estimação do tamanho de Nanopartículas de Prata (AgNPs).

#	**Origem**	**Destino**	**Saída**	**Entrada**
1	ReadCSV	MLP	agar	agar
2	ReadCSV	MLP	pH	pH
3	ReadCSV	MLP	temperatura	temperatura
4	ReadCSV	MLP	tempo	tempo
5	MLP	WriteCSV	tamanho	tamanho

Depois da definição da arquitetura é possível realizar a etapa de configuração. A Tabela 16 apresenta os valores das configurações de cada módulo. Para o caso do MLP, as configurações foram extraídas dos experimentos realizados em ARAGAO et al. (2015). O módulo MLP utiliza o conjunto de treinamento para ajustar os pesos da RNA, tornando possível a avaliação as entradas. Entretanto, para evitar a repetição do treinamento a cada nova execução, o usuário pode fazer o *download* do modelo treinado para utilizá-lo em execuções posteriores, tornando desnecessária a realização de um novo treinamento.

Tabela 16 – Configurações utilizadas no problema de estimação do tamanho AgNPs.

#	**Módulo**	**Configuração**	**Valor**
1	ReadCSV	Orientação de Leitura	Coluna
2	ReadCSV	Cabeçalho	Não
3	MLP	Conjunto de Treinamento	Arquivo CSV
4	MLP	Arquitetura da Rede Neural	4-12-5-1
5	MLP	Taxa de Aprendizado	0,01
6	MLP	Função de Ativação	Sigmoide
7	MLP	Precisão Requerida	0,000001
8	MLP	Número Máximo de Iterações	10000
9	WriteCSV	Não possui configurações	-

Após a construção e configuração da arquitetura, é possível acionar sua execução. A Tabela 17 apresenta os dados de entrada (ágar, pH, temperatura e tempo) e o resultado (tamanho da AgNP) produzido pelo módulo MLP. Vale ressaltar que esse módulo normaliza os dados de entrada e de treinamento no intuito de otimizar seu funcionamento.

Tabela 17 – Dados de entrada e resultados obtidos para o problema de estimação do tamanho de Nanopartículas de Prata (AgNPs).

#	**Ágar**	**pH**	**Temperatura**	**Tempo**	**Tamanho**
1	0,030	8	70	10	69,8768
2	0,030	8	70	30	39,8025
3	0,030	10	70	10	15,6059
4	0,030	11	70	30	1,3642
5	0,010	9	70	10	23,6262
6	0,030	11	90	30	1,1118
7	0,005	10	70	30	2,5636
8	0,010	11	70	30	1,0350
9	0,005	8	70	10	54,0652
10	0,020	10	70	20	4,9170
11	0,010	8	90	30	23,7136
12	0,020	10	90	20	3,5901
13	0,010	11	90	10	1,6338
14	0,005	9	90	20	8,8397
15	0,030	9	90	10	33,8647
16	0,005	11	90	20	1,4431
17	0,010	9	90	30	6,2806
18	0,020	9	90	20	12,5954
19	0,005	11	70	30	1,0277
20	0,020	8	70	20	48,0865

4.4.3 Alocação de Equipes de Desenvolvimento

Diferente dos outros dois exemplos apresentados, esse exibe uma característica interessante e ao mesmo tempo desafiadora: a combinação de duas técnicas diferentes para solucionar o problema. Esse exemplo discute como a Athena lida com Sistemas Híbridos. Para essa discussão escolheu-se um problema tradicional da Engenharia de *Software*: a alocação de equipes de desenvolvimento (HARMAN, 2007).

A alocação equipes de desenvolvimento é essencial para o processo ágil de desenvolvimento de software, entretanto esse problema é NP-difícil, uma vez que compreende a atribuição de equipes multifuncionais e auto organizadas (HARMAN et al., 2012). Muitos pesquisadores têm direcionado esforços para aplicar técnicas de Inteligência Computacional para resolver este problema. Em (BRITTO et al., 2012) é proposta uma abordagem híbrida baseada na meta-heurística multiobjetivo NSGA-II e em sistemas de inferência *fuzzy* para resolver esse problema. Essa abordagem - representada pela Figura 35 - pode ser sumarizada em três etapas sem prejuízo ao seu entendimento. São elas:

- **1ª etapa - Estimação da Produtividade**: nessa etapa um Sistema de Inferência *Fuzzy* é utilizado para estimar a produtividade dos desenvolvedores com base nas notas de conhecimento, habilidade e atitude de cada desenvolvedor, obtidas por meio de questionários aplicados na empresa;

- **2ª etapa - Geração das equipes**: nessa etapa são geradas as equipes buscando maximizar a produtividade total da equipe e minimizar o seu custo com salários. A produtividade da equipe é obtida somando a produtividade dos desenvolvedores selecionados e o custo é obtido somando o salário dos desenvolvedores pertencentes àquela equipe. Além disso, deve-se obedecer a restrição do tamanho das equipes;

- **3ª etapa - Estimação da qualidade das equipes**: nessa etapa outro Sistema de Inferência *Fuzzy* é utilizado para estimar a qualidade das equipes geradas na etapa anterior, uma vez que serão equipes com maior qualidade e também maior custo. Esse Sistema *Fuzzy* fará essa ponderação para tentar sugerir as melhores equipes, balanceando custo e produtividade. Esse módulo possui como entrada a produtividade e custo da equipe.

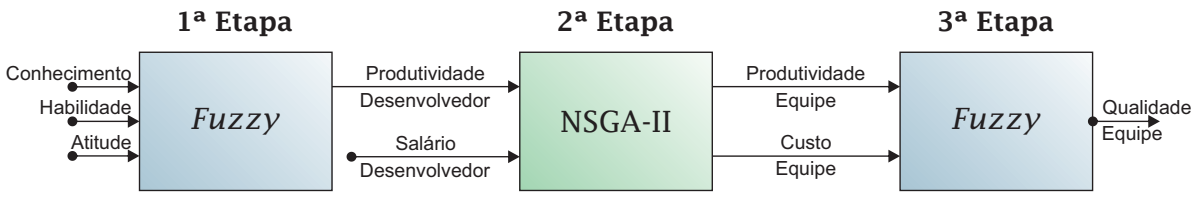


Figura 35 – Abordagem proposta para resolver o problema de alocação de equipes.

A partir do entendimento do problema é possível utilizar a Athena para auxiliar o desenvolvimento de uma solução baseada em Inteligência Computacional. Nesse exemplo, criou-se um sistema de IC com o nome "Alocação de Equipes" e a descrição, assim como nos demais exemplos, abriga o relato do problema.

Após a definição das informações básicas é possível criar uma arquitetura para o sistema. Nesse exemplo, utilizou-se uma arquitetura com cinco módulos: ReadCSV, *Fuzzy* (I), NSGA-II, *Fuzzy* (II) e WriteCSV. É importante enfatizar que esses módulos foram escolhidos considerando a abordagem descrita em (BRITTO et al., 2012). A Figura 36 apresenta a versão final dessa arquitetura.

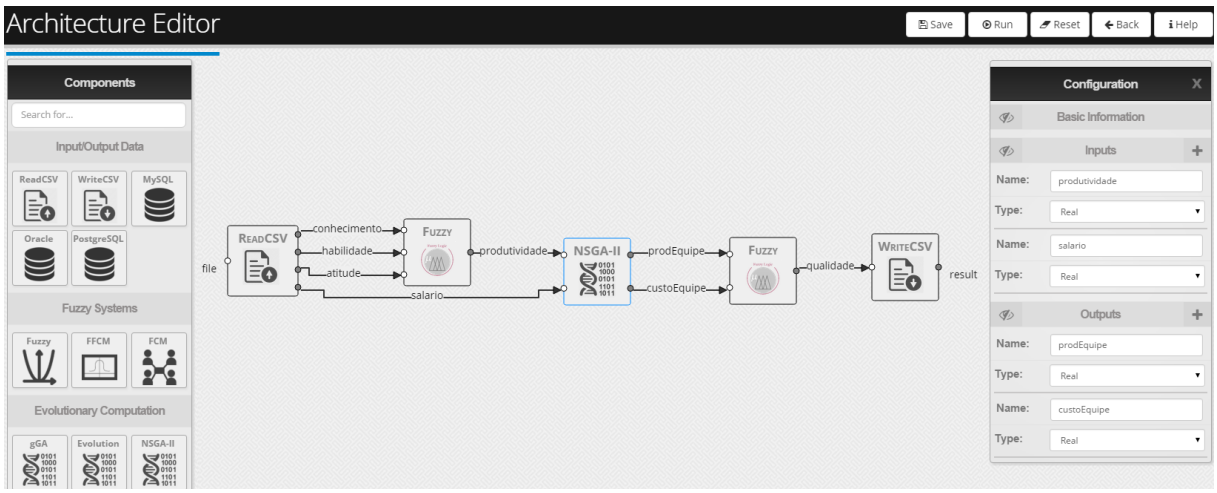


Figura 36 – Arquitetura para solucionar o problema de alocação de equipes.

ReadCSV: a primeira etapa da abordagem é a estimação da produtividade dos desenvolvedores com base nas notas de conhecimento, habilidade e atitude. Essas informações, unidas com o salário de cada desenvolvedor foram sumarizadas em um arquivo CSV. Dessa forma, o módulo ReadCSV será encarregado de injetar esses dados no sistema. Para isso foram definidas uma entrada do tipo arquivo (File) e quatro saídas do tipo Real.

Fuzzy (I): a estimação da produtividade é realizada utilizando um Sistema de Inferência *Fuzzy* de Mamdani que possui três entradas (conhecimento, habilidade e atitude) e uma variável de saída (produtividade). Dessa forma, foi selecionado um módulo *Fuzzy* com três entradas e uma saída, todas do tipo Real. Esse módulo utiliza os conjuntos *fuzzy* e a base de regras especificada no arquivo de configuração FCL para avaliar as entradas e inferir as saídas.

NSGA-II: a segunda etapa da abordagem é geração das equipes utilizando a meta-heurística NSGA-II. O objetivo dessa etapa é encontrar equipes com baixo custo (medido pelo salário dos desenvolvedores) e alta produtividade (medida pela produtividade dos desenvolvedores). Para isso foi selecionado o módulo NSGA-II com duas entradas (produtividade e custo) e duas saídas (prodEquipe e custoEquipe), todas do tipo Real.

Fuzzy (II): a terceira e última etapa refere-se à estimação da qualidade das equipes. Esse processo é realizado com um Sistema de Inferência *Fuzzy* ajustado de acordo com o arquivo de configuração FCL. Dessa forma, foi selecionado outro módulo *Fuzzy* com duas entradas (prodEquipe e custoEquipe) e uma saída (qualidade), todas do tipo Real.

WriteCSV: finalmente, utilizou-se o módulo WriteCSV para realizar a persistência dos resultados obtidos. Para isso, foram definidas uma entrada do tipo Real (qualidade) e uma saída do tipo (File).

A Tabela 18 apresenta as conexões definidas para essa arquitetura. Observe que as colunas Saída e Entrada possuem valores iguais. Esse padrão foi adotado apenas para facilitar o entendimento dos exemplos, entretanto esses nomes podem ser diferentes.

Tabela 18 – Configurações utilizadas para solucionar o problema de alocação de equipes.

#	**Origem**	**Destino**	**Saída**	**Entrada**
1	ReadCSV	*Fuzzy* (I)	conhecimento	conhecimento
2	ReadCSV	*Fuzzy* (I)	habilidade	habilidade
3	ReadCSV	*Fuzzy* (I)	atitude	atitude
4	ReadCSV	NSGA-II	salario	salario
5	*Fuzzy* (I)	NSGA-II	produtividade	produtividade
6	NSGA-II	*Fuzzy* (II)	prodEquipe	prodEquipe
7	NSGA-II	*Fuzzy* (II)	custoEquipe	custoEquipe
8	*Fuzzy* (II)	WriteCSV	qualidade	qualidade

Depois da definição da arquitetura é possível realizar a etapa de configuração dos módulos. A Tabela 19 apresenta os valores das configurações de cada módulo. Esses valores foram extraídos a partir dos experimentos realizados em BRITTO et al. (2012).

Tabela 19 – Configurações utilizadas para solucionar o problema de alocação de equipes.

#	**Módulo**	**Configuração**	**Valor**
1	ReadCSV	Orientação de Leitura	Coluna
2	ReadCSV	Cabeçalho	Sim
3	*Fuzzy* (I)	FCL	Arquivo FCL (Tabela 20)
4	NSGA-II	Tamanho da população	100
5	NSGA-II	Número máximo de iterações	25.000
6	NSGA-II	Operador de Seleção	*BinaryTournament2*
7	NSGA-II	Operador de *Crossover*	*SinglePointCrossover*
8	NSGA-II	Operador de Mutação	*BitFlipMutation*
9	NSGA-II	Probabilidade de *Crossover*	90%
10	NSGA-II	Probabilidade de Mutação	5%
11	NSGA-II	Função de Maximização	$produtividade - (10 \times abs(3 - solution\_size))$
12	NSGA-II	Função de Minimização	$salario + (1000 \times abs(3 - solution\_size))$
13	*Fuzzy* (II)	FCL	Arquivo FCL (Tabela 21)
14	WriteCSV	Não possui configurações	-

Analisando a Tabela 19 é possível identificar novamente a aplicação da gramática de expressões disponibilizada pela biblioteca JEP. As configurações 11 e 12 foram definidas por meio de sentenças matemáticas, utilizando as variáveis de entrada e a função módulo

($abs()$). O termo *solution_size* é uma variável especifica do módulo NSGA-II e seu valor representa o tamanho da solução encontrada. Esse valor é obtido por meio da quantidade de 1's presentes no vetor solução, pois atualmente o NSGA-II só trabalha com problemas de otimização nos quais as variáveis são binárias. Dessa forma, o trecho $abs(3 - solution\_size)$) funciona como uma restrição que penaliza soluções com tamanho diferente de 3. Isso fará com que a meta-heurística busque formar equipes com no máximo três integrantes.

As tabelas 20 e 21 apresentam, respectivamente, as bases de regras utilizadas para estimar a produtividade dos desenvolvedores e para inferir a qualidade das equipes.

Tabela 20 – Base de regras utilizada para inferir a produtividade dos desenvolvedores.

	Conhecimento / Atitude								
Habilidade	B/B	B/M	B/A	M/B	M/M	M/A	A/B	A/M	A/A
B	MB	B	B	B	B	B	B	M	M
M	B	M	M	M	M	M	A	A	MA
A	B	M	A	A	A	A	MA	MA	MA

Nota: MB - Muito Baixo; B - Baixo; M - Médio; A - Alto; MA - Muito Alto.

Tabela 21 – Base de regras utilizada para estimar a qualidade das equipes.

	Custo		
Produtividade	Baixo	Médio	Alto
Baixo	Baixo	Baixo	Muito Baixo
Médio	Alto	Médio	Baixo
Alto	Muito Alto	Alto	Médio

A Tabela 22 apresenta os dados de entrada obtidos pelo módulo ReadCSV e os resultados da etapa de inferência da produtividade e a Tabela 23 exibe as equipes formadas com seus respectivos valores de produtividade, custo e qualidade.

Tabela 22 – Dados de entrada e resultados dos processo de inferência da produtividade.

Dev	**Conhecimento**	**Habilidade**	**Atitude**	**Salário**	**Produtividade**
1	8,95	6,88	7,83	6750,00	9,09
2	8,36	7,21	8,13	5000,00	9,34
3	7,00	6,50	8,13	3750,00	7,50
4	7,78	5,96	5,50	5000,00	7,15
5	7,68	6,43	5,50	5000,00	6,91
6	5,76	7,21	7,50	3750,00	6,25
7	7,14	5,23	6,67	5000,00	5,20
8	7,14	5,83	7,14	3750,00	5,29
9	8,13	7,50	8,44	5750,00	9,31
10	7,58	6,43	6,14	5000,00	6,59
11	7,94	8,50	8,44	5000,00	9,28
12	5,63	5,68	6,73	3750,00	5,00
13	6,39	5,42	7,83	3750,00	7,24
14	6,15	5,83	7,12	3000,00	5,17
15	7,94	8,03	9,17	5000,00	9,28

Tabela 23 – Equipes formadas pelo algoritmo NSGA-II e avaliados pelo *Fuzzy*.

#	Tamanho	Desenvolvedores	Produtividade	Custo	Qualidade
1	3	2,9,11	27,93	15.750	7,50
2	3	2,11,15	27,90	15.000	7,50
3	3	2,3,9	26,15	14.500	7,50
4	3	2,3,15	26,12	13.750	7,50
5	3	2,3,13	24,08	12.500	7,50

Por fim, a Figura 37 exibe um dos gráficos gerados pelo módulo *Fuzzy* (I) apresentando a função de pertinência da variável conhecimento, e a Figura 38 exibe o gráfico com as soluções encontradas pelo módulo NSGA-II.

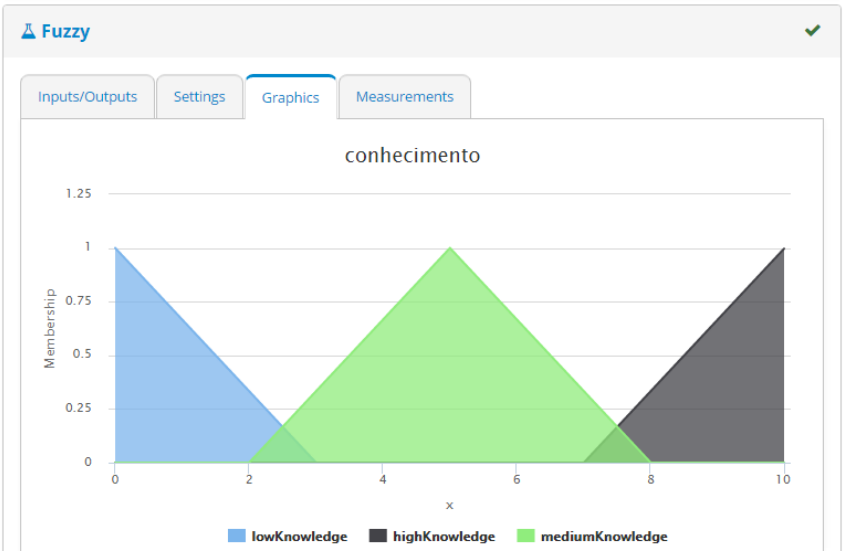


Figura 37 – Gráfico gerado pelo módulo *Fuzzy* (I) com a função de pertinência da variável *conhecimento*.

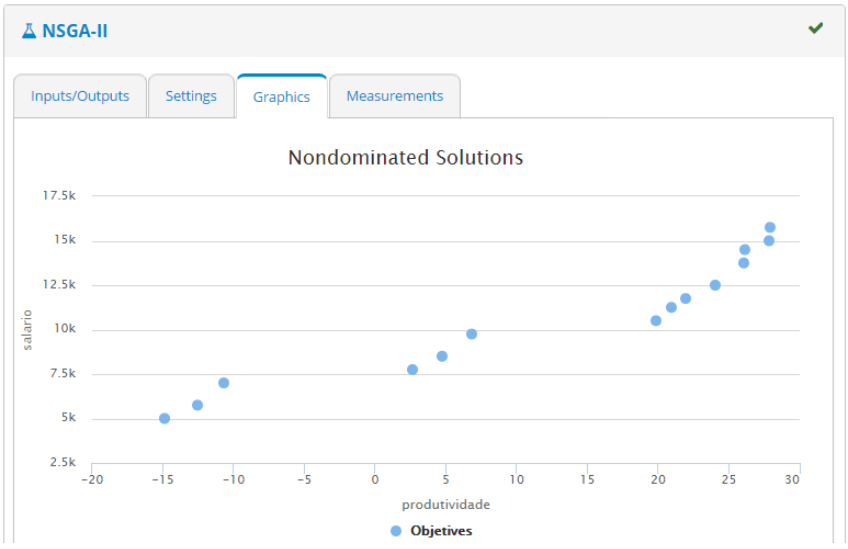


Figura 38 – Gráfico gerado pelo módulo NSGA-II com as soluções encontradas.

Parte IV

Resultados e Discussões

5 Estudo Experimental

A construção de um conhecimento válido a respeito de determinado problema deve ser realizada por meio de um método científico. Em geral, as etapas de investigação envolvem a observação cuidadosa do problema, a formulação das questões de pesquisa e das hipóteses a serem testadas, a realização de experimentos, o estabelecimento de relações por meio de análises estatísticas e, por fim, a formulação das conclusões. Esse método auxilia o pesquisador no entendimento da solução proposta, suas adequações e limitações (LAKATOS; MARCONI, 1991).

A investigação das dificuldades no desenvolvimento de Sistemas Baseados em Inteligência Computacional - problema abordado neste trabalho - possui um fator preponderante: o desenvolvimento de *software* depende do ser humano. Entretanto, nenhum dos trabalhos analisados no Capítulo 2 apresentou um estudo empírico que considerasse o fator humano para avaliar a ferramenta proposta. Isso motivou a realização de uma avaliação experimental seguindo as diretrizes da Engenharia de *Software* Experimental (WOHLIN et al., 2012). Essa avaliação é descrita nas seções seguintes.

5.1 Definição dos Objetivos

O propósito desse estudo experimental é analisar o desenvolvimento de sistemas de Inteligência Computacional com e sem a utilização da Athena, com o intuito de avaliar o impacto dessas duas técnicas, com respeito a eficiência (esforço), a eficácia e a percepção de qualidade (satisfação dos participantes), do ponto de vista do pesquisador, no contexto de alunos de graduação e pós-graduação em Ciência da Computação desenvolvendo Sistemas Inteligentes para resolver problemas reais.

5.2 Questões e Métricas

O estudo foi planejado com o objetivo de responder às seguintes questões:

- **QP1**: o esforço necessário para o desenvolvimento de sistemas de IC com a utilização da ferramenta Athena é menor que o esforço exigido para desenvolver os mesmos sistemas de Inteligência Computacional sem a utilização Athena? Utilizou-se a medida de tempo em segundos como métrica para avaliar esforço aplicado no desenvolvimento de sistemas de IC. Quanto maior o tempo, maior o esforço aplicado no desenvolvimento;

- **QP2**: a eficácia dos sistemas de IC desenvolvidos com a Athena é maior que a eficácia dos sistemas de IC desenvolvidos sem o apoio da Athena? Utilizou-se como métrica para avaliar a eficácia o inverso da quantidade de erros de programação/configuração encontrados nesses sistemas ($1/(quantidade\_de\_erros + 1)$). Entretanto, os erros não foram classificados de acordo com sua criticidade. Essa métrica será aperfeiçoada em trabalho futuros;
- **QP3**: a percepção de qualidade dos usuários quanto ao desenvolvimento de Sistemas de IC com o auxilio da Athena é maior que sistemas de IC desenvolvidos sem o suporte dessa ferramenta? Devido a natureza subjetiva dessa questão, foram utilizados questionários para avaliar a satisfação dos voluntários.

5.3 Hipóteses

As hipóteses deste estudo experimental são:

- Hipótese nula, H_0 *esforço*: Não há diferença no esforço, medido em termos de tempo em segundos, para criar sistemas de Inteligência Computacional com e sem o uso da Athena. H_0 *esforço*: Tempo (Athena) = Tempo (Sem suporte da Athena). Hipótese alternativa, H_1 *esforço*: Tempo (Athena) $\neq$ Tempo (Sem suporte da Athena);
- Hipótese nula, H_0 *eficácia*: Não há diferença na eficácia dos sistemas de IC desenvolvidos com e sem o uso da Athena. H_0 *eficácia*: Eficácia (Athena) = Eficácia (Sem suporte da Athena). Hipótese alternativa, H_1 *eficácia*: Eficácia (Athena) $\neq$ Eficácia (Sem suporte da Athena);
- Hipótese nula, H_0 *qualidade*: A qualidade dos sistemas de IC criados utilizando a Athena, medida em termos da percepção de qualidade dos usuários especialistas, é igual à percepção de qualidade de sistemas criados sem o suporte da ferramenta. H_0 *qualidade*: Qualidade (Athena) = Qualidade (Sem suporte da Athena). Hipótese alternativa, H_1 *qualidade*: Qualidade (Athena) $\neq$ Qualidade (Sem suporte da Athena).

5.4 Seleção de Variáveis

Variáveis Independentes são aquelas que podem ser controladas e/ou modificadas durante o experimento. Essas variáveis possuem efeitos sobre as variáveis dependentes. Neste estudo, as ferramentas (Athena e bibliotecas de código) utilizadas no desenvolvimento dos SBIC representam as variáveis independentes.

Variáveis Dependentes são as responsáveis por mensurar o efeito dos tratamentos utilizados no experimento. As variáveis consideradas são o tempo em segundos para construção dos SBIC, a eficácia desses sistemas e a percepção de qualidade dos participantes.

5.5 Objetos do Estudo

Os objetos do estudo são mecanismos utilizados para verificar o relacionamento causa-efeito entre o processo em análise (desenvolvimento de SBIC) e os tratamentos (implementação com e sem a Athena). Durante a execução do experimento os tratamentos são aplicados ao conjunto de objetos para a obtenção dos resultados (TRAVASSOS; GUROV; AMARAL, 2002). Nesta avaliação experimental, considerou-se como objetos dois trabalhos que apresentam Sistemas Inteligentes para solucionar problemas reais. São eles:

- **Estimação do Tamanho de AgNPs** (ARAGAO et al., 2015): esse trabalho emprega uma Rede Neural Artificial do tipo MLP para realizar o mapeamento do processo de síntese de Nanopartículas de Prata (AgNPs);

- **Alocação de Equipes de Desenvolvimento** (BRITTO et al., 2012): propõe uma abordagem híbrida baseada no algoritmo NSGA-II e em Sistemas de Inferência *Fuzzy* para resolver o problema da alocação de equipes de desenvolvimento em projetos ágeis. O problema consiste na seleção dos desenvolvedores que irão compor a equipe, buscando maximizar a produtividade e minimizar os custos.

Os dois objetos foram implementados pelo pesquisador de forma tradicional, utilizando as bibliotecas de código JFuzzyLogic, Neuroph e JMetal, e utilizando a Athena para servir de referência durante a análise das implementações dos participantes. As soluções obtidas foram verificadas utilizando como base os resultados apresentados em BRITTO et al. (2012) e ARAGAO et al. (2015).

5.6 Seleção dos Participantes

A seleção dos participantes é uma etapa importante para o experimento, pois está diretamente ligada à capacidade de generalização dos resultados obtidos. O cenário ideal para um experimento é a seleção aleatória de participantes, caracterizando uma amostra representativa da população. No entanto, por conta das dificuldades inerentes a realização do experimento, este é considerado um *quasi-experiment*, pois os participantes foram escolhidos de acordo com a conveniência (WOHLIN et al., 2012).

Os participantes escolhidos para a realização desse estudo são alunos da disciplina de Tópicos em Inteligência Artificial do curso de Ciência da Computação (DC/UFPI), ofertada no segundo período de 2015. Essa disciplina é optativa, portanto não há pre-requisitos formais para sua realização. Entretanto, por ser ofertada tanto para a graduação como para a pós-graduação, os professores recomendam que o aluno já tenha cursado as disciplinas de Programação Orientada à Objetos (POO), Inteligência Artificial e Engenharia de *Software*.

Antes da execução, os participantes foram devidamente caracterizados para que fosse possível analisar os resultados de acordo com os diferentes níveis de conhecimento em programação e nas técnicas de Inteligência Computacional. Além disso, essa caracterização pode auxiliar na compreensão dos *outliers*, ou seja, valores atípicos e que apresentam grande afastamento das demais observações. O Apêndice C apresenta o questionário de caracterização dos participantes.

O estudo iniciou com 18 participantes, no entanto, apenas 15 concluíram todas as etapas. A maioria dos alunos (86,7%) já tinha cursado as três disciplinas recomendadas (POO, IA e Engenharia de *Software*) e o restante (13,3%) tinha cursado apenas POO. Dentre os 15 participantes que concluíram o estudo, 47% são alunos de pós-graduação, 20% tinham experiência profissional trabalhando com codificação em Java, 27% afirmaram ter conhecimento básico nas bibliotecas JFuzzyLogic e JMetal, entretanto apenas dois dos participantes (13%) tinham experiência com a Neuroph. Essa tipificação dos voluntários motivou a realização de treinamentos nas técnicas e ferramentas de IC.

A Figura 39 apresenta a caracterização dos 15 participantes quanto aos conhecimentos em programação, Inteligência Computacional e ferramentas/*frameworks* de IC.

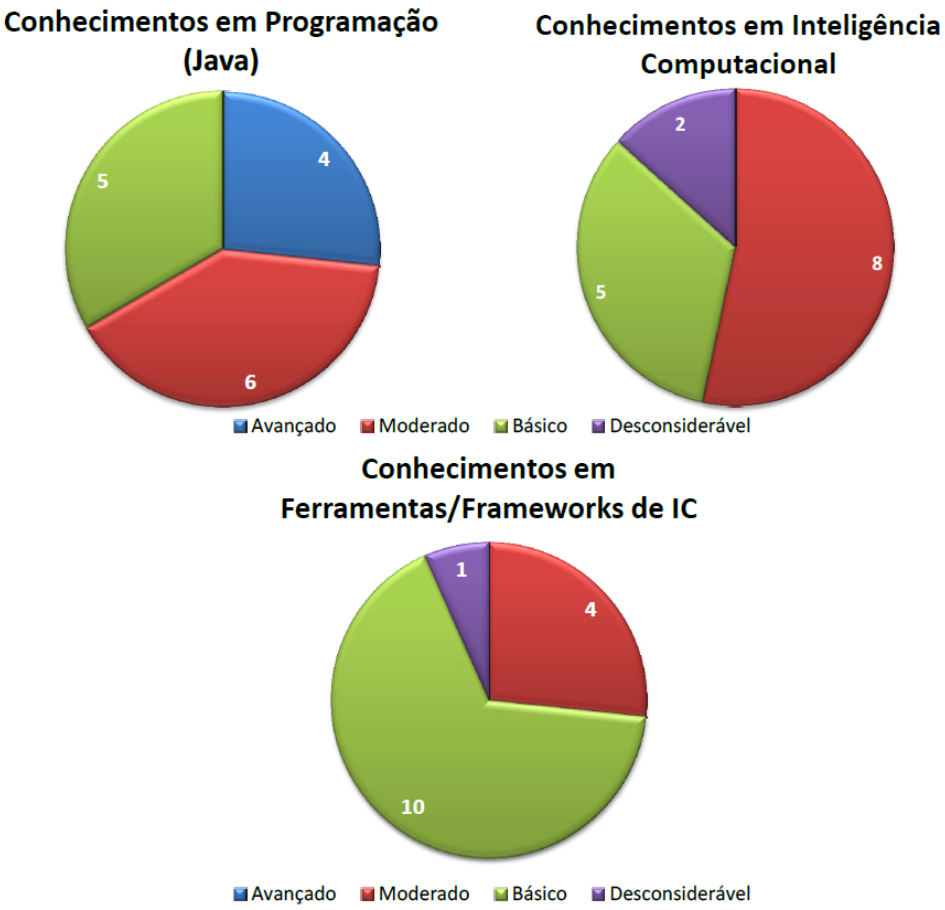


Figura 39 – Caracterização dos participantes quanto aos conhecimentos primordiais para a realização do estudo experimental.

5.7 Contexto e Instrumentação

O contexto do experimento é composto das condições em que o estudo é executado (TRAVASSOS; GUROV; AMARAL, 2002). Como essa avaliação foi realizada em ambiente laboratorial, o contexto é *in-vitro*, contando com a participação de alunos desenvolvendo sistemas para solucionar problemas reais.

Os instrumentos de um experimento são divididos em três tipos: objetos, instruções e manuais, e instrumentos de medição. Os objetos, instruções e manuais foram disponibilizados aos participantes de forma digital e impressa. Quanto aos instrumentos de medição, utilizou-se uma planilha eletrônica para coletar os dados de esforço no caso da implementação manual e a ferramenta UseSkill (SOUZA et al., 2015) no caso do desenvolvimento utilizando a Athena.

A UseSkill (Figura 40) é uma ferramenta utilizada para auxiliar a realização de testes de usabilidade remotamente em sistemas Web de forma não intrusiva (SOUZA et al., 2015). Ela é incorporada no ambiente de teste por meio de um *plugin* do Chrome e captura todas as ações que os usuários realizam nos sistemas Web. Com isso, foi possível capturar todas as ações que os participantes realizaram ao utilizar a Athena. Essas informações foram fundamentais, não só para o experimento, mas para avaliar a usabilidade da Athena. A utilização da UseSkill é melhor detalhada nas seções seguintes.

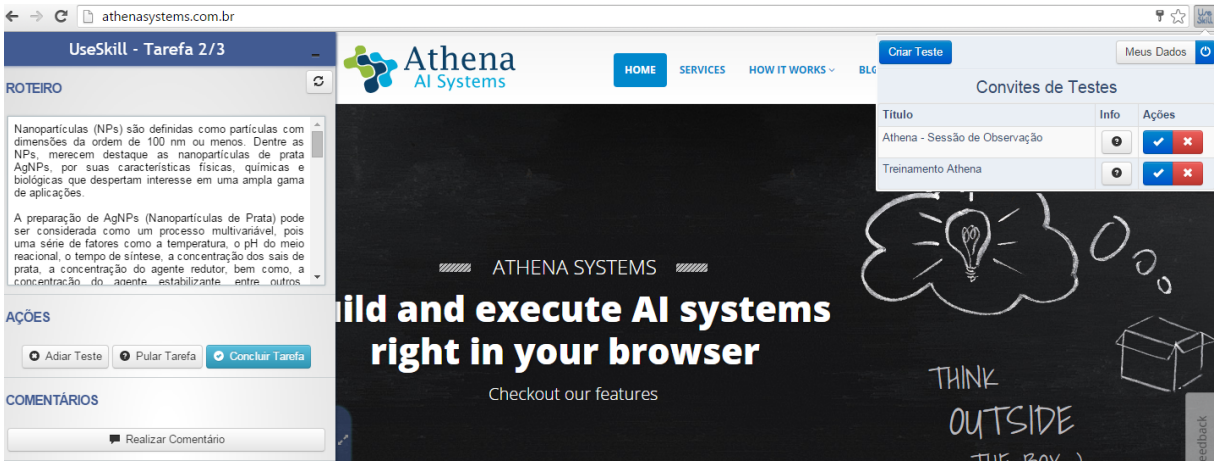


Figura 40 – *Plugin* da UseSkill para o navegador Chrome em ação.

5.8 Desenho Experimental

O desenho experimental define o modo como são realizadas as avaliações dos tratamentos aplicados aos participantes utilizando os objetos do estudo. Neste estudo adotou-se o desenho um fator (desenvolvimento de sistemas de IC) com dois tratamentos (com e sem a Athena) sem *crossover*, entretanto existem algumas variações desse desenho que precisam ser discutidas:

Um fator com dois tratamentos usando grupo de controle e pré-teste (D1): vantagens: i) a utilização do grupo de controle permite a análise de uma variável por vez; ii) o pré-teste permite a verificação da similaridade entre os resultados dos grupos. Desvantagens: i) a utilização de um grupo de controle pode gerar ameaças em relação a questões sociais e a maturação; ii) custo moderado. *Hipóteses* utilizadas nesse tipo de desenho: H_0: $\mu 1 = \mu 2$; H_1: $\mu 1 \neq \mu 2$, $\mu 1 < \mu 2$ ou $\mu 1 > \mu 2$ (μ representa o valor médio obtido pela métrica). *Testes estatísticos*: t-test e Mann-Whitney (WOHLIN et al., 2012).

Tabela 24 – Desenho experimental com um grupo de controle e um pré-teste.

	Desenvolvimento de Sistemas de IC			
	1ª Etapa		**2ª Etapa**	
Grupos	Não-Híbrido	Híbrido	Não-Híbrido	Híbrido
Grupo 1	Manual	Manual	Manual	Manual
Grupo 2	Manual	Manual	Athena	Athena

Um fator com dois tratamentos completamente aleatórios (D2): os participantes dos grupos 1 e 2 são selecionados aleatoriamente. Vantagens: i) minimiza os efeitos da maturação; ii) baixo custo de execução. Desvantagens: i) baixa precisão dos resultados. *Hipóteses*: H_0: $\mu 1 = \mu 2$; H_1: $\mu 1 \neq \mu 2$, $\mu 1 < \mu 2$ ou $\mu 1 > \mu 2$. *Testes estatísticos*: t-test e Mann-Whitney (WOHLIN et al., 2012).

Tabela 25 – Desenho experimental com tratamentos completamente aleatórios.

	Desenvolvimento de Sistemas de IC			
	Abordagem Manual		**Abordagem com Athena**	
Grupos	Não-Híbrido	Híbrido	Não-Híbrido	Híbrido
Grupo 1	X	X		
Grupo 2			X	X

Um fator com dois tratamentos com crossover (D3): os participantes dos grupos 1 e 2 são selecionados aleatoriamente, entretanto a ordem de aplicação dos tratamentos é alternada entre os grupos. Vantagens: i) melhora a precisão do experimento porque cada participante utiliza os dois tratamentos; ii) minimiza o efeito da ordem de aplicação dos tratamentos. Desvantagens: i) gera uma ameaça à validade interna do experimento em relação a maturação, pois os participantes podem aprender com o primeiro tratamento. *Hipóteses*: $d_j = y_{1,j} - y_{2,j}$ e μ_d é a média das diferenças; H_0: $\mu_d = 0$; H_1: $\mu_d \neq 0$, $\mu_d < 0$ ou $\mu_d > 0$. *Testes estatísticos*: Paired t-test e Wilcoxon (WOHLIN et al., 2012).

Tabela 26 – Desenho experimental com *crossover*.

	Desenvolvimento de Sistemas de IC			
	Abordagem Manual		**Abordagem com Athena**	
Grupos	Não-Híbrido	Híbrido	Não-Híbrido	Híbrido
Grupo 1	1ª Etapa	1ª Etapa	2ª Etapa	2ª Etapa
Grupo 2	2ª Etapa	2ª Etapa	1ª Etapa	1ª Etapa

O desenho D1 possui custo moderado e precisão nos resultados moderada. O desenho D2 possui um custo baixo, porém a precisão também é baixa. O desenho D3 possui custo alto e precisão alta. Dessa forma, considerando esses três esquemas, percebe-se que ao melhorar a precisão dos resultados o custo acaba aumentando. Porém, quanto maior o custo, maior será o investimento no experimento e maior será a probabilidade do surgimento de ameaças à validade interna, por exemplo, a mortalidade (abandono do experimento).

Consequentemente, decidiu-se pela utilização de um desenho experimental de um fator e dois tratamentos sem *crossover*. Esse modelo foi concebido a partir da adaptação de D2 e D3, e acaba priorizando a redução das ameaças relacionadas a questões sociais e mortalidade em detrimento da maturação. Entretanto, como a Athena é o primeiro tratamento, o efeito da maturação será negativo para a ferramenta Athena e positivo para a abordagem manual. Esse desenho é apresentando pela Tabela 27.

Tabela 27 – Desenho experimental utilizado na avaliação empiríca da Athena.

	Desenvolvimento de Sistemas de IC			
	1ª Etapa		**2ª Etapa**	
	1ª Sessão	2ª Sessão	3ª Sessão	4ª Sessão
Grupos	Abordagem com Athena		Abordagem Manual	
Grupo Único	Treinamento (2h)	Não-Híbrido Híbrido	Treinamento (2h)	Não-Híbrido Híbrido

O estudo foi realizado em duas etapas e quatro sessões, conforme apresentado na Tabela 27. Na primeira etapa todos os participantes desenvolveram um sistema não-híbrido e outro híbrido com o auxilio da Athena. Na segunda etapa, eles desenvolveram os mesmos sistemas sem o apoio da Athena. Antes do início de cada etapa, os participantes receberam um treinamento de 2 horas nas técnicas e ferramentas de IC que foram utilizadas no experimento. Durante o treinamento, eles realizaram as mesmas atividades que foram exigidas durante a execução, porém os objetos eram apenas similares.

5.9 Operação

Antes de iniciar o experimento, foi realizada uma breve apresentação, aos participantes, sobre as atividades associadas ao estudo, sem deixar claro quais eram as hipóteses envolvidas. No fim, ratificou-se que seria preservado o anonimato dos estudantes, com relação aos dados do estudo, explicando ainda como eles seriam utilizados. Todos concordaram com o estudo, não impondo qualquer restrição ao que foi relatado. Como o estudo não envolve riscos aos participantes, decidiu-se não utilizar o termo de consentimento apresentado no Apêndice C.

A primeira etapa iniciou-se com um treinamento sobre as técnicas de Inteligência Computacional necessárias para a execução das atividades: Sistemas de Inferência *Fuzzy*,

Perceptron Multicamadas (MLP) e o algoritmo genético NSGA-II. Depois os participantes foram apresentados à ferramenta Athena. Foi solicitado que eles desenvolvessem sistemas para resolver três problemas: inferir o valor da gorjeta do garçom considerando a qualidade do serviço e do atendimento; classificar os valores obtidos por uma porta lógica XOR com duas entradas; e solucionar o problema da mochila multiobjetivo. As instruções foram disponibilizadas de forma impressa e digitalizada. Durante o treino, os alunos podiam consultar o pesquisador para esclarecer possíveis dúvidas. Essa preparação ocorreu no horário da disciplina de Tópicos em Inteligência Artificial, ministrada pelo Departamento de Computação (UFPI) para alunos da graduação e pós-graduação.

Na segunda sessão, os participantes realizaram a execução da primeira etapa. Nessa fase, eles foram apresentados aos dois objetos do estudo: problema da estimação do tamanho de AgNPs e alocação de equipes de desenvolvimento (ver Seção 5.5). Durante essa fase, os alunos não podiam esclarecer dúvidas com o pesquisador responsável. Essa fase foi realizada no laboratório de informática do Departamento de Computação da UFPI e todas as máquinas já estavam preparadas, inclusive com o *plugin* da UseSkill funcionando. Essa ferramenta foi utilizada para coletar todas as ações dos usuários e o tempo de realização das sessões (ver Seção 5.7).

A segunda etapa também iniciou com um treinamento sobre as técnicas de IC necessárias para a execução das atividades, entretanto o tempo foi reduzido porque eles já tinham visto esses algoritmos durante treinamento anterior. O foco principal desse treino foi a utilização dos *frameworks* selecionados para implementar as técnicas de forma manual no estudo: JFuzzyLogic, Neuroph e JMetal. Foi solicitado que eles desenvolvessem sistemas para resolver os mesmos problemas discutidos no treinamento da Athena, entretanto, dessa vez, o desenvolvimento foi realizado com codificação em Java e com o apoio dos *frameworks* supracitados. Durante essa preparação os alunos podiam consultar o pesquisador para esclarecer possíveis dúvidas.

Por fim, na quarta sessão, os participantes realizaram a execução da segunda etapa. Nessa fase, eles foram apresentados novamente aos dois objetivos do estudo: problema da estimação do tamanho de AgNPs e alocação de equipes de desenvolvimento. Similar à execução da 1ª etapa, o ambiente já estava todo preparado e, após a conclusão dessa fase, os participantes foram submetidos ao questionário pós-experimento (ver Apêndice C) para que fosse possível obter *feedbacks* sobre a Athena e sobre a própria condução do estudo.

Dos 18 participantes, três foram eliminados porque não concluíram as duas etapas. Dois desses participantes abandonaram a disciplina e desistiram do experimento antes da 1ª etapa. O último realizou a 1ª etapa, mas não compareceu à 2ª etapa por motivo de saúde e não foi possível marcar uma nova rodada de execução. Os dados dos 15 participantes que restaram foram utilizados para obter os resultados desse experimento. A seção seguinte apresenta as análises estatísticas realizadas e discute os resultados obtidos.

5.10 Resultados e Discussões

As Tabelas 28 e 29 apresentam os resultados coletados durante a execução do experimento, quanto ao esforço e à eficácia dos sistemas de IC. A interpretação desses dados iniciou-se a partir da construção dos diagramas de caixa (*box-plot*) para que fosse possível visualizar a dispersão e variação dos dados, a assimetria da distribuição, os *quartis* e os *outliers* (medidas discrepantes). Essas informações são relevantes para a correta aplicação dos testes estatísticos durante a verificação das hipóteses (DIXON; JR, 1957).

Tabela 28 – Resultados do experimento quanto ao esforço despendido no desenvolvimento dos Sistemas Baseados em Inteligência Computacional.

	Esforço de Desenvolvimento medido em Segundos			
	Estimação do Tamanho de AgNPs		**Alocação de Equipes**	
#	**Athena**	**Manual**	**Athena**	**Manual**
1	1241,14	2624,46	1376,06	3267,33
2	597,32	3300,00	981,35	24300,00
3	1041,60	3000,00	1019,85	4800,00
4	564,25	2271,89	1283,56	6360,00
5	658,39	4380,00	1515,27	9120,01
6	695,28	5286,00	862,48	7065,00
7	738,88	3440,36	1034,63	5085,98
8	810,03	2046,95	878,81	5460,00
9	535,94	2073,30	1135,51	7650,00
10	632,34	2640,42	1086,09	5652,09
11	732,86	9300,00	1732,17	19620,00
12	940,47	5520,00	1620,53	23520,00
13	545,90	4200,00	847,09	11700,00
14	342,57	1171,07	503,98	3234,78
15	397,35	1449,99	602,21	1988,34

Tabela 29 – Resultados do experimento quanto à eficácia dos SBIC desenvolvidos pelos participantes.

	Eficácia dos Sistemas de IC			
	Estimação do Tamanho de AgNPs		**Alocação de Equipes**	
#	**Athena**	**Manual**	**Athena**	**Manual**
1	1,00	1,00	1,00	1,00
2	1,00	0,00	0,50	0,00
3	1,00	0,33	1,00	0,33
4	1,00	1,00	1,00	0,33
5	1,00	1,00	1,00	0,50
6	1,00	0,00	0,50	0,00
7	1,00	1,00	1,00	0,33
8	1,00	1,00	1,00	0,33
9	1,00	1,00	1,00	0,33
10	1,00	1,00	1,00	0,50
11	1,00	1,00	1,00	0,00
12	1,00	0,50	1,00	0,33
13	1,00	1,00	1,00	0,33
14	1,00	1,00	1,00	1,00
15	1,00	1,00	1,00	1,00

Nota: A eficácia igual a ZERO significa que o código desenvolvido pelo participante não executa.

As figuras 41 e 42 apresentam, respectivamente, os diagramas de caixa para as métricas de esforço e de eficácia. Foram encontrados quatro *outliers* nos dados de esforço e seis nos dados de eficácia. Esses valores estão destacados nas Tabelas 28 e 29, e são justificados por três fatores: baixo domínio da linguagem de programação Java por parte dos participantes 2, 11 e 12; problemas de usabilidade no editor gráfico da Athena, identificados pelo participante 1; e, por fim, falha na configuração de execução dos sistemas desenvolvidos pelos participantes 2 e 6 como auxílio da Athena. Em geral, os *outliers* são removidos do conjunto de dados antes da realização dos testes estatísticos, entretanto, devido ao fato de que nenhuma das causas de aparecimento desses valores atípicos está relacionada a erros de medição ou condução do experimento, decidiu-se pela manutenção dessas informações, as quais representam a variabilidade inerente dos elementos da população (FIGUEIRA, 1998).

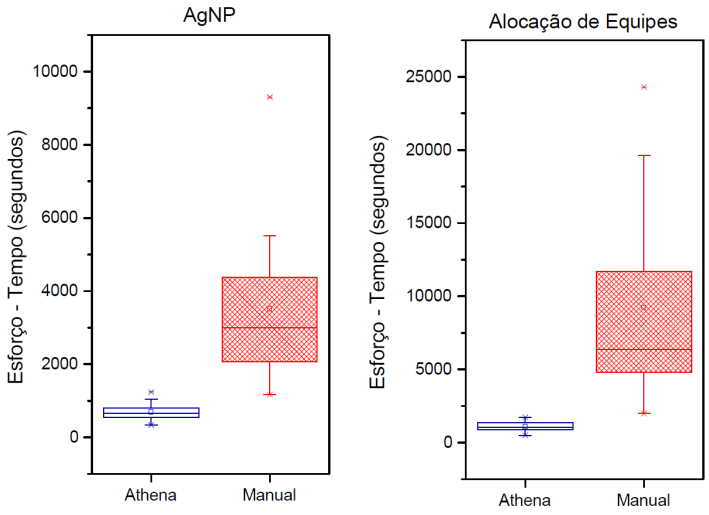


Figura 41 – *Box-plot* do esforço para o desenvolvimento dos sistemas de IC.

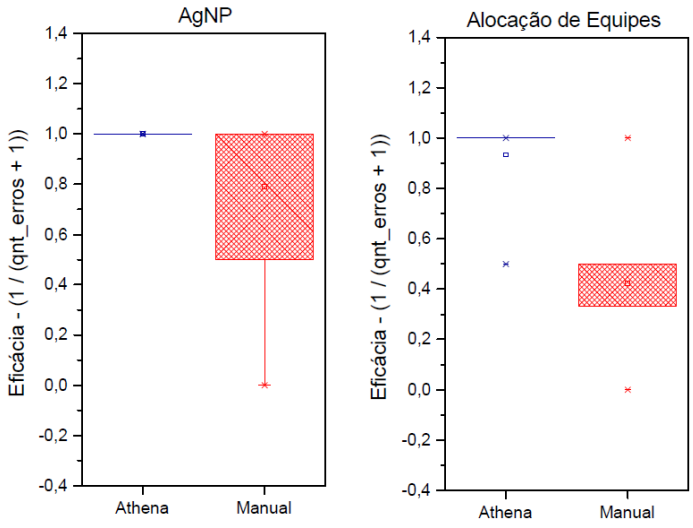


Figura 42 – *Box-plot* da eficácia dos sistemas de IC.

Analisando a Figura 41, pode-se observar que o desenvolvimento de SBIC utilizando a Athena requer menos esforço do que o desenvolvimento com o apoio de bibliotecas de código (PEER, 2004; PAMPARA; ENGELBRECHT; CLOETE, 2008). Esse resultado era esperado devido ao fato de que a Athena remove a complexidade associada à programação das técnicas de IC e, apesar de incluir a necessidade de conhecer o funcionamento da própria ferramenta, o método Athena ainda foi capaz de obter resultados significativamente melhores, principalmente quando se trata de Sistemas Híbridos.

Quanto à eficácia, a Figura 42 apresenta indícios de que a implementação manual dos sistemas de IC é propensa a erros. Esse comportamento já foi constatado por outros pesquisadores (ALBA et al., 2002). Dessa forma, considerando a correta configuração das técnicas, a Athena garante a obtenção de resultados confiáveis. Além disso, em ambos os diagramas de caixa, é possível notar que a variabilidade dos dados obtidos com a Athena é menor do que a variância dos dados da abordagem manual (sem Athena).

Apesar da verificação desses indícios, a análise apenas dos diagramas de caixa não fornece evidências claras para definir se as hipóteses do estudo devem ser aceitas ou rejeitadas (DIXON; JR, 1957). Dessa forma, faz-se necessário a utilização de alguns testes estatísticos para identificar se as métricas obtidas com o auxílio da Athena são, de fato, estatisticamente diferentes das métricas obtidas com o apoio das bibliotecas de código. Esses testes foram realizados com o apoio do *software* OriginPro 9.1 (SEIFERT, 2014).

Inicialmente, os dados foram submetidos ao teste de normalidade *Anderson-Darling* (SCHOLZ; STEPHENS, 1987). Essa etapa é importante, pois alguns testes estatísticos possuem premissas relacionadas à normalidade das amostras.

Tabela 30 – Resultado do teste de normalidade.

Dados relacionados ao Esforço						
Trat.	**Objeto**	$\overline{X}$	$\sqrt{\sigma^2}$	α	**Anderson-Darling** ($\rho - value$)	**Normal**
Athena	AgNP	698,288	237,144	0,05	0,440	Sim
Manual	AgNP	3513,629	2057,258	0,05	0,063	Sim
Athena	Equipes	1098,639	353,703	0,05	0,798	Sim
Manual	Equipes	9254,902	7312,100	0,05	0,001	Não
Dados relacionados à Eficácia						
Athena	AgNP	1,000	0,000	0,05	0,000	Não
Manual	AgNP	0,789	0,381	0,05	0,000	Não
Athena	Equipes	0,933	0,176	0,05	0,000	Não
Manual	Equipes	0,422	0,338	0,05	0,003	Não

Nota: $\overline{X}$ - Média dos Valores; $\sqrt{\sigma^2}$ - Desvio Padrão; α - Nível de Significância.

Conforme exibido na Tabela 30, pode-se afirmar com o nível de confiança de 95% que apenas os três primeiros conjuntos de dados seguem a distribuição normal. Dessa forma, utilizou-se o teste não-paramétrico de *Wilcoxon* com amostras pareadas (WILCOXON, 1945; GIBBONS; CHAKRABORTI, 2011) para avaliar as hipóteses deste experimento.

O teste *Wilcoxon* retornou um *p-value* igual a $0,00006$ quando aplicado aos dados de esforço coletados durante o experimento. Esse teste foi realizado com as amostras

pareadas, de forma a comparar o esforço despendido no desenvolvimento de cada objeto com o apoio dos diferentes tratamentos. Dessa forma, pode-se afirmar com 99% de certeza (nível de significância igual a $0,01$) que as duas distribuições são estatisticamente diferentes. Portanto, a hipótese H_0 *esforço* foi rejeitada e a resposta à Questão de Pesquisa 1 (QP1) é: *considerando o contexto deste experimento, o esforço necessário para o desenvolvimento de sistemas de IC com a utilização da ferramenta Athena é, de fato, menor que o esforço exigido para desenvolver os mesmos sistemas de IC sem o apoio da ferramenta.*

Segundo a análise estatística, as amostras de eficácia dos sistemas desenvolvidos com e sem o apoio da Athena para solucionar o problema da estimação do tamanho de AgNPs não são estatisticamente diferentes (*p-value* igual a $0,125$ e nível de significância igual a $0,01$). Entretanto, as amostras obtidas para o problema de alocação de equipes são estatisticamente diferentes (*p-value* igual a $0,00049$ e nível de significância igual a $0,01$). Essa distinção deve-se ao fato de que a resolução do problema de alocação de equipes envolve mais de uma técnica de IC. Isso aumenta a complexidade do desenvolvimento e acaba gerando mais erros. Assim, a hipótese H_0 *eficácia* não pode ser rejeitada e a resposta à Questão de Pesquisa 2 (QP2) é: *a eficácia dos sistemas de IC desenvolvidos com a Athena não é maior que a eficácia dos sistemas de IC desenvolvidos sem o apoio da ferramenta. Entretanto, observou-se que a construção de soluções híbridas sem o auxílio da Athena é propensa a erros.*

Tabela 31 – Resultados das análises estatísticas utilizando o teste Wilcoxon.

Dados relacionados ao Esforço (QP1)						
Trat. 1	**Objeto 1**	**Trat. 2**	**Objeto 2**	α	**Wilcoxon Test** ($\rho - value$)	**Estatísticamente Diferentes**
Athena	AgNP	Manual	AgNP	0,01	0,00006	Sim
Athena	Equipes	Manual	Equipes	0,01	0,00006	Sim
Dados relacionados à Eficácia (QP2)						
Athena	AgNP	Manual	AgNP	0,01	0,12500	Não
Athena	Equipes	Manual	Equipes	0,01	0,00049	Sim

Nota: α - Nível de Significância.

Tabela 32 – Resultados das análises de correlação utilizando o coeficiente de Pearson.

Dados relacionados ao Esforço (QP1)							
	Athena AgNP	Athena Equipes	Manual AgNP	Manual Equipes	Técnicas de IC	Ferramentas de IC	Java
Athena - AgNP	1,00	-	-	-	-0,03	-0,03	-0,26
Athena - Equipes	-	1,00	-	-	-0,48	-0,52	-0,09
Manual - AgNP	-	-	1,00	-	-0,77	-0,66	-0,23
Manual - Equipes	-	-	-	1,00	-0,51	-0,51	-0,28
Técnicas de IC	-0,03	-0,48	-0,77	-0,51	1,00	-	-
Ferramentas de IC	-0,03	-0,52	-0,66	-0,51	-	1,00	-
Conhecimento em Java	-0,26	-0,09	-0,23	-0,28	-	-	1,00
Dados relacionados à Eficácia (QP2)							
Athena - AgNP	1,00	-	-	-	0,00	0,00	0,00
Athena - Equipes	-	1,00	-	-	0,49	0,14	0,47
Manual - AgNP	-	-	1,00	-	0,41	-0,01	0,26
Manual - Equipes	-	-	-	1,00	0,66	0,34	0,42
Técnicas de IC	0,00	0,49	0,41	0,66	1,00	-	-
Ferramentas de IC	0,00	0,14	-0,01	0,34	-	1,00	-
Conhecimento em Java	0,00	0,47	0,26	0,42	-	-	1,00

A Tabela 31 exibe o resumo dos testes estatísticos usando o método de Wilcoxon. Além disso, fez-se um estudo para avaliar a relação entre os dados obtidos na caracterização dos participantes e os resultados do experimento. Esse estudo ratificou a relação inversamente proporcional entre o esforço empregado no desenvolvimento de Sistemas Inteligentes sem o apoio da Athena e o nível de conhecimento em técnicas e ferramentas de IC, por meio da correlação de Pearson (DIXON; JR, 1957) ($\alpha = 0,05$). A Tabela 32 sintetiza os resultados das análises de correlação. Nela é possível perceber também que o conhecimento em Java possui uma correlação fraca com o esforço e a eficácia dos sistemas construídos com a Athena.

Por fim, devido ao aspecto subjetivo da terceira Questão de Pesquisa (QP3), utilizou-se as respostas do questionário pós-experimento (ver Apêndice C) para obter uma conclusão sobre esse quesito. Além disso, os itens iniciais desse questionário buscaram identificar problemas no planejamento e condução do experimento.

A maioria dos participantes (80%) respondeu que o estudo foi bem executado e que os treinamentos foram adequados ou muito adequados ao propósito do experimento. Quanto à Athena, 80% dos voluntários qualificaram a Athena como excelente e todos (100%) atestaram que estão dispostos a utilizar a ferramenta em ambiente acadêmico e industrial. Assim, a hipótese H_0 *qualidade* foi rejeitada pelo pesquisador e a resposta à Questão de Pesquisa 3 (QP3) é: *considerando o contexto deste estudo e os comentários dos participantes, a percepção de qualidade dos usuários quanto ao desenvolvimento de Sistemas de IC com o auxílio da Athena é maior que sistemas de IC desenvolvidos sem o suporte dessa ferramenta.* A Figura 43 sintetiza a avaliação de seis dos dez critérios presentes no questionário pós-experimento.

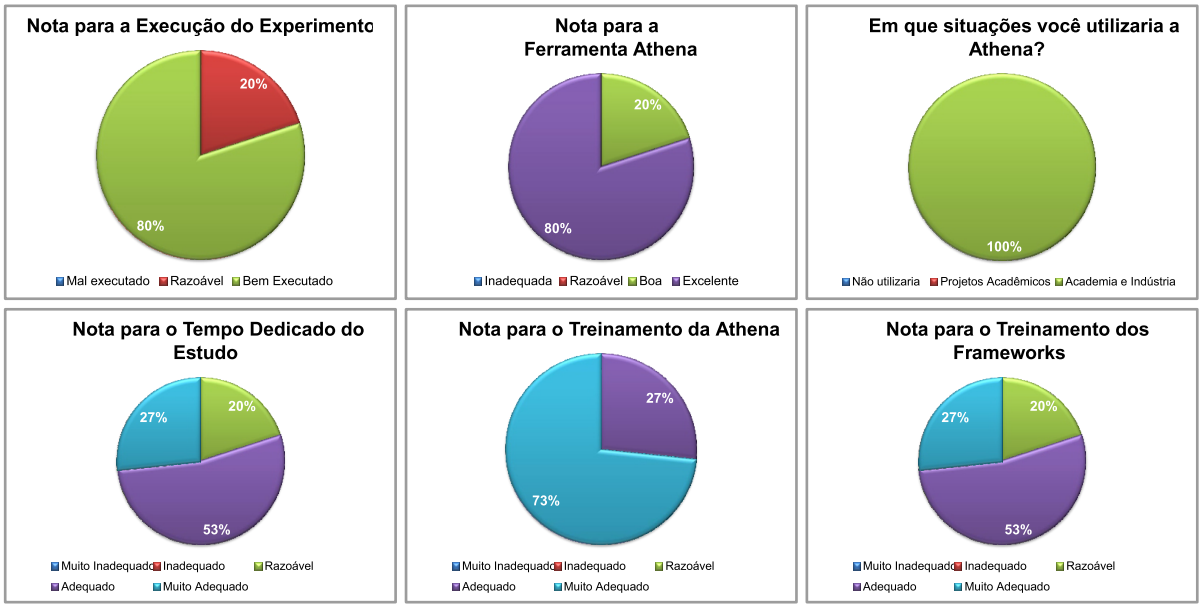


Figura 43 – Avaliação da condução do experimento e da ferramenta Athena.

5.11 Ameaças à Validade

Uma questão fundamental a respeito dos resultados de um estudo experimental é o quão válido são os resultados obtidos. Portanto, é importante tratar os aspectos relacionados às ameaças à validade durante a fase de planejamento do experimento, para que seja possível antecipar e contornar problemas que tornem os resultados inválidos para a população em análise (WOHLIN et al., 2012).

Segundo Cook e Campbell (COOK; CAMPBELL; DAY, 1979), existem quatro tipos de ameaças à validade de um experimento: validade de conclusão, validade interna, validade de construção e validade externa.

5.11.1 Validade de Conclusão

Essa categoria de ameaça à validade preocupa-se com questões que afetam a capacidade de chegar à conclusão correta sobre as relações entre os tratamentos e os resultados do experimento. Essas questões incluem, por exemplo, a escolha do método estatístico, a escolha do tamanho da amostra e cuidados na aplicação e medição do experimento. A seguir são apresentadas algumas ameaças que podem ocorrer nesse estudo e como foram tratadas:

Escolher um teste com baixo poder estatístico: o poder do teste estatístico está na habilidade de revelar padrões verdadeiros dentro de um conjunto de dados. Se o poder do teste é baixo, então há uma grande probabilidade de que a conclusão seja errada. Para evitar esse problema, o teste estatístico deve ser escolhido de acordo com o desenho experimental adotado. A escolha dos testes estatísticos utilizados nesse experimento foi baseada nas recomendações apresentadas por WOHLIN et al. (2012).

Violar as premissas dos testes estatísticos: determinados testes estatísticos possuem algumas premissas que devem ser consideradas para sua utilização, por exemplo, assumir que o conjunto de dados segue a distribuição normal. A violação dessas premissas pode levar o pesquisador a acreditar em conclusões erradas. Para evitar esse problema, foi realizada uma análise previa dos dados, com o objetivo de identificar se seguem ou não uma distribuição normal. O teste utilizado foi o *Anderson-Darling* (SCHOLZ; STEPHENS, 1987).

Confiabilidade das medições: a validade de um experimento está diretamente ligada com o nível de confiança das medições. O princípio básico relacionado a essa ameaça à validade diz que quando se realiza a medição de um evento duas vezes, os resultados devem ser iguais. Portanto, medidas objetivas, ou seja, que não dependem da interpretação do ser humano, são mais confiáveis que medidas subjetivas. Neste estudo adotou-se duas medidas objetivas (esforço e eficácia) e uma medida subjetiva (percepção de qualidade).

Confiabilidade da aplicação dos tratamentos: essa ameaça preocupa-se com o risco de que a implementação dos tratamentos não seja a mesma aplicada a todos os participantes. Para mitigar essa ameaça, esse estudo utilizou dois tratamentos e com apenas uma implementação para cada tratamento. Portanto, não há a possibilidade de que o mesmo tratamento seja aplicado de forma diferente aos participantes.

Distúrbios na configuração do experimento: elementos externos podem causar distúrbios nos resultados, por exemplo, barulho durante o experimento ou interrupções externas, etc. Para mitigar essa ameaça, esse estudo foi realizado em ambiente silencioso e fora do ambiente de trabalho ou estudo dos participantes.

5.11.2 Validade Interna

Se uma relação é observada entre o tratamento e os resultados, é necessário assegurar que essa relação é do tipo causa-efeito e que não é decorrente de um fator que não foi controlado. Em outras palavras, esse tipo de ameaça procura atestar que os tratamentos causam os resultados. Dentre os fatores que podem impactar na validade interna do experimento, destaca-se: a seleção e agrupamento dos participantes, o modo como os participantes são tratados ou o acontecimento de algum evento especial durante o estudo, etc. A seguir são apresentadas algumas ameaças à validade interna e como foram tratadas:

História: em um experimento, diferentes tratamentos podem ser aplicados aos mesmos objetos em diferentes momentos. Então há o risco de que a história afete os resultados do estudo, pois as circunstâncias em ambas ocasiões não são as mesmas. Em decorrência desse risco, as etapas de realização deste experimento foram previamente agendadas com os participantes e não foram marcadas imediatamente após finais de semanas e feridos, ou em dias com acontecimentos únicos, por exemplo, final de campeonato brasileiro de futebol.

Maturação: esse efeito ocorre porque os participantes reagem de diferentes maneiras com o passar do tempo. Por exemplo, os participantes podem ser afetados negativamente por tarefas chatas e repetitivas ou positivamente com o aprendizado durante o experimento. A maturação foi propositalmente utilizada contra a Athena para reduzir os custos do estudo. Isso foi possível com a adoção de um grupo único e com a definição da Athena como primeiro tratamento a ser aplicado. Dessa forma, ao utilizar a abordagem manual (sem Athena), os participantes possivelmente absorveram conhecimentos sobre os elementos envolvidos na experimentação. Essa organização acabou reduzindo os custos, pois não foi necessário realizar o *crossover* apresentado no desenho experimental D3.

Mortalidade: esse efeito deve-se aos participantes que abandonam o experimento. Neste estudo foi realizada uma análise para caracterizar o motivo da desistência e procurar identificar se os desistentes representam um conjunto considerável da amostra.

Ameaças sociais: esse efeito deve-se ao fato de que os participantes podem ter seus resultados influenciados pelo relacionamento com os demais grupos do estudo, criando rivalidades ou ressentimentos. Para evitar problemas com essa ameaça foi realizada uma apresentação abordando os objetivos do estudo, além de deixar claro que os participantes não estão inseridos em nenhum tipo de competição e que os resultados são confidenciais.

5.11.3 Validade de Construção

Essa categoria de ameaça preocupa-se com a relação entre a teoria e a observação, ou seja, se o tratamento reflete a causa e o resultado reflete o efeito. Dentro dessa categoria existem algumas ameaças relacionadas ao desenho do experimento e a fatores sociais, entretanto para este estudo identificou-se apenas uma ameaça a ser tratada: a expectativa dos pesquisadores. Essa ameaça foi mitigada com a expressa proibição de qualquer ajuda por parte do pesquisador responsável durante a execução do experimento.

5.11.4 Validade Externa

Essa categoria de ameaça preocupa-se com a generalização dos resultados obtidos pelo estudo. Existem três riscos principais: utilizar participantes não adequados, realizar o experimento em ambiente não adequado ou com ferramentas inadequadas e executar o estudo em um período no qual a história possa afetar os resultados. A seguir são apresentadas algumas ameaças à validade externa e como foram tratadas:

Seleção dos Participantes: a seleção dos participantes é uma ameaça à validade externa do experimento porque a amostra pode não ser representativa frente à população na qual desejamos generalizar os resultados. Para ampliar a capacidade de generalização dos resultados obtidos, este estudo foi realizado com alunos de pós-graduação e graduação.

Ferramentas inadequadas: para evitar efeitos negativos da utilização de materiais inadequados, os *frameworks* utilizados na abordagem manual foram selecionadas de acordo com estudos que comprovam sua utilização na prática industrial e acadêmica.

História: esse efeito está relacionado a realização do experimento em um período no qual a história possa influenciar os resultados, por exemplo, se estudo for realizado logo após a realização de uma prova prática sobre algoritmos de IC, os participantes tendem a se comportar diferente do que alguns dias antes. Logo, a data do experimento foi cuidadosamente proposta, levando em consideração o perfil dos participantes.

5.11.5 Prioridade entre as Ameaças

Existem conflitos entre alguns tipos de ameaças à validade, pois ao melhorar um ponto, outro acaba perdendo. Por exemplo, realizar um estudo experimental com estudantes de graduação possibilita a utilização de um grande número de participantes,

reduzindo a heterogeneidade da amostra e aumentando a confiabilidade dos resultados. Entretanto, a capacidade de generalização dos resultados para a indústria de *software* é baixa, ou seja, as ameaças à validade de construção foram bem tratadas, porém a validade externa foi prejudicada. Portanto, dependendo do tipo do experimento, as ameaças à validade podem ser diferentemente priorizadas.

Estudos relacionados a pesquisas aplicadas, que é a área-alvo da maioria dos experimentos de Engenharia de *Software*, costumam priorizar a relação de causa-efeito entre os tratamentos e os resultados (validade interna) e a capacidade de generalização dos resultados (validade externa). Este estudo também seguiu essa ordem de prioridade, vindo logo em seguida a validade de construção e a validade de conclusão.

5.12 Estudo de Usabilidade

Conforme descrito anteriormente, utilizou-se a ferramenta UseSkill para coletar todas as ações realizadas na Athena durante o experimento. Isso instigou a realização de um pequeno estudo de usabilidade dentro da própria avaliação experimental.

O funcionamento da UseSkill, ilustrado pela Figura 44, exige a realização de quatro etapas: criar testes; participar de teste; gerar relatórios; e análise dos relatórios (SOUZA et al., 2015). Dessa forma, cadastrou-se o roteiro do experimento e os participantes do estudo foram convidados a realizar o teste. Esse roteiro solicitava ao voluntário a realização de três ações: *login* na Athena; construção de um sistema de IC para solucionar o problema da estimação do tamanho de AgNPs; e o desenvolvimento de um SBIC para resolver o problema de alocação de equipes de desenvolvimento. Depois que todos os participantes finalizaram as tarefas, foi possível analisar as ações capturadas.

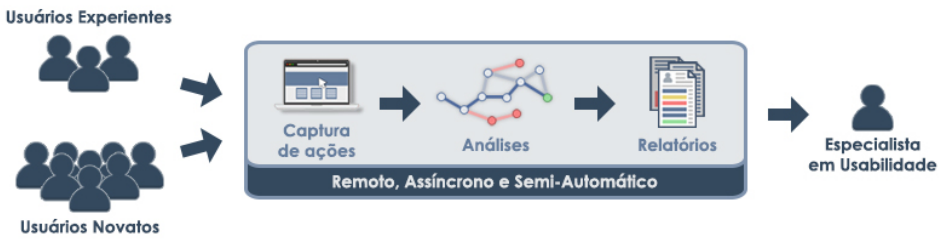


Figura 44 – Método proposto pela UseSkill.

Esse estudo não foi realizado com o rigor necessário para ser considerado uma avaliação de usabilidade, pois não foi possível concluir a análise das ações, todos os problemas foram identificados por meio do *feedback* dos próprios usuários e o estudo foi realizado pelo pesquisador responsável por este trabalho, o qual não possui vastos conhecimentos em usabilidade. Mesmo assim, uma série de problemas foram identificados. A Tabela 33 exibe esses problemas e especifica quais já foram corrigidos.

Tabela 33 – Problemas de usabilidade identificados na Athena.

#	Problema	Estado
1	Perda das configurações de execução ao atualizar a página	Corrigido
2	Barra de ações no editor deveria ser fixa	Corrigido
3	Ausência de um feedback ao salvar a descrição dos módulos	Corrigido
4	Inconsistência no padrão de exibição dos botões de informação	Pendente
5	Mensagem confusa ao solitar o tipo de execução (síncrona ou assíncrona)	Corrigido
6	Dificuldades ao definir o nome e tipo das variáveis	Corrigido
7	Não há a possibilidade de inverter a orientação dos módulos (vertical ou horizontal)	Corrigido
8	Algumas telas não são responsive	Pendente
9	Ao adicionar dois módulos iguais fica difícil distingui-los durante a configuração de execução	Corrigido

Como não foi possível concluir a análise das ações, devido a baixa *expertise* nessa área, decidiu-se utilizar esses dados em uma avaliação de usabilidade criteriosa, que será realizada em parceria com os especialistas em usabilidade, pertencentes ao grupo de pesquisa do Laboratório EaSII (UFPI). Essa nova avaliação está incluída nos trabalhos futuros e representa mais uma oportunidade para a obtenção de novas contribuições científicas.

5.13 Estudos Anteriores

Antes da execução deste estudo experimental, dois outros estudos já haviam sido realizados, porém sem o rigor exigido para tal avaliação. Mesmo assim, esses estudos anteriores serviram de preparação para esta realização mais formal.

A primeira avaliação foi realizada em 2014 com o objetivo de analisar a eficiência (esforço) do desenvolvimento de SBIC com e sem a utilização da Athena. Quatro estudantes do curso de Ciência da Computação da UFPI participaram desse estudo preliminar: dois graduandos e dois graduados. Eles possuíam boas habilidades em programação e apenas um deles não possuía bons conhecimentos em algoritmos de Inteligência Computacional. Os estudantes foram divididos em dois grupos de tal forma que houvesse um graduando e um graduado em cada grupo. Os alunos graduandos realizaram a implementação dos problemas (alocação de equipes de desenvolvimento e priorização de casos de testes de regressão) utilizando a Athena, enquanto que os graduados desenvolveram utilizando os *frameworks*: JMetal (DURILLO; NEBRO; ALBA, 2010) e jFuzzyLogic (CINGOLANI; ALCALÁ-FDEZ, 2013). Ambos registraram o tempo gasto (esforço) em horas.

A Figura 45 exibe o tempo gasto pelos estudantes para realizar as tarefas designadas. Pode-se observar que o desenvolvimento utilizando a Athena requer significativamente menos esforço (aqui mensurado pelo tempo de desenvolvimento) quando comparado com o desenvolvimento seguindo a abordagem manual, ou seja, com o auxílio de *frameworks*. Ambas as implementações, abordagem manual e abordagem com a Athena, foram consideradas corretas pelo pesquisador, pois apresentaram resultados simulares aos expostos nos artigos científicos (BRITTO et al., 2012) e (SANTOS-NETO et al., 2012).

Considerando esse contexto, foi possível perceber indícios de que a Athena poderia reduzir o tempo necessário para construir SBIC, especialmente quando se tratasse de abordagens híbridas, ou seja, que combinam duas ou mais técnicas de IC. Todavia, essa primeira avaliação possui uma série de ameaças à validade, por exemplo, baixa quantidade de participantes (apenas quatro) e baixo conhecimento dos alunos que implementaram o sistema sem o apoio da Athena, em relação aos *frameworks* utilizados. Essas ameaças motivaram o planejamento deste estudo rigoroso.

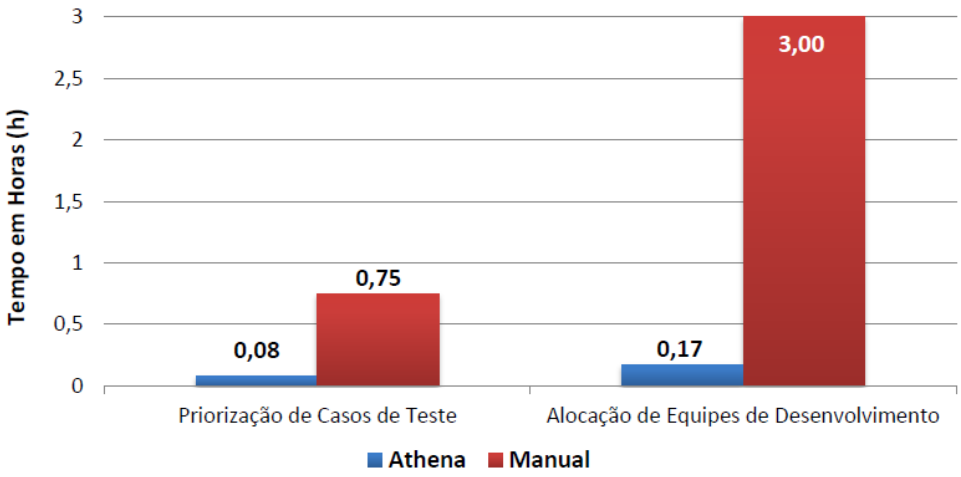


Figura 45 – Esforço gasto (tempo em horas) para desenvolver os Sistemas Baseados em Inteligência Computacional com e sem o uso da Athena.

Após a conclusão do plano experimental que definiu este experimento, decidiu-se realizar um estudo piloto para garantir que o arcabouço construído estava adequado para a avaliar o desenvolvimento de Sistemas Inteligentes. Esse estudo piloto foi a segunda iniciativa. Ele foi realizado com 8 alunos do curso de Ciência da Computação e acabou evidenciando um falha no planejamento do experimento. O tempo dedicado ao desenvolvimento dos sistemas foi limitado em 45 minutos. Essa restrição gerou 100% de mortalidade, pois os participantes não conseguiram concluir a construção dos sistemas utilizando a abordagem manual. Assim, foi necessário remover essa restrição para evitar essa alta taxa de mortalidade.

Por fim, notou-se que, em todos os experimentos realizados, o tempo de execução dos sistemas construídos com a Athena é maior do que o tempo de execução dos sistemas construídos a partir da abordagem manual. Essa situação motivou a realização de uma avaliação de performance, com o objetivo de estudar o impacto do tamanho do problema no tempo de execução das técnicas de IC implementadas com o auxílio da Athena. Essa avaliação seria a base para esclarecer a relação custo-benefício da execução de algoritmos de IC nessa ferramenta, entretanto o estudo não foi realizado com o devido rigor e, portanto, não foi possível obter nenhuma conclusão. Essa avaliação de performance está descrita no Apêndice E e foi incluída nesta dissertação apenas para auxiliar trabalhos futuros.

5.14 Considerações Finais

Neste capítulo apresentou-se um estudo experimental realizado com o intuito de analisar o processo de desenvolvimento de Sistemas Inteligentes com e sem o auxílio da ferramenta Athena. Esse processo foi avaliado considerando os seguintes critérios: esforço, medido em segundos, despendido na construção dos sistemas; eficácia do sistemas de IC e percepção de qualidade medida com questionários aplicados aos voluntários do estudo.

O experimento mostrou que a Athena pode reduzir o custo do desenvolvimento de Sistemas Baseados em Inteligência Computacional, especialmente quando se trata de abordagens híbridas. Essa redução de custos ocorre em função da diminuição do tempo despendido com a construção dos sistemas.

Quanto à eficácia, o estudo apontou que não há diferenças significativas entre a qualidade das respostas obtidas com a Athena e com o auxílio de bibliotecas de código. Quanto à percepção de qualidade, medida em termos de satisfação dos participantes da avaliação empírica, a Athena foi avaliada positivamente. Todos os voluntários ratificaram o interesse em utilizar a ferramenta em ambientes acadêmicos e industriais.

Finamente, ressalta-se que é importante a realização de um Estudo de Caso (WOHLIN et al., 2012) dentro de uma empresa para solidificar os resultados percebidos no ambiente acadêmico. Essa proposta será discutida na seção de Trabalhos Futuros.

6 Conclusões e Trabalhos Futuros

O estudo dos Sistemas Inteligentes vem se mostrando cada vez mais importante devido à sua capacidade de solucionar diversos tipos de problemas nas mais variadas áreas do conhecimento. Isso reforça a transversalidade dessa linha de pesquisa e justifica não só o crescimento do número de trabalhos científicos que aplicam técnicas inteligentes em ambientes complexos e dinâmicos mas, também, o fortalecimento do desenvolvimento de ferramentas aptas a auxiliar a construção, manutenção e aplicação de Sistemas Baseados em Inteligência Computacional. Entretanto, após uma análise crítica dessas ferramentas de apoio, percebeu-se lacunas passíveis de exploração, por exemplo, a dificuldade na construção de Sistemas Híbridos e na integração com outros sistemas.

Considerando esse contexto, o objetivo geral deste trabalho é propor um ambiente integrado (arquitetura e ferramenta) capaz de facilitar o desenvolvimento de Sistemas Baseados em Inteligência Computacional. Para concretizar esse objetivo, decidiu-se adotar uma metodologia composta pelas seguintes ações: elaborar um estudo detalhado sobre as técnicas de Inteligência Computacional, evidenciando suas características fundamentais; realizar um mapeamento sistemático sobre ferramentas de IC, objetivando a identificação dos principais trabalhos dessa linha de pesquisa; especificar uma arquitetura capaz de auxiliar o desenvolvimento de SBIC; desenvolver uma ferramenta Web que atenda as especificações da arquitetura proposta; avaliar a ferramenta por meio de estudos empíricos; e sumarizar os resultados obtidos para publicação em eventos científicos.

A Athena - principal resultado produzido por esta pesquisa - materializa o ambiente integrado (arquitetura e ferramenta) capaz de auxiliar a construção, manutenção e aplicação de SBIC. Esse ambiente foi desenvolvido com os seguintes princípios: i) simplificar o desenvolvimento de Sistemas Inteligentes por meio de uma arquitetura modular que encapsula a complexidade interna das técnicas, deixando exposto apenas o que interessa ao usuário final (entradas, configurações e saídas); ii) permitir a inclusão de novos módulos em tempo de execução; iii) disponibilizar as técnicas de IC seguindo o modelo de *software* como serviço, para obter vantagens da Computação em Nuvem (alta disponibilidade, serviço sob demanda, mobilidade e performance); e iv) oportunizar a colaboração entre grupos de pesquisa espalhados pelo mundo.

Atualmente, a ferramenta possui 35 módulos com técnicas pertencentes às cinco subáreas da Inteligência Computacional. Os usuários podem desenvolver Sistemas Inteligentes híbridos de forma simples e escalável, utilizando um editor gráfico que propicia a programação visual das técnicas. Além disso, a Athena está em processo de registro no Instituto Nacional da Propriedade Industrial (INPI).

As contribuições deste trabalho são:

- **Realização de um mapeamento sistemático**: desenvolveu-se um estudo secundário com o objetivo de construir conhecimento a partir das publicações relacionadas às ferramentas para desenvolvimento de SBIC (Capítulo 2). A descrição da metodologia possibilita a replicação desse tipo de estudo e os resultados facilitam a escolha de uma ferramenta para resolver determinado problema e serviram de base para o direcionamento de esforços durante a construção da Athena;

- **A padronização da utilização das técnicas de IC**: para facilitar a aplicação dos algoritmos de IC, a arquitetura proposta prescreve a padronização da utilização desses algoritmos por meio do conceito de módulo. Dessa forma, independente da complexidade da técnica de IC, ela deve ser vista como um módulo que utiliza sua lógica interna para processar as entradas e devolver as saídas. Essa lógica ainda pode ser ajustada pelo usuário com as configurações de execução. Além disso, como os dados de entrada e saída também são padronizados, os módulos podem ser facilmente interligados para construir Sistemas Híbridos. Essa contribuição, apresentada no Capítulo 3, é o principal pilar da Arquitetura Proposta. Além disso, pode-se destacar o detalhamento do modelo conceitual dessa arquitetura (ver Seção 3.1), viabilizando sua utilização por outros pesquisadores para trabalhos futuros;

- **Athena**: após a definição da arquitetura desenvolveu-se uma ferramenta que permite a construção de Sistemas Inteligentes de forma simples, dinâmica e escalável. A Athena funciona independente de plataforma computacional e permite que seus usuários compartilhem resultados de experimentos com outros pesquisadores ao redor do mundo. O Capítulo 4 apresenta uma visão geral sobre a ferramenta e alguns exemplos de utilização. Além disso, existem outras funcionalidades que não foram descritas neste documento, mas que estão presentes na documentação da ferramenta;

- **Avaliação da proposta**: realizou-se uma série de estudos empíricos para avaliar a proposta deste trabalho. O primeiro foi uma avaliação experimental (Capítulo 5), conduzida de acordo com as recomendações da Engenharia de *Software* Experimental, com o objetivo de caracterizar a proposta em relação ao esforço, eficácia e percepção de qualidade dos sistemas desenvolvidos com o apoio da Athena. Nesse experimento pode-se constatar que a Athena reduz os custos associados à construção de SBIC, mantendo o nível de eficácia e elevando a percepção de qualidade. Depois, realizou-se um estudo de performance (Apêndice E) para investigar o impacto do tamanho do problema no desempenho da ferramenta. Esse estudo não foi realizado com o rigor exigido para tal avaliação, portanto não foi possível obter conclusões concretas sobre a performance da proposta. Entretanto, essa avaliação preliminar serviu como preparação para uma investigação mais formal.

Destaca-se como principal contribuição desta pesquisa a reunião de diferentes tecnologias em torno de uma arquitetura e ferramenta desenvolvidas para unificar as técnicas de IC dentro de um novo conceito, cunhado a partir deste trabalho, denominado de Inteligência Computacional como Serviço (CIaaS). Conclui-se que, devido aos resultados das avaliações, a abordagem proposta é adequada para auxiliar o desenvolvimento de Sistemas Baseados em Inteligência Computacional.

A seguir são citadas as publicações do autor obtidas durante o mestrado. Foram publicados dois artigos decorrentes diretamente deste trabalho (1 e 2) e outros dois em parceria com outros pesquisadores do grupo de pesquisa Lab. EaSII (3 e 4). Além disso, o resultado final desta pesquisa será enviado para o *Knowledge-Based Systems Journal* (5).

1. Oliveira, P.; Souza, M.; Braga, R.; Brito, R.; Lira Rabelo, R. e Neto, P. S. **Athena: A Visual Tool to Support the Development of Computational Intelligence Systems**. 26th IEEE International Conference on Tools with Artificial Intelligence (ICTAI). 2014.

2. Braga R.; Oliveira, P. A.; Souza, M.; Neto, P. S.; Rabêlo, R. e Britto, B. **Ferramentas para Desenvolvimento de Sistemas Baseados em Inteligência Computacional: Um Mapeamento Sistemático**. XII Simpósio Brasileiro de Automação Inteligente (SBAI). 2015.

3. Oliveira, P.; Costa, D. A.; Souza, M. e Santos Neto, P. **SELF: an Easy Way to Perform Acceptance Testing**. XIII Simpósio Brasileiro de Qualidade de Software (SBQS). 2014.

4. Souza, M.; Oliveira, P.; Ribeiro, R. F.; Brito, R. e Santos Neto, P. **UseSkill: uma ferramenta de apoio à avaliação de usabilidade de sistemas Web**. XIV Simpósio Brasileiro de Qualidade de Software (SBQS). 2015.

5. Oliveira, P.; Souza, M.; Braga, R.; Brito, R.; Lira Rabelo, R. e Neto, P. S. **Athena: A Visual Tool for Supporting the Development of Computational Intelligence Systems - Extended version**. Knowledge-Based Systems Journal (em desenvolvimento).

6.1 Desafios e Limitações

Apesar dos resultados obtidos, esta pesquisa ainda apresenta alguns desafios e limitações. A maior delas refere-se aos mecanismos de adaptação ao problema embutidos na ferramenta. Conforme apresentado na Seção 4.2, a versão atual da Athena dispõe de módulos aptos a solucionar diversas classes de problemas: problema da mochila mono-objetivo e multiobjetivo, otimização combinatória com variáveis binárias, classificação,

regressão e agrupamento de dados e caixeiro viajante. Entretanto, a ferramenta só lida com tipos primitivos (lógico, inteiro, real, matriz e texto) e não é permitido ao usuário a construção de novos tipos de dados. Além disso, a Athena dispõe de poucos mecanismos para auxiliar a definição de heurísticas baseadas em população e não há nenhuma padronização que facilite a definição de estruturas de vizinhança para algoritmos de busca.

Essa limitação reduz o leque de aplicações da ferramenta, portanto faz-se necessário futuros investimentos na ampliação desses recursos. Existem outras classes de problemas que devem ser incluídos, por exemplo, problema de atribuição linear ou quadrática, problema de roteamento de veículos, dentre outros. A Tabela 5 do trabalho proposto por PAREJO et al. (2012) apresenta uma série de características que podem ser incorporadas à Athena para suprir a necessidade de uma melhor adequação ao problema.

A formulação das funções objetivo e de restrições também precisa ser aprimorada. Atualmente, a Athena oferece um mecanismo, baseado em uma gramática de expressões matemáticas, que permite a definição de funções simples, tais como as apresentadas na Tabela 19. No entanto, alguns problemas exigem funções mais complexas e, por isso, é interessante incorporar uma Linguagem de Domínio Específico (DSL) (DEURSEN; KLINT; VISSER, 2000), a fim de facilitar a construção de tais funções.

Outra limitação da Arquitetura Proposta refere-se ao fato de que ela não está adaptada a determinados algoritmos de pré-processamento de dados, por exemplo, remoção de atributos utilizando o Ganho de Informação (QUINLAN, 1986). Nesse caso, não é possível determinar as variáveis de saída antes da execução. Esse quesito merece uma investigação detalhada para aprimorar a AP e possibilitar a resolução de problemas de Aprendizagem de Máquina.

Um grande desafio em relação à Arquitetura Proposta diz respeito ao princípio da alta performance. A arquitetura prescreve a utilização do modelo de Infraestrutura como Serviço (IaaS) para possibilitar o ajuste dos recursos computacionais (processamento e memória) necessários para a execução dos experimentos. Entretanto, não está claro como o usuário poderá acessar essa função e quem irá arcar com os custos. Não existem também restrições explícitas quanto à forma como os dados de entrada e saída devem ser tratados. Dessa forma, um módulo pode acabar gerando um consumo excessivo de processamento e memória. Além disso, a obtenção dos dados de entrada por meio do *upload* tradicional de arquivos pode ser bastante lenta. Esse processo pode ser otimizado por meio da integração com outras ferramentas, como Google Drive e Dropbox. Por fim, ressalta-se a importância da realização de uma avaliação rigorosa sobre a performance da Athena. Somente após esse estudo haverá comprovações científicas de que o desempenho da proposta é adequado à aplicação em contextos que exigem muita eficiência no processamento e armazenamento de dados.

A avaliação da ferramenta também pode ser vista como uma limitação, pois não

foi possível realizar um Estudo de Caso (WOHLIN et al., 2012) dentro de uma empresa. Não foram encontradas empresas com o perfil adequado para esse tipo de estudo e que estivessem dispostas a participar desta pesquisa. Além disso, o experimento controlado foi realizado apenas com alunos (mestrado e graduação) e a avaliação de usabilidade foi superficial. Esses fatores são limitações deste trabalho. Entretanto, apesar de todos esses fatores, a Athena deve ser adotada pelos pesquisadores do departamento de Pesquisa e Desenvolvimento (P&D) da empresa Infoway[1], devido ao seu potencial de crescimento.

Finalmente, pode-se notar que mesmo com todo o trabalho realizado até o momento, ainda há muito espaço para melhorias e trabalhos futuros. Dessa forma, tais limitações representam novas oportunidades de trabalho.

6.2 Trabalhos Futuros

Destacam-se algumas ideias para trabalhos futuros:

- **Desenvolvimento de novos módulos**: atualmente a Athena dispõe de 35 módulos, entretanto é possível incorporar uma infinidade de algoritmos de diversas subáreas da Inteligência Artificial, além de mecanismos auxiliares. Exemplos: algoritmos de Aprendizagem de Máquina (*e.g.*, BayesNet, Kstar, SVM, NaiveBayes), pré-processamento de dados (*e.g.*, adicionar atributo, discretização, randomização), balanceamento de classes (*e.g.*, Resample, SMOTE), algoritmo baseados em trajetória (*e.g.*, busca local, busca tabu, *Simulated Annealing*), outros algoritmos de IC (*e.g.*, *Fuzzy* Tipo 2, FastPGA, *Multiobjective Max-Min Ant System* (M3AS));

- **Athena Marketplace**: a proposta deste trabalho prescreve a construção de módulos a partir da reunião de outros módulos (ver Capítulo 3). Isso permite a navegabilidade em diferentes níveis de abstração. Dessa forma, um sistema complexo (composto por vários módulos) pode ser abstraído dentro de um módulo, reduzindo sua complexidade para o usuário. Esse conceito pode ser ampliado para permitir que esse sistema se torne um serviço dentro da Athena. Por exemplo, o sistema de IC para estimação do tamanho das AgNPs (ver Seção 4.4) poderia ser convertido em um serviço disponível para todos os usuários da ferramenta. Além disso, as configurações poderiam ser pré-definidas durante a construção do serviço, o que facilitaria sua utilização por pesquisadores de outras áreas. Essa ideia é a base para a construção de um Athena *Marketplace*, similar ao Google *Marketplace*, oferecendo diversos tipos de serviços baseados em Inteligência Computacional;

- **Novas funcionalidades**: a Athena foi desenvolvida para ser um ambiente capaz de facilitar o desenvolvimento de Sistemas Inteligentes, entretanto, para que esse

[1] Infoway website - http://infoway-br.com/.

ambiente se torne auto-suficiente, faz-se necessário o desenvolvimento de novas funcionalidades como: projeto de experimentos (característica A2.5 da Tabela 6), análises estatísticas (característica A2.6 da Tabela 6), inclusão de hiper-heurísticas, execuções distribuídas, etc. Essas novas funções, aliadas ao fortalecimento da política de testes automatizados, certamente têm potencial para colocar a Athena entre as ferramentas mais importantes dessa linha de pesquisa;

- **Revisão Sistemática**: o Capítulo 2 apresenta um mapeamento sistemático realizado para identificar as ferramentas que oferecem apoio à construção de Sistemas Baseados em Inteligência Computacional. Entretanto, esse estudo tem caráter quantitativo e não qualitativo. Dessa forma, seria interessante estender o mapeamento para criar uma revisão sistemática, na qual as ferramentas seriam avaliadas de forma qualitativa (KEELE, 2007);

- **Estudo de Usabilidade**: a facilidade de uso pode definir o sucesso ou fracasso de uma aplicação (FERNANDEZ; INSFRAN; ABRAHÃO, 2011). A Seção 5.12 apresenta uma pequena avaliação de usabilidade realizada com dados coletados na execução do experimento controlado. Essa avaliação apontou uma série de problemas que foram corrigidos para aprimorar a utilização da ferramenta, entretanto ainda há espaço para um estudo de usabilidade mais criterioso. Uma ferramenta que pode auxiliar nessa atividade é a UseSkill (SOUZA et al., 2015);

- **Realização de novos estudos empíricos**: este trabalho apresenta duas avaliações empíricas: um experimento controlado e uma avaliação preliminar sobre a performance da Athena. Os resultados desses experimentos são conclusivos quanto a adequabilidade da Athena no apoio ao desenvolvimento de Sistemas Inteligentes em ambiente acadêmico. Entretanto, essa conclusão não pode ser generalizada para o ambiente industrial com 100% de certeza. Dessa forma, ressalta-se a importância da realização de duas novas avaliações: um Estudo de Caso (WOHLIN et al., 2012) dentro de uma empresa e uma avaliação rigorosa sobre o desempenho ferramenta. Outro trabalho futuro interessante seria a replicação do experimento controlado em outros instituições de ensino para solidificar ainda mais os resultados obtidos;

Por fim, como a Athena permanecerá disponível nos servidores da Universidade Federal do Piauí (UFPI) por meio do link http://easii.ufpi.br/athena e outros pesquisadores continuarão a realizar estudos em torno dessa proposta, ressalta-se a possibilidade de inúmeros trabalhos futuros com a utilização da ferramenta no desenvolvimento de outras pesquisas científicas.

Referências

AGRAWAL, R.; SRIKANT, R. Fast algorithms for mining association rules. In: *Proceedings of the 20th International Conference Very Large Data Bases (VLDB)*. [S.l.: s.n.], 1994. v. 1215, p. 487–499. Citado na página 69.

AICKELIN, U.; CAYZER, S. The danger theory and its application to artificial immune systems. *Proceedings of the 1th International Conference on Artificial Immune Systems*, 2008. Citado na página 20.

ALBA, E. et al. Mallba: A library of skeletons for combinatorial optimisation. In: *Euro-Par Parallel Processing*. [S.l.]: Springer Berlin Heidelberg, 2002, (Lecture Notes in Computer Science, v. 2400). p. 927–932. Citado 3 vezes nas páginas 6, 30 e 97.

ALLIANCE, O. *Osgi service platform, release 3*. [S.l.]: IOS Press, Inc., 2003. Citado 2 vezes nas páginas 63 e 151.

AMDAHL, G. M. Validity of the single processor approach to achieving large scale computing capabilities. In: ACM. *Proceedings of the Spring Joint Computer Conference*. [S.l.], 1967. p. 483–485. Citado na página 156.

ANCHIETA, R. T. et al. Using stylometric features for sentiment classification. In: *Computational Linguistics and Intelligent Text Processing*. [S.l.]: Springer International Publishing, 2015, (Lecture Notes in Computer Science, v. 9042). p. 189–200. Citado na página 5.

ANDERSON, R. L. Recent advances in finding best operating conditions. *Journal of the American Statistical Association*, Taylor & Francis Group, v. 48, n. 264, p. 789–798, 1953. Citado na página 18.

ANNALURU, R.; DAS, S.; PAHWA, A. Multi-level ant colony algorithm for optimal placement of capacitors in distribution systems. In: IEEE. *Congress on Evolutionary Computation*. [S.l.], 2004. v. 2, p. 1932–1937. Citado na página 19.

ARAGAO, A. et al. Uma aplicação de redes neurais artificiais e otimização por enxame de partículas para estimativa do tamanho de nanopartículas de prata. *XII Simpósio Brasileiro de Automação Inteligente (SBAI)*, 2015. Citado 4 vezes nas páginas 16, 77, 79 e 89.

ARMBRUST, M. et al. A view of cloud computing. *Communications of the ACM*, ACM, v. 53, n. 4, p. 50–58, 2010. Citado na página 153.

BACK, T.; FOGEL, D. B.; MICHALEWICZ, Z. *Handbook of evolutionary computation*. [S.l.]: IOP Publishing Ltd., 1997. Citado na página 18.

BAZGAN, C.; HUGOT, H.; VANDERPOOTEN, D. Solving efficiently the 0–1 multi-objective knapsack problem. *Computers & Operations Research*, Elsevier, v. 36, n. 1, p. 260–279, 2009. Citado na página 71.

BAZTERRA, V. E.; FERRARO, M. B.; FACELLI, J. C. Modified genetic algorithm to model crystal structures. i. benzene, naphthalene and anthracene. *The Journal of Chemical Physics*, AIP Publishing, v. 116, n. 14, p. 5984–5991, 2002. Citado na página 5.

BENDALE, A. et al. Visionblocks: A social computer vision framework. In: IEEE. *3th International Conference on Social Computing (SocialCom)*. [S.l.], 2011. p. 521–526. Citado na página 8.

BENI, G.; WANG, J. Swarm intelligence in cellular robotic systems. In: *Proceedings NATO Advanced Workshop on Robotics and Biological Systems*. Il Ciocco, Tuscany, Italy: [s.n.], 1989. Citado na página 4.

BEZDEK, J. . Intelligence: Computational versus artificial. *IEEE Transactions on Neural Networks*, v. 4, n. 5, p. 737, 1993. Citado na página 3.

BEZDEK, J. C. What is computational intelligence? *Computational Intelligence: Imitating Life*, Piscataway, NJ: IEEE Press, p. 1–12, 1994. Citado na página 3.

BEZDEK, J. C.; EHRLICH, R.; FULL, W. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, Elsevier, v. 10, n. 2, p. 191–203, 1984. Citado 2 vezes nas páginas 21 e 68.

BLUM, C.; ROLI, A. Hybrid metaheuristics: an introduction. In: *Hybrid Metaheuristics*. [S.l.]: Springer, 2008. p. 1–30. Citado na página 7.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. *UML: guia do usuário*. [S.l.]: Elsevier Brasil, 2006. Citado na página 59.

BRAGA, R. et al. Ferramentas para desenvolvimento de sistemas baseados em inteligência computacional: Um mapeamento sistemático. *XII Simpósio Brasileiro de Automação Inteligente (SBAI)*, 2015. Citado 3 vezes nas páginas 6, 9 e 10.

BRAZ-JUNIOR, G. et al. Classification of breast tissues using moran's index and geary's coefficient as texture signatures and svm. *Computers in Biology and Medicine*, v. 39, n. 12, p. 1063 – 1072, 2009. Citado na página 4.

BREIMAN, L. *Classification and regression trees*. [S.l.]: Chapman & Hall/CRC, 1984. Citado na página 69.

BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001. Citado na página 69.

BREMERMANN, H. J. Optimization through evolution and recombination. *Self-organizing systems*, Washington, DC, v. 93, p. 106, 1962. Citado na página 18.

BRIDGES, S. M.; VAUGHN, R. B. Fuzzy data mining and genetic algorithms applied to intrusion detection. In: *Proceedings of 12th Annual Canadian Information Technology Security Symposium*. [S.l.: s.n.], 2000. p. 109–122. Citado na página 22.

BRITTO, R. et al. A hybrid approach to solve the agile team allocation problem. In: *IEEE Congress on Evolutionary Computation (CEC), 2012, Brisbane*. [S.l.: s.n.], 2012. p. 1–8. Citado 7 vezes nas páginas 5, 18, 80, 81, 82, 89 e 104.

CAHON, S.; MELAB, N.; TALBI, E. G. Paradiseo: A framework for the reusable design of parallel and distributed metaheuristics. *Journal of Heuristics*, Springer, v. 10, n. 3, p. 357–380, 2004. Citado na página 30.

CAMELO, J.; RABELO, R. L.; PASSOS, E. Automatic mapping between gameplay and aesthetics rpg character attributes through fuzzy system. In: *Brazilian Symposium on Computer Games and Digital Entertainment (SBGAMES)*. [S.l.: s.n.], 2014. p. 220–229. Citado 2 vezes nas páginas 6 e 22.

CARDIE, C. Using decision trees to improve case-based learning. In: *Proceedings of the 10th International Conference on Machine Learning*. [S.l.: s.n.], 1993. p. 25–32. Citado na página 23.

CARDOSO, J.; VALETTE, R. *Redes de Petri*. [S.l.]: Editora da UFSC, 1997. Citado na página 62.

CARO, G. D.; DORIGO, M. Antnet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, p. 317–365, 1998. Citado na página 19.

CASTILLO, O.; MELIN, P. Computational intelligence software: Type-2 fuzzy logic and modular neural networks. In: IEEE. *International Joint Conference on Neural Networks*. [S.l.], 2008. p. 1820–1827. Citado na página 30.

CASTO, L. Nunes de; ZUBEN, F. J. V. An evolutionary immune network for data clustering. In: IEEE. *Proceedings of 6th Brazilian Symposium on Neural Networks*. [S.l.], 2000. p. 84–89. Citado na página 20.

CASTRO, L. N. D. *Fundamentals of natural computing: basic concepts, algorithms, and applications*. [S.l.]: CRC Press, 2006. Citado 3 vezes nas páginas 18, 20 e 36.

CASTRO, L. N. D.; ZUBEN, F. J. V. The clonal selection algorithm with engineering applications. In: *Proceedings of the Genetic and Evolutionary Computational Conference*. [S.l.: s.n.], 2000. p. 36–39. Citado na página 20.

CASTRO, L. N. D.; ZUBEN, F. J. V. Learning and optimization using the clonal selection principle. *Transactions on Evolutionary Computation*, IEEE, v. 6, n. 3, p. 239–251, 2002. Citado na página 69.

CASTRO, L. N. de; TIMMIS, J. An artificial immune network for multimodal function optimization. In: IEEE. *Proceedings of the Congress on Evolutionary Computation*. [S.l.], 2002. v. 1, p. 699–704. Citado 2 vezes nas páginas 4 e 20.

CASTRO, L. N. de; ZUBEN, F. J. V. Ainet: an artificial immune network for data analysis. *Data mining: a heuristic approach*, Idea Group Publishing, USA, v. 1, p. 231–259, 2001. Citado na página 20.

CASTRO, L. N. de; ZUBEN, F. J. V. An immunological approach to initialize centers of radial basis function neural networks. In: CITESEER. *Proceedings of the 5th Brazilian Conference on Neural Networks*. [S.l.], 2001. p. 79–84. Citado na página 20.

CASTRO, L. N. de; ZUBEN, F. J. V. An immunological approach to initialize feedforward neural network weights. In: SPRINGER. *Artificial Neural Nets and Genetic Algorithms*. [S.l.], 2001. p. 126–129. Citado na página 20.

CAYZER, S.; AICKELIN, U. A recommender system based on idiotypic artificial immune networks. *Journal of Mathematical Modelling and Algorithms*, Springer, v. 4, n. 2, p. 181–198, 2005. Citado na página 20.

CHELLAPILLA, K.; FOGEL, D. B. Evolution, neural networks, games, and intelligence. *Proceedings of the IEEE*, IEEE, v. 87, n. 9, p. 1471–1496, 1999. Citado na página 18.

CHEN, G.-C.; JUANG, C.-F. Object detection using color entropies and a fuzzy classifier. *Computational Intelligence Magazine*, IEEE, v. 8, n. 1, p. 33–45, 2013. Citado 2 vezes nas páginas 5 e 22.

CHEN, S.-C.; YEN, D. C.; HWANG, M. I. Factors influencing the continuance intention to the usage of web 2.0: An empirical study. *Computers in Human Behavior*, Elsevier, v. 28, n. 3, p. 933–941, 2012. Citado na página 36.

CHO, J. et al. Modeling and inverse controller design for an unmanned aerial vehicle based on the self-organizing map. *IEEE Transactions on Neural Networks*, v. 17, n. 2, p. 445–460, March 2006. Citado na página 5.

CHUAN, Z. et al. A lvq-based neural network anti-spam email approach. *SIGOPS Operating Systems Review*, ACM, New York, NY, USA, v. 39, n. 1, p. 34–39, jan 2005. Citado na página 5.

CINGOLANI, P.; ALCALÁ-FDEZ, J. jfuzzylogic: a java library to design fuzzy logic controllers according to the standard for fuzzy control programming. *International Journal of Computational Intelligence Systems*, Taylor & Francis, v. 6, n. sup1, p. 61–75, 2013. Citado na página 104.

CLEMENTS, P. et al. *Documenting software architectures: views and beyond*. [S.l.]: Pearson Education, 2002. Citado na página 51.

COAKLEY, J. R.; BROWN, C. E. Artificial neural networks in accounting and finance: Modeling issues. *International Journal of Intelligent Systems in Accounting, Finance & Management*, John Wiley & Sons, Ltd., v. 9, n. 2, p. 119–144, 2000. Citado na página 17.

COFFEY, P. *Benchmarking the Amazon Elastic Compute Cloud (EC2)*. Tese (Doutorado) — Worcester Polytechnic Institute, 2011. Citado 2 vezes nas páginas 153 e 157.

COLE, N.; LOUIS, S.; MILES, C. Using a genetic algorithm to tune first-person shooter bots. In: *Congress on Evolutionary Computation (CEC)*. [S.l.: s.n.], 2004. v. 1, p. 139–145 Vol.1. Citado na página 6.

COOK, T. D.; CAMPBELL, D. T.; DAY, A. *Quasi-experimentation: Design & analysis issues for field settings*. [S.l.]: Houghton Mifflin Boston, 1979. v. 351. Citado na página 100.

CRAMER, N. L. A representation for the adaptive generation of simple sequential programs. In: *Proceedings of the First International Conference on Genetic Algorithms*. [S.l.: s.n.], 1985. p. 183–187. Citado na página 18.

CROCKFORD, D. The application/json media type for javascript object notation (json). 2006. Citado na página 64.

DARWIN, C. *Origin of Species*. [S.l.]: DMP, 1978. Citado na página 18.

DASGUPTA, D. Computational intelligence in cyber security. In: *IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety (CIHSPS 2006)*. [S.l.: s.n.], 2006. p. 2–3. Citado na página 5.

DASGUPTA, D.; FORREST, S. An anomaly entection algorithm inspired by the immune syste. In: *Artificial immune systems and their applications*. [S.l.]: Springer, 1999. p. 262–277. Citado na página 20.

DEB, K. et al. A fast and elitist multiobjective genetic algorithm: Nsga-II. *IEEE Transactions on Evolutionary Computation*, v. 6, n. 2, p. 182–197, 2002. Citado na página 68.

DEURSEN, A. V.; KLINT, P.; VISSER, J. Domain-specific languages: An annotated bibliography. *Sigplan Notices*, v. 35, n. 6, p. 26–36, 2000. Citado na página 110.

DILLON, T.; WU, C.; CHANG, E. Cloud computing: issues and challenges. In: IEEE. *24th International Conference on Advanced Information Networking and Applications (AINA)*. [S.l.], 2010. p. 27–33. Citado na página 10.

DIXON, W. J.; JR, F. J. M. Introduction to statistical analysis. McGraw-Hill, 1957. Citado 3 vezes nas páginas 95, 97 e 99.

DORIGO, M.; MANIEZZO, V.; COLORNI, A. The Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, v. 26, n. 1, p. 29–41, 1996. Citado 2 vezes nas páginas 19 e 68.

DORNE, R.; VOUDOURIS, C. Hsf: A generic framework to easily design meta-heuristic methods. In: CITESEER. *4th Metaheuristics International Conference (MI'2001), Porto, Portugal*. [S.l.], 2001. p. 423–428. Citado na página 30.

DU, K. L. Clustering: A neural network approach. *Neural Networks*, Elsevier, v. 23, n. 1, p. 89–107, 2010. Citado na página 17.

DUCH, W.; MANDZIUK, J. *Challenges for computational intelligence*. [S.l.]: Springer Science & Business Media, 2007. v. 63. Citado na página 3.

DURILLO, J. J.; NEBRO, A. J. jmetal: A java framework for multi-objective optimization. *Advances in Engineering Software*, Elsevier, v. 42, n. 10, p. 760–771, 2011. Citado na página 30.

DURILLO, J. J.; NEBRO, A. J.; ALBA, E. The jmetal framework for multi-objective optimization: Design and architecture. In: *Congress on Evolutionary Computation (CEC)*. [S.l.]: IEEE, 2010. p. 1–8. Citado 2 vezes nas páginas 10 e 104.

EASON, G.; NOBLE, B.; SNEDDON, I. On certain integrals of lipschitz-hankel type involving products of bessel functions. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, The Royal Society, v. 247, n. 935, p. 529–551, 1955. Citado na página 77.

EBERHART, R. C. *Computational Intelligence: Concepts to Implementations*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007. Citado 3 vezes nas páginas 3, 15 e 68.

EBERT, C. An approach to fuzzy data analysis for software quality control. In: *Proc 1st European Congress on Fuzzy and Intelligent Technologies EUFIT-95, Aachen, Germany*. [S.l.: s.n.], 1993. p. 1156–1161. Citado 2 vezes nas páginas 5 e 22.

EL-FAKI, M. S.; ZHANG, N.; PETERSON, D. E. Weed detection using color machine vision. *Transactions of the ASAE-American Society of Agricultural Engineers*, [St. Joseph, Mich., American Society of Agricultural Engineers], 1958-c2005., v. 43, n. 6, p. 1969–1978, 2000. Citado na página 5.

ELGAALI, E.; GARCIA, L.; OJIMA, D. Sensitivity of irrigation water balance to climate change in the great plains of colorado. *Transactions of the ASAE*, v. 49, n. 5, p. 1315–1322, 2006. Citado na página 5.

ELLEN, R. A.; CAMPBELL, D. A. A framework for executing computational intelligence over distributed embedded nodes. In: *The IASTED International Conference on Computational Intelligence*. [S.l.]: ACTA Press, 2005. p. 129–134. Citado 3 vezes nas páginas 30, 40 e 55.

ENGELBRECHT, A. P. *Computational Intelligence: An Introduction*. [S.l.]: Wiley Publishing, 2007. Citado 10 vezes nas páginas 4, 6, 15, 17, 18, 19, 20, 21, 34 e 41.

FAVILLA, J.; MACHION, A.; GOMIDE, F. Fuzzy traffic control: adaptive strategies. In: IEEE. *2th International Conference on Fuzzy Systems*. [S.l.], 1993. p. 506–511. Citado na página 22.

FERNANDEZ, A.; INSFRAN, E.; ABRAHÃO, S. Usability evaluation methods for the web: A systematic mapping study. *Information and Software Technology*, Elsevier, v. 53, n. 8, p. 789–817, 2011. Citado 2 vezes nas páginas 36 e 112.

FERREIRA, A. B. d. H. *Novo dicionário da língua portuguesa*. [S.l.]: Nova Fronteira, 1986. Citado na página 5.

FIELDING, R. et al. *Hypertext transfer protocol–HTTP/1.1*. [S.l.], 1999. Citado na página 56.

FIGUEIRA, M. M. C. Identificação de outliers. *Millenium*, Instituto Politécnico de Viseu, 1998. Citado na página 96.

FINK, A.; VOSS, S. Hotframe: A heuristic optimization framework. In: *Optimization Software Class Libraries*. [S.l.]: Springer, 2002. p. 81–154. Citado na página 30.

FISHER, D. H. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, Springer, v. 2, n. 2, p. 139–172, 1987. Citado na página 69.

FLACH, P. A.; LACHICHE, N. *The Tertius system*. 1999. Citado na página 69.

FLANAGAN, D. *JavaScript: The Definitive Guide*. [S.l.]: O'reilly, 2011. Citado na página 64.

FOGEL, L. J.; OWENS, A. J.; WALSH, M. J. *Artificial Intelligence Through Simulated Evolution*. [S.l.]: John Wiley, 1966. Citado na página 18.

FORREST, S. et al. Self-nonself discrimination in a computer. In: IEEE. *Proceedings of the Symposium on Research in Security and Privacy*. [S.l.], 1994. p. 202–212. Citado na página 20.

FRAIN, B. *Responsive Web Design with HTML5 and CSS3*. [S.l.]: Packt Publishing Ltd, 2012. Citado na página 64.

FRANKLIN, B.; BERGERMAN, M. Cultural algorithms: Concepts and experiments. In: IEEE. *Proceedings of the Congress on Evolutionary Computation*. [S.l.], 2000. v. 2, p. 1245–1251. Citado na página 18.

FRASER, A. S. Simulation of genetic systems by automatic digital computers. ii. effects of linkage on rates of advance under selection. *Australian Journal of Biological Science*, v. 10, p. 492–499, 1957. Citado 2 vezes nas páginas 4 e 17.

FRASER, A. S. Simulation of genetic systems by automatic digital computers. *Australian Journal of Biological Sciences*, CSIRO, v. 13, n. 2, p. 150–162, 1960. Citado na página 18.

FREITAS, L. B. d. L.; SILVEIRA, P. G.; PIETA, M. A. M. Um estudo sobre o desenvolvimento da gratidão na infância. *Interamerican Journal of Psychology*, Sociedad Interamericana de Psicología, v. 43, n. 1, p. 49–56, 2009. Citado na página 5.

FRIEDBERG, R. M. A learning machine: Part i. *IBM Journal of Research and Development*, IBM, v. 2, n. 1, p. 2–13, 1958. Citado na página 18.

FUKUYAMA, Y.; YOSHIDA, H. A particle swarm optimization for reactive power and voltage control in electric power systems. In: IEEE. *Proceedings of the Congress on Evolutionary Computation*. [S.l.], 2001. v. 1, p. 87–93. Citado na página 19.

GAMMA, E. et al. *Design patterns: elements of reusable object-oriented software*. [S.l.]: Pearson Education, 1994. Citado 3 vezes nas páginas 51, 56 e 64.

GANGULY, S.; SAHOO, N.; DAS, D. Multi-objective expansion planning of electrical distribution networks using comprehensive learning particle swarm optimization. In: *Applications of Soft Computing*. [S.l.]: Springer, 2009. p. 193–202. Citado na página 5.

GARLAN, D. Software architecture: a roadmap. In: ACM. *Proceedings of the Conference on the Future of Software Engineering*. [S.l.], 2000. p. 91–101. Citado na página 51.

GIBBONS, J. D.; CHAKRABORTI, S. *Nonparametric statistical inference*. [S.l.]: Springer, 2011. Citado na página 97.

GONZALEZ, F.; DASGUPTA, D.; KOZMA, R. Combining negative selection and classification techniques for anomaly detection. In: IEEE. *Proceedings of the Congress on Evolutionary Computation*. [S.l.], 2002. v. 1, p. 705–710. Citado na página 20.

GOONATILAKE, S.; KHEBBAL, S. *Intelligent hybrid systems*. [S.l.]: John Wiley & Sons, Inc., 1994. Citado na página 23.

GOSLING, J. *The Java language specification*. [S.l.]: Addison-Wesley Professional, 2000. Citado na página 63.

GREENSMITH, J.; AICKELIN, U.; CAYZER, S. Introducing dendritic cells as a novel immune-inspired algorithm for anomaly detection. In: *Artificial Immune Systems*. [S.l.]: Springer, 2005. p. 153–167. Citado na página 20.

HALL, M. et al. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, ACM, v. 11, n. 1, p. 10–18, 2009. Citado 3 vezes nas páginas 10, 38 e 41.

HARMAN, M. The current state and future of search based software engineering. In: IEEE. *Future of Software Engineering*. [S.l.], 2007. p. 342–357. Citado na página 80.

HARMAN, M. et al. Search based software engineering: Techniques, taxonomy, tutorial. In: *Empirical Software Engineering and Verification*. [S.l.]: Springer, 2012. p. 1–59. Citado 2 vezes nas páginas 71 e 80.

HARTIGAN, J. A.; WONG, M. A. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, JSTOR, v. 28, n. 1, p. 100–108, 1979. Citado na página 69.

HAYKIN, S. *Redes neurais*. [S.l.]: Bookman, 2001. Citado 2 vezes nas páginas 17 e 68.

HAYKIN, S. S. et al. *Neural networks and learning machines*. [S.l.]: Pearson Education Upper Saddle River, 2009. v. 3. Citado na página 16.

HEBB, D. O. The organization of behavior. *Brain research bulletin*, Elsevier, 1949. Citado na página 15.

HERNANDES, E. et al. Using gqm and tam to evaluate start-a tool that supports systematic review. *CLEI Electronic Journal*, Centro Latinoamericano de Estudios en Informática, v. 15, n. 1, p. 3–3, 2012. Citado na página 28.

HIGGINS, J. P.; GREEN, S. et al. *Cochrane handbook for systematic reviews of interventions*. [S.l.]: Wiley Online Library, 2008. v. 5. Citado na página 25.

HOFMANN, M.; KLINKENBERG, R. *RapidMiner: Data mining use cases and business analytics applications*. [S.l.]: CRC Press, 2013. Citado na página 10.

HOLLAND, J. H. *Adaptation in Natural and Artificial Systems*. [S.l.]: MIT Press, 1975. Citado na página 18.

HOPFIELD, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, National Acad Sciences, v. 79, n. 8, p. 2554–2558, 1982. Citado na página 4.

HUNT, J. E.; COOKE, D. E. Learning using an artificial immune system. *Journal of network and computer applications*, Elsevier, v. 19, n. 2, p. 189–212, 1996. Citado na página 20.

IBA, H.; IWATA, M.; HIGUCHI, T. Gate-level evolvable hardware: Empirical study and application. In: *Evolutionary Algorithms in Engineering Applications*. [S.l.]: Springer, 1997. p. 259–276. Citado 2 vezes nas páginas 5 e 18.

IEEE. Ieee standard glossary of software engineering terminology. *IEEE Std 610.12-1990*, p. 1–84, Dec 1990. Citado na página 71.

JADEJA, Y.; MODI, K. Cloud computing-concepts, architecture and challenges. In: IEEE. *International Conference on Computing, Electronics and Electrical Technologies (ICCEET)*. [S.l.], 2012. p. 877–880. Citado na página 10.

JAMSA, K. A. *Cloud Computing: SaaS, PaaS, IaaS, Virtualization, Business Models, Mobile, Security and More*. [S.l.]: Jones & Bartlett Publishers, 2012. Citado 2 vezes nas páginas 57 e 62.

JEONG, J.; OH, S.-Y. Automatic rule generation for fuzzy logic controllers using rule-level co-evolution of subpopulations. In: IEEE. *Proceedings of the Congress on Evolutionary Computation*. [S.l.], 1999. v. 3. Citado na página 22.

JIN, X.; REYNOLDS, R. G. Mining knowledge in large scale databases using cultural algorithms with constraint handling mechanisms. In: IEEE. *Proceedings of the Congress on Evolutionary Computation*. [S.l.], 2000. v. 2, p. 1498–1506. Citado na página 18.

JOHNSON, J.; SUGISAKA, M. Complexity science for the design of swarm robot control systems. In: *26th Annual Confjerence of the Industrial Electronics Society*. [S.l.: s.n.], 2000. v. 1, p. 695–700 vol.1. Citado na página 18.

JUN, J.-H.; LEE, D.-W.; SIM, K.-B. Realization of cooperative strategies and swarm behavior in distributed autonomous robotic systems using artificial immune system. In: IEEE. *Proceedings of International Conference on Systems, Man and Cybernetics*. [S.l.], 1999. v. 6, p. 614–619. Citado na página 20.

KARABOGA, D.; BASTURK, B. Artificial bee colony (abc) optimization algorithm for solving constrained optimization problems. In: *Foundations of Fuzzy Logic and Soft Computing*. [S.l.]: Springer, 2007. p. 789–798. Citado na página 19.

KASPER, M. Shape optimization by evolution strategy. *Transactions on Magnetics*, IEEE, v. 28, n. 2, p. 1556–1560, 1992. Citado na página 18.

KEELE, S. Guidelines for performing systematic literature reviews in software engineering. In: *Technical report (EBSE)*. [S.l.: s.n.], 2007. Citado 3 vezes nas páginas 25, 37 e 112.

KENNEDY, J.; EBERHART, R. C. Particle swarm optimization. IEEE, p. 1942–1948, 1995. Citado na página 19.

KEWLEY, R. H.; EMBRECHTS, M. J. Computational military tactical planning system. *Transactions on Systems, Man, and Cybernetics*, IEEE, v. 32, n. 2, p. 161–171, 2002. Citado na página 18.

KHAN, J. et al. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature medicine*, Nature Publishing Group, v. 7, n. 6, p. 673–679, 2001. Citado 3 vezes nas páginas 4, 8 e 17.

KITCHENHAM, B.; PICKARD, L.; PFLEEGER, S. L. Case studies for method and tool evaluation. *IEEE Software*, v. 12, n. 4, p. 52–62, 1995. Citado na página 25.

KNIGHT, T.; TIMMIS, J. A multi-layered immune inspired approach to data mining. In: NOTTINGHAM, UK. *Proceedings of the 4th International Conference on Recent Advances in Soft Computing*. [S.l.], 2002. p. 266–271. Citado na página 20.

KOHONEN, T. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, Springer, v. 43, n. 1, p. 59–69, 1982. Citado na página 17.

KOHONEN, T.; SOMERVUO, P. Self-organizing maps of symbol strings. *Neurocomputing*, Elsevier, v. 21, n. 1, p. 19–30, 1998. Citado na página 68.

KONAR, A. *Computational Intelligence: Principles, Techniques and Applications*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005. Citado 2 vezes nas páginas 3 e 15.

KONDO, D. et al. Cost-benefit analysis of cloud computing versus desktop grids. In: IEEE. *International Symposium on Parallel & Distributed Processing*. [S.l.], 2009. p. 1–12. Citado na página 157.

KOSKO, B. Fuzzy function approximation. In: *International Joint Conference on Neural Networks*. [S.l.: s.n.], 1992. v. 1, p. 209–213 vol.1. Citado na página 22.

KRASNOGOR, N.; SMITH, J. Mafra: A java memetic algorithms framework. 2000. Citado na página 30.

KRONFELD, M.; PLANATSCHER, H.; ZELL, A. The eva2 optimization framework. In: *Learning and Intelligent Optimization*. [S.l.]: Springer, 2010. p. 247–250. Citado na página 30.

LACKOVIĆ, M. An approach to optical network design using general heuristic optimization framework. *Journal of Information and Organizational Sciences*, Fakultet organizacije i informatike Sveučilišta u Zagrebu, v. 34, n. 2, p. 175–194, 2010. Citado na página 30.

LAHSASNA, A. et al. Design of a fuzzy-based decision support system for coronary heart disease diagnosis. *Journal of medical systems*, Springer, v. 36, n. 5, p. 3293–3306, 2012. Citado 2 vezes nas páginas 5 e 22.

LAKATOS, E. M.; MARCONI, M. de A. *Metodologia científica*. [S.l.]: Atlas São Paulo, 1991. Citado na página 87.

LANZA-GUTIERREZ, J. M. et al. Optimizing energy consumption in heterogeneous wireless sensor networks by means of evolutionary algorithms. In: *Applications of Evolutionary Computation*. [S.l.]: Springer Berlin Heidelberg, 2012, (Lecture Notes in Computer Science, v. 7248). p. 1–10. Citado na página 18.

LEE, Y. C.; TAHERI, J.; ZOMAYA, A. Y. A parallel metaheuristic framework based on harmony search for scheduling in distributed computing systems. *International Journal of Foundations of Computer Science*, World Scientific, v. 23, n. 02, p. 445–464, 2012. Citado na página 30.

LEON, C.; MIRANDA, G.; SEGURA, C. Metco: a parallel plugin-based framework for multi-objective optimization. *International Journal on Artificial Intelligence Tools*, World Scientific, v. 18, n. 04, p. 569–588, 2009. Citado na página 38.

LIENIG, J. Physical design of vlsi circuits and the application of genetic algorithms. In: *Evolutionary Algorithms in Engineering Applications*. [S.l.]: Springer, 1997. p. 277–292. Citado na página 5.

LIMA, B. V. A. de; MACHADO, V. P. Machine learning algorithms applied in automatic classification of social network users. In: *2012 Fourth International Conference on Computational Aspects of Social Networks (CASoN)*. [S.l.: s.n.], 2012. Citado na página 5.

LÓPEZ-CAMACHO, E. et al. jmetalcpp: optimizing molecular docking problems with a c++ metaheuristic framework. *Bioinformatics*, Oxford Univ Press, p. btt679, 2013. Citado na página 30.

LUGER, G. F. *Inteligência Artificial: estruturas e estratégias para a solução de problemas complexos*. [S.l.]: Bookman, 2004. Citado na página 3.

MACHADO, V.; NETO, A. D. D.; MELO, J. D. D. A neural network multiagent architecture applied to industrial networks for dynamic allocation of control strategies using standard function blocks. *IEEE Transactions on Industrial Electronics*, IEEE, v. 57, n. 5, p. 1823–1834, 2010. Citado na página 5.

MAGELE, C. et al. Higher order evolution strategies for the global optimization of electromagnetic devices. *Transactions on Magnetics*, IEEE, v. 29, n. 2, p. 1775–1778, 1993. Citado na página 18.

MAMDANI, E. H. Application of fuzzy logic to approximate reasoning using linguistic synthesis. *IEEE Transactions on Computers*, v. 26, n. 12, p. 1182–1191, 1977. Citado 2 vezes nas páginas 21 e 68.

MARSTON, S. et al. Cloud computing—the business perspective. *Decision Support Systems*, Elsevier, v. 51, n. 1, p. 176–189, 2011. Citado na página 153.

MATZINGER, P. Tolerance, danger, and the extended family. *Annual review of immunology*, v. 12, n. 1, p. 991–1045, 1994. Citado na página 20.

MAYRHAUSER, A. von; ANDERSON, C.; MRAZ, R. Using a neural network to predict test case effectiveness. In: *Proceedings of IEEE Aerospace Applications Conference*. [S.l.: s.n.], 1995. p. 77–91 vol.2. Citado na página 5.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, Springer, v. 5, n. 4, p. 115–133, 1943. Citado 2 vezes nas páginas 15 e 16.

MEKSANGSOUY, P.; CHAIYARATANA, N. Dna fragment assembly using an ant colony system algorithm. In: IEEE. *Congress on Evolutionary Computation*. [S.l.], 2003. v. 3, p. 1756–1763. Citado na página 19.

MERKL, D. Structuring software for reuse-the case of self-organizing maps. In: *IJCNN '93-Nagoya. Proceedings of International Joint Conference on Neural Networks, 1993.* [S.l.: s.n.], 1993. v. 3, p. 2468–2471 vol.3. Citado na página 5.

MICHEL, A. N.; FARRELL, J. A. Associative memories via artificial neural networks. *IEEE Control Systems Magazine*, IEEE, v. 10, n. 3, p. 6–17, 1990. Citado na página 17.

MIERSWA, I. et al. Yale: Rapid prototyping for complex data mining tasks. In: ACM. *Proceedings of the 12th International Conference on Knowledge Discovery and Data Mining*. [S.l.], 2006. p. 935–940. Citado 2 vezes nas páginas 38 e 41.

MILANO, M.; ROLI, A. Magma: a multiagent architecture for metaheuristics. *Transactions on Systems, Man, and Cybernetics*, IEEE, v. 34, n. 2, p. 925–941, 2004. Citado na página 30.

MITCHELL, T. *Machine Learning*. [S.l.]: McGraw-Hill, 1997. Citado na página 69.

MOONEY, R. J.; OURSTON, D. A multistrategy approach to theory refinement. *Machine Learning IV: A multistrategy approach*, Morgan Kauffmann, v. 4, p. 141–164, 1994. Citado na página 23.

MORA, A. M.; SQUILLERO, G. Applications of evolutionary computation. Springer, 2015. Citado na página 18.

MOREIRA, P. M.; REIS, L. P.; SOUSA, A. A. de. i-om: Intelligent optimization: For computer graphics and visualization. In: IEEE. *5th Iberian Conference on Information Systems and Technologies (CISTI)*. [S.l.], 2010. p. 1–6. Citado na página 30.

MYLLYMÄKI, P. et al. B-course: A web-based tool for bayesian and causal data analysis. *International Journal on Artificial Intelligence Tools*, World Scientific, v. 11, n. 03, p. 369–387, 2002. Citado na página 38.

NAIR, L. S.; LAURENCIN, C. T. Silver nanoparticles: synthesis and therapeutic applications. *Journal of biomedical nanotechnology*, American Scientific Publishers, v. 3, n. 4, p. 301–316, 2007. Citado na página 77.

NEETHLING, M.; ENGELBRECHT, A. P. Determining rna secondary structure using set-based particle swarm optimization. In: IEEE. *Congress on Evolutionary Computation*. [S.l.], 2006. p. 1670–1677. Citado na página 19.

NGAI, E. W.; WAT, F. Fuzzy decision support system for risk analysis in e-commerce development. *Decision support systems*, Elsevier, v. 40, n. 2, p. 235–255, 2005. Citado na página 22.

NILSSON, N. J. *Artificial intelligence: a new synthesis*. [S.l.]: Morgan Kaufmann, 1998. Citado na página 3.

OLIVEIRA, A. de; LORENA, L. A constructive genetic algorithm for gate matrix layout problems. *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, IEEE, v. 21, n. 8, p. 969–974, Aug 2002. Citado na página 5.

OLIVEIRA, P. et al. Athena: A visual tool to support the development of computational intelligence systems. In: *IEEE 26th International Conference on Tools with Artificial Intelligence (ICTAI)*. [S.l.: s.n.], 2014. p. 950–959. Citado 3 vezes nas páginas 8, 9 e 30.

OMKAR, S. N.; SENTHILNATH, J. Neural network and swarm intelligence for data mining. In: _____. *Integration of Swarm Intelligence and Artificial Neural Network*. [S.l.]: World Scientific, 2011. cap. 2, p. 23–65. Citado na página 5.

OMRAN, M.; SALMAN, A.; ENGELBRECHT, A. P. Image classification using particle swarm optimization. In: SINGAPORE. *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning*. [S.l.], 2002. v. 1, p. 18–22. Citado na página 19.

OSTROWSKI, D. A.; REYNOLDS, R. G. Knowledge-based software testing agent using evolutionary learning with cultural algorithms. In: IEEE. *Proceedings of the Congress on Evolutionary Computation*. [S.l.], 1999. v. 3. Citado na página 18.

PAMPARA, G.; ENGELBRECHT, A. P.; CLOETE, T. Cilib: A collaborative framework for computational intelligence algorithms - part i. In: *International Joint Conference on Neural Networks (IJCNN)*. [S.l.]: IEEE, 2008. p. 1750–1757. Citado 6 vezes nas páginas 6, 10, 30, 40, 55 e 97.

PAREJO, J. et al. Fom: A framework for metaheuristic optimization. In: *International Conference on Computational Science (ICCS)*. [S.l.]: Springer, 2003. p. 886–895. Citado na página 30.

PAREJO, J. A. et al. Metaheuristic optimization frameworks: a survey and benchmarking. *Soft Computing*, Springer, v. 16, n. 3, p. 527–561, 2012. Citado 6 vezes nas páginas 7, 23, 41, 43, 45 e 110.

PARKER, R. G.; RARDIN, R. L. *Discrete optimization*. [S.l.]: Elsevier, 2014. Citado na página 68.

PARSONS, S.; JONES, G. Acoustic identification of twelve species of echolocating bat by discriminant function analysis and artificial neural networks. *Journal of Experimental Biology*, The Company of Biologists Ltd, v. 203, n. 17, p. 2641–2656, 2000. Citado na página 5.

PATRICK, M. et al. Subdomain-based test data generation. *Journal of Systems and Software*, v. 103, p. 328 – 342, 2015. Citado na página 5.

PAVLIDIS, N. G. et al. Computational intelligence methods for financial time series modeling. *International Journal of Bifurcation and Chaos*, v. 16, n. 07, p. 2053–2062, 2006. Citado na página 5.

PEDRYCZ, W.; SOSNOWSKI, Z. Fuzzy object-oriented system design. *Fuzzy Sets and Systems*, v. 99, n. 2, p. 121 – 134, 1998. Citado na página 5.

PEER, E. S. *A Serendipitous Software Framework for Facilitating Collaboration in Computational iIntelligence*. [S.l.], 2004. Citado 2 vezes nas páginas 6 e 97.

PENA-REYES, C. A.; SIPPER, M. Applying fuzzy coco to breast cancer diagnosis. In: IEEE. *Proceedings of the IEEE Congress on Evolutionary Computation*. [S.l.], 2000. v. 2, p. 1168–1175. Citado na página 22.

PETERS, T. K.; KORALEWSKI, H.-E.; ZERBST, E. W. The evolution strategy - a search strategy used in individual optimization of electrical parameters for therapeutic carotid sinus nerve stimulation. *Transactions on Biomedical Engineering*, IEEE, v. 36, n. 7, p. 668–675, 1989. Citado na página 18.

PETERSEN, K. et al. Systematic mapping studies in software engineering. In: BRITISH COMPUTER SOCIETY SWINTON, UK. *12th International Conference on Evaluation and Assessment in Software Engineering*. [S.l.], 2008. v. 17. Citado 3 vezes nas páginas 15, 25 e 26.

PETRI, C. A. Kommunikation mit automaten. 1962. Citado na página 62.

PETTICREW, M.; ROBERTS, H. *Systematic reviews in the social sciences: A practical guide*. [S.l.]: John Wiley & Sons, 2008. Citado na página 26.

PRAMANIK, S.; KOZMA, R.; DASGUPTA, D. Dynamical neuro-representation of an immune model and its application for data classification. In: IEEE. *Proceedings of the International Joint Conference on Neural Networks*. [S.l.], 2002. v. 1, p. 130–135. Citado na página 20.

PRESSMAN, R. S. *Engenharia de Software*. [S.l.]: McGraw Hill Brasil, 2011. Citado 2 vezes nas páginas 41 e 71.

PRESSMAN, R. S.; JAWADEKAR, W. S. Software engineering. *New York 1992*, 1987. Citado na página 51.

QUINLAN, J. R. Induction of decision trees. *Machine learning*, Springer, v. 1, n. 1, p. 81–106, 1986. Citado na página 110.

QUINLAN, J. R. Simplifying decision trees. *International Journal of Man-Machine Studies*, Elsevier, v. 27, n. 3, p. 221–234, 1987. Citado na página 69.

QUINLAN, J. R. *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993. Citado na página 69.

RABELO, R.; CARNEIRO, A.; BRAGA, R. An energetic operation policy using fuzzy controllers for maximization of benefits in the brazilian hydrothermal power system. In: IEEE. *IEEE Bucharest PowerTech*. [S.l.], 2009. p. 1–7. Citado 2 vezes nas páginas 5 e 22.

RECHENBERG, I. Cybernetic solution path of an experimental problem. Ministry of Aviation, Royal Aircraft Establishment, 1965. Citado na página 68.

REINALDO, F. et al. Multi-strategy learning made easy. In: *10th WSEAS International Conference on Computers*. [S.l.: s.n.], 2006. Citado na página 30.

REYNOLDS, R.; AL-SHEHRI, H. The use of cultural algorithms with evolutionary programming to guide decision tree induction in large databases. In: IEEE. *Proceedings of the International Conference on Evolutionary Computation (World Congress on Computational Intelligence)*. [S.l.], 1998. p. 541–546. Citado na página 18.

REZENDE, S. O. *Sistemas Inteligentes: fundamentos e aplicações*. [S.l.]: Editora Manole Ltda, 2003. Citado 4 vezes nas páginas 7, 21, 22 e 23.

RICH, E.; KNIGHT, K. Artificial intelligence. *McGraw-Hill, New*, 1991. Citado na página 3.

RICHARDSON, L.; RUBY, S. *RESTful web services*. [S.l.]: "O'Reilly Media, Inc.", 2008. Citado na página 64.

ROSENBERG, F. et al. Metaheuristic optimization of large-scale qos-aware service compositions. In: IEEE. *International Conference on Services Computing (SCC)*. [S.l.], 2010. p. 97–104. Citado na página 30.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958. Citado 2 vezes nas páginas 16 e 68.

ROSIN, C. D. *Coevolutionary search among adversaries*. Tese (Doutorado) — University of California at San Diego, 1997. Citado na página 18.

ROSS, T. J. *Fuzzy logic with engineering applications*. [S.l.]: John Wiley & Sons, 2009. Citado 2 vezes nas páginas 21 e 68.

RUAN, D.; KACPRZYK, J.; FEDRIZZI, M. *Soft computing for risk evaluation and management: Applications in technology, environment and finance*. [S.l.]: Springer Science & Business Media, 2001. v. 76. Citado na página 5.

RUMELHART, D. E. et al. *Parallel distributed processing*. [S.l.]: IEEE, 1988. v. 1. Citado 2 vezes nas páginas 16 e 68.

RUSSELL, S.; NORVIG, P. *Inteligência Artificial*. [S.l.]: Elsevier, 2013. v. 3. Citado na página 15.

SALMAN, A.; AHMAD, I.; AL-MADANI, S. Particle swarm optimization for task assignment problem. *Microprocessors and Microsystems*, Elsevier, v. 26, n. 8, p. 363–371, 2002. Citado na página 19.

SANTOS-NETO, P. et al. Regression testing prioritization based on fuzzy inference systems. In: *SEKE*. [S.l.]: Knowledge Systems Institute Graduate School, 2012. p. 273–278. Citado 3 vezes nas páginas 5, 23 e 104.

SARLIJA, N.; BENSIC, M.; ZEKIC-SUSAC, M. A neural network classification of credit applicants in consumer credit scoring. In: *Artificial Intelligence and Applications*. [S.l.: s.n.], 2006. p. 205–210. Citado na página 5.

SCHOLZ, F. W.; STEPHENS, M. A. K-sample anderson–darling tests. *Journal of the American Statistical Association*, Taylor & Francis Group, v. 82, n. 399, p. 918–924, 1987. Citado 2 vezes nas páginas 97 e 100.

SCHWEFEL, H.-P. *Evolutionsstrategie und numerische Optimierung*. Tese (Doutorado) — Technical University Berlin, 1975. Citado 2 vezes nas páginas 18 e 68.

SECREST, B. R. *Traveling salesman problem for surveillance mission using particle swarm optimization*. [S.l.], 2001. Citado na página 19.

SEIFERT, E. Originpro 9.1: Scientific data analysis and graphing software - software review. *Journal of chemical information and modeling*, ACS Publications, v. 54, n. 5, p. 1552–1552, 2014. Citado na página 97.

SELVATHI, D.; SELVARAJ, H.; SELVI, S. T. Hybrid approach for brain tumor segmentation in magnetic resonance images using cellular neural networks and optimization techniques. *International Journal of Computational Intelligence and Applications*, World Scientific, v. 9, n. 01, p. 17–31, 2010. Citado na página 5.

SHAR, L.; BRIAND, L.; TAN, H. Web application vulnerability prediction using hybrid program analysis and machine learning. *IEEE Transactions on Dependable and Secure Computing*, PP, n. 99, p. 1–1, 2014. Citado na página 5.

SHAW, M. Writing good software engineering research papers: minitutorial. In: IEEE. *Proceedings of the 25th international conference on software engineering*. [S.l.], 2003. p. 726–736. Citado na página 34.

SHI, Y.; EBERHART, R. C. Fuzzy adaptive particle swarm optimization. In: IEEE. *Proceedings of the Congress on Evolutionary Computation*. [S.l.], 2001. v. 1, p. 101–106. Citado na página 22.

SHIRANI, H. et al. Determining the features influencing physical quality of calcareous soils in a semiarid region of iran using a hybrid pso-dt algorithm. *Geoderma*, Elsevier, v. 259, p. 1–11, 2015. Citado na página 5.

SILVA, I. N. da; SPATTI, D. H.; FLAUZINO, R. A. *Redes Neurais Artificiais para engenharia e ciências aplicadas curso prático*. [S.l.]: Artliber Editora, 2010. Citado 2 vezes nas páginas 3 e 68.

SILVA, R. et al. Automatic detection of motorcyclists without helmet. In: *2013 XXXIX Latin American Computing Conference (CLEI)*. [S.l.: s.n.], 2013. p. 1–7. Citado 3 vezes nas páginas 5, 16 e 17.

SOUSA, T.; NEVES, A.; SILVA, A. Swarm optimisation as a new tool for data mining. In: IEEE. *Proceedings of International Parallel and Distributed Processing Symposium*. [S.l.], 2003. Citado na página 19.

SOUZA, M. et al. Useskill: uma ferramenta de apoio à avaliação de usabilidade de sistemas web. *XIV Simpósio Brasileiro de Qualidade de Software (SBQS)*, 2015. Citado 3 vezes nas páginas 91, 103 e 112.

SREEPATHI, S.; MAHINTHAKUMAR, G. K. Optimus: A parallel optimization framework with topology aware pso and applications. In: IEEE. *SC Companion: High Performance Computing, Networking, Storage and Analysis (SCC)*. [S.l.], 2012. p. 1524–1525. Citado na página 30.

STEWART, D. B.; VOLPE, R. A.; KHOSLA, P. K. Design of dynamically reconfigurable real-time software using port-based objects. *Transactions on Software Engineering*, IEEE, v. 23, n. 12, p. 759–776, 1997. Citado na página 40.

STOCKT, S. Van der. *A Generic Neural Network Framework Using Design Patterns*. Tese (Doutorado) — University of Pretoria, 2007. Citado na página 6.

STORN, R. On the usage of differential evolution for function optimization. In: IEEE. *Proceedings of the Biennial Conference of the North American Fuzzy Information Processing Society*. [S.l.], 1996. p. 519–523. Citado na página 18.

STORN, R.; PRICE, K. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, Springer, v. 11, n. 4, p. 341–359, 1997. Citado na página 18.

STÜTZLE, T.; HOOS, H. H. Max–min ant system. *Future generation computer systems*, Elsevier, v. 16, n. 8, p. 889–914, 2000. Citado 3 vezes nas páginas 68, 71 e 73.

SUGENO, M.; KANG, G. Structure identification of fuzzy model. *Fuzzy sets and systems*, Elsevier, v. 28, n. 1, p. 15–33, 1988. Citado na página 21.

SUI, R.; THOMASSON, J. Ground-based sensing system for cotton nitrogen status determination. *Transactions of the ASABE*, American Society of Agricultural Engineers, v. 49, n. 6, p. 1983–1991, 2006. Citado na página 5.

TAKAGI, T.; SUGENO, M. Fuzzy identification of systems and its applications to modeling and control. *Transactions on Systems, Man and Cybernetics*, IEEE, n. 1, p. 116–132, 1985. Citado na página 21.

TALBI, E.-G. A taxonomy of hybrid metaheuristics. *Journal of heuristics*, Kluwer Academic Publishers, v. 8, n. 5, p. 541–564, 2002. Citado 2 vezes nas páginas 7 e 23.

TANSCHEIT, R. Sistemas fuzzy. *Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro*, 2004. Citado 3 vezes nas páginas 15, 21 e 22.

THAMMA, P.; BARAI, S. Prediction of compressive strength of cement using gene expression programming. In: *Applications of Soft Computing*. [S.l.]: Springer, 2009. p. 203–212. Citado na página 5.

TIAN, H.; SHANG, Z. Artificial neural network as a classification method of mice by their calls. *Ultrasonics*, Elsevier, v. 44, p. e275–e278, 2006. Citado na página 5.

TIMMIS, J. *Artificial immune systems: a novel data analysis technique inspired by the immune network theory*. Tese (Doutorado) — University of Wales, Aberystwyth, 2000. Citado na página 20.

TIMMIS, J.; NEAL, M.; HUNT, J. Data analysis using artificial immune systems, cluster analysis and kohonen networks: some comparisons. In: IEEE. *Proceedings of International Conference on Systems, Man and Cybernetics*. [S.l.], 1999. v. 3, p. 922–927. Citado na página 20.

TRAVASSOS, G. H.; GUROV, D.; AMARAL, E. *Introdução à engenharia de software experimental*. [S.l.]: UFRJ, 2002. Citado 2 vezes nas páginas 89 e 91.

TURING, A. M. Intelligent machinery, a heretical theory. *The Turing Test: Verbal Behavior as the Hallmark of Intelligence*, MIT Press, p. 105, 1948. Citado 2 vezes nas páginas 3 e 15.

VENKATARAMAN, P. *Applied optimization with MATLAB programming*. [S.l.]: John Wiley & Sons, 2009. Citado na página 10.

VERAS, R. et al. Assessing the accuracy of macula detection methods in retinal images. In: IEEE. *18th International Conference on Digital Signal Processing (DSP)*. [S.l.], 2013. p. 1–6. Citado na página 5.

VICENTE, B. G. L. Z.; CEZARE, M. J.; SILVA, I. N. Controlador neural de marcha lenta para motores de combustão interna. *VIII Simpósio Brasileiro de Automação Inteligente*, 2007. Citado na página 17.

VOUDOURIS, C. et al. iopt: A software toolkit for heuristic search methods. In: SPRINGER. *7th International Conference on Principles and Practice of Constraint Programming - CP 2001*. [S.l.], 2001. p. 716–729. Citado na página 30.

WAGNER, S.; AFFENZELLER, M. *Heuristiclab: A generic and extensible optimization environment*. [S.l.]: Springer, 2005. Citado 2 vezes nas páginas 30 e 40.

WAGNER, S. et al. Architecture and design of the heuristiclab optimization environment. In: *Advanced Methods and Applications in Computational Intelligence*. [S.l.]: Springer International Publishing, 2014, (Topics in Intelligent Engineering and Informatics, v. 6). p. 197–261. Citado na página 7.

WATKINS, A.; BOGGESS, L. A resource limited artificial immune classifier. In: *Proceedings of the Congress on Evolutionary Computation (CEC)*. [S.l.: s.n.], 2002. v. 1, p. 926–931. Citado na página 69.

WEYLAND, D.; MONTEMANNI, R.; GAMBARDELLA, L. M. A metaheuristic framework for stochastic combinatorial optimization problems based on gpgpu with a case study on the probabilistic traveling salesman problem with deadlines. *Journal of Parallel and Distributed Computing*, Elsevier, v. 73, n. 1, p. 74–85, 2013. Citado na página 30.

WIDROW, B.; HOFF, M. Adaptive switching circuits. Defense Technical Information Center, 1960. Citado 2 vezes nas páginas 16 e 68.

WIERINGA, R. et al. Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. *Requirements Engineering*, Springer, v. 11, n. 1, p. 102–107, 2006. Citado na página 33.

WILCOXON, F. Individual comparisons by ranking methods. *Biometrics bulletin*, JSTOR, v. 1, n. 6, p. 80–83, 1945. Citado na página 97.

WOHLIN, C. et al. *Experimentation in software engineering*. [S.l.]: Springer, 2012. Citado 8 vezes nas páginas 87, 89, 92, 100, 106, 110, 112 e 157.

WOLPERT, D. H.; MACREADY, W. G. No free lunch theorems for optimization. *Transactions on Evolutionary Computation*, IEEE, v. 1, n. 1, p. 67–82, 1997. Citado na página 47.

YOO, S.; HARMAN, M. Pareto efficient multi-objective test case selection. In: *Proceedings of the International Symposium on Software Testing and Analysis*. New York, NY, USA: ACM, 2007. (ISSTA '07), p. 140–150. Citado na página 71.

ZADEH, L. A. Fuzzy sets. *Information and Control*, v. 8, p. 338–353, 1965. Citado 2 vezes nas páginas 4 e 21.

ZADEH, L. A. Fuzzy logic and approximate reasoning. *Synthese*, Springer, v. 30, n. 3-4, p. 407–428, 1975. Citado na página 21.

ZADEH, L. A. Fuzzy logic, neural networks, and soft computing. *Communications of the ACM*, ACM, v. 37, n. 3, p. 77–84, 1994. Citado 3 vezes nas páginas 3, 7 e 22.

ZADEH, L. A. Roles of soft computing and fuzzy logic in the conception, design and deployment of information/intelligent systems. In: *Computational intelligence: soft computing and fuzzy-neuro integration with applications*. [S.l.]: Springer, 1998. p. 1–9. Citado na página 4.

ZHA, X. F. Soft computing framework for intelligent human–machine system design, simulation and optimization. *Soft Computing*, Springer, v. 7, n. 3, p. 184–198, 2003. Citado na página 30.

ZHANG, Q.; STANLEY, S. J. Real-time water treatment process control with artificial neural networks. *Journal of Environmental Engineering*, American Society of Civil Engineers, v. 125, n. 2, p. 153–160, 1999. Citado 2 vezes nas páginas 5 e 17.

ZHANG, Y.; HARMAN, M.; MANSOURI, S. A. The multi-objective next release problem. In: *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*. New York, NY, USA: ACM, 2007. (GECCO '07), p. 1129–1137. Citado na página 5.

ZHANG, Z.; FRIEDRICH, K. Artificial neural networks applied to polymer composites: a review. *Composites Science and technology*, Elsevier, v. 63, n. 14, p. 2029–2044, 2003. Citado na página 5.

ZHU, T.; TSO, S.; LO, K. Wavelet-based fuzzy reasoning approach to power-quality disturbance recognition. *IEEE Transactions on Power Delivery*, v. 19, n. 4, p. 1928–1935, Oct 2004. Citado na página 5.

ZITZLER, E. et al. *SPEA2: Improving the strength Pareto evolutionary algorithm.* [S.l.]: Eidgenössische Technische Hochschule Zürich (ETH), Institut für Technische Informatik und Kommunikationsnetze (TIK) Zürich, Switzerland, 2001. Citado na página 68.

Apêndices

APÊNDICE A – *Strings* de Pesquisa

Scopus: *TITLE-ABS-KEY("Computational Intelligence Framework" OR "Artificial Intelligence Framework" OR "Metaheuristic Framework" OR "Soft Computing Framework" OR "Computational Intelligence tool" OR "Artificial Intelligence tool" OR "Metaheuristic tool" OR "Soft Computing tool" OR "Computational Intelligence service" OR "Artificial Intelligence service" OR "Metaheuristic service" OR "Soft Computing service" OR "Computational Intelligence library" OR "Artificial Intelligence library" OR "Metaheuristic library" OR "Soft Computing library" OR "Computational Intelligence api" OR "Artificial Intelligence api" OR "Metaheuristic api" OR "Soft Computing api" OR "metaheuristic optimization framework" OR "metaheuristic optimization tool" OR "metaheuristic optimization library" OR "metaheuristic optimization service" OR "metaheuristic optimization api" OR "heuristic optimization framework" OR "heuristic optimization tool" OR "heuristic optimization library" OR "heuristic optimization service" OR "heuristic optimization api" OR "framework for computational intelligence" OR "framework for artificial intelligence" OR "framework for soft computing" OR "framework for Metaheuristic"OR "tool for computational intelligence"OR "tool for artificial intelligence" OR "tool for soft computing" OR "tool for Metaheuristic" OR "service for computational intelligence" OR "service for artificial intelligence" OR "service for soft computing" OR "service for Metaheuristic" OR "api for computational intelligence" OR "api for artificial intelligence" OR "api for soft computing" OR "api for Metaheuristic").*

Compendex: *(("Computational Intelligence Framework" OR "Artificial Intelligence Framework" OR "Metaheuristic Framework" OR "Soft Computing Framework" OR "Computational Intelligence tool" OR "Artificial Intelligence tool" OR "Metaheuristic tool" OR "Soft Computing tool" OR "Computational Intelligence service" OR "Artificial Intelligence service" OR "Metaheuristic service" OR "Soft Computing service" OR "Computational Intelligence library" OR "Artificial Intelligence library" OR "Metaheuristic library" OR "Soft Computing library" OR "Computational Intelligence api" OR "Artificial Intelligence api" OR "Metaheuristic api" OR "Soft Computing api" OR "metaheuristic optimization framework" OR "metaheuristic optimization tool" OR "metaheuristic optimization library" OR "metaheuristic optimization service" OR "metaheuristic optimization api" OR "heuristic optimization framework" OR "heuristic optimization tool" OR "heuristic optimization library" OR "heuristic optimization service" OR "heuristic optimization api" OR "framework for computational intelligence" OR "framework for artificial intelligence" OR "framework for soft computing" OR "framework for Metaheuristic"OR "tool for computational intelligence"OR "tool for artificial intelligence" OR "tool for soft computing" OR "tool for Metaheuristic" OR "service for computational intelligence" OR "service for*

artificial intelligence" OR "service for soft computing" OR "service for Metaheuristic" OR "api for computational intelligence" OR "api for artificial intelligence" OR "api for soft computing" OR "api for Metaheuristic") WN AB).

Web of Science: *TS=("Computational Intelligence Framework" OR "Artificial Intelligence Framework" OR "Metaheuristic Framework" OR "Soft Computing Framework" OR "Computational Intelligence tool" OR "Artificial Intelligence tool" OR "Metaheuristic tool" OR "Soft Computing tool" OR "Computational Intelligence service" OR "Artificial Intelligence service" OR "Metaheuristic service" OR "Soft Computing service" OR "Computational Intelligence library" OR "Artificial Intelligence library" OR "Metaheuristic library" OR "Soft Computing library" OR "Computational Intelligence api" OR "Artificial Intelligence api" OR "Metaheuristic api" OR "Soft Computing api" OR "metaheuristic optimization framework" OR "metaheuristic optimization tool" OR "metaheuristic optimization library" OR "metaheuristic optimization service" OR "metaheuristic optimization api" OR "heuristic optimization framework" OR "heuristic optimization tool" OR "heuristic optimization library" OR "heuristic optimization service" OR "heuristic optimization api" OR "framework for computational intelligence" OR "framework for artificial intelligence" OR "framework for soft computing" OR "framework for Metaheuristic"OR "tool for computational intelligence"OR "tool for artificial intelligence" OR "tool for soft computing" OR "tool for Metaheuristic" OR "service for computational intelligence" OR "service for artificial intelligence" OR "service for soft computing" OR "service for Metaheuristic" OR "api for computational intelligence" OR "api for artificial intelligence" OR "api for soft computing" OR "api for Metaheuristic").*

APÊNDICE B – Avaliação Comparativa dos Trabalhos

Tabela 34 – Cobertura das funcionalidades pertencentes à área *Técnicas de IA* (1)

ID	Funcionalidade	Peso	*AFRANCI*	*Athena*	*HeuristicLab*	*I-om*	*IT2FLS*	*Optimus*	*CILib*	*Ellen*	*EvA2*	*FOM*	*HSF*	*HOTFRAME.*	*iOpt*	*jMetal*
A1.1.1	Perceptron/Adaline	0,250	✓	✓												
A1.1.2	Perceptron Multicamadas	0,250	✓	✓	✓		✓		✓	✓						
A1.1.3	Redes de funções de base radial	0,250		✓												
A1.1.4	Redes auto-organizáveis	0,250		✓												
A1.2.1	Algoritmo genético	0,143		✓	✓	✓			✓		✓	✓	✓	✓	✓	✓
A1.2.2	Estratégia evolutiva	0,143		✓	✓						✓					✓
A1.2.3	Programação genética	0,143			✓						✓					
A1.2.4	Programação evolutiva	0,143														
A1.2.5	Evolução diferencial	0,143									✓					✓
A1.2.6	Evolução cultural	0,143														
A1.2.7	Coevolução	0,143							✓							
A1.3.1	PSO	0,250			✓			✓	✓		✓					✓
A1.3.2	PSO discreto	0,250							✓							✓
A1.3.3	Ant System	0,125		✓												
A1.3.4	Ant Colony System	0,125			✓				✓							
A1.3.5	Max-Min Ant System	0,125		✓												
A1.3.6	Ant System Rank	0,125														
A1.4.1	CLONALG	0,334		✓					✓							
A1.4.2	optIA	0,334														
A1.4.3	aiNet	0,334														
A1.5.1	Sistemas de Inferência Fuzzy	0,500		✓					✓	✓						
A1.5.2	Fuzzy Tipo II	0,167					✓									
A1.5.3	Fuzzy C-Means	0,167		✓												
A1.5.4	Algoritmos Neuro-Fuzzy	0,167								✓						
A1.6.1	Steepest descent/hill climbing	0,167			✓	✓					✓	✓	✓			
A1.6.2	Simulated annealing	0,167			✓	✓					✓	✓	✓	✓	✓	
A1.6.3	Tabu search	0,167			✓							✓	✓	✓	✓	
A1.6.4	Scatter search	0,167			✓						✓					✓
A1.6.5	GRASP	0,167										✓				
A1.6.6	Variable neighborhood search (VNS)	0,167			✓							✓				
A1.7.1	Apriori	0,112		✓												
A1.7.2	Tertius	0,112		✓												
A1.7.3	HotSpot	0,112														
A1.7.4	Redes de Bayes	0,055														
A1.7.5	C4.5	0,055	✓	✓												
A1.7.6	KStar	0,055														
A1.7.7	NaiveBayes	0,055														
A1.7.8	RandomForest	0,055		✓	✓											
A1.7.9	RandomTree	0,055		✓												
A1.7.10	Cobweb	0,167		✓												
A1.7.11	K-Means	0,167		✓	✓				✓							
A1.8.1	PGA	0,062														✓
A1.8.2	MOGA	0,062									✓					
A1.8.3	NSGA	0,062									✓					
A1.8.4	NSGA-II	0,062		✓	✓						✓					✓
A1.8.5	NPGA	0,062														
A1.8.6	SPEA	0,062									✓					
A1.8.7	SPEA-II	0,062		✓							✓					✓
A1.8.8	PAES	0,062														✓
A1.8.9	PESA	0,062									✓					
A1.8.10	PESA-II	0,062									✓					
A1.8.11	MOMGA	0,062														
A1.8.12	ARMOGA	0,062														
A1.8.13	*Multobjective Ant System*	0,062														
A1.8.14	*Multiobjective PSO*	0,062														✓
A1.8.15	POSA	0,062														
A1.8.16	MOSA	0,062														

Tabela 35 – Cobertura das funções pertencentes à área *Suporte ao processo de otimização* (1)

ID	Funcionalidade	Peso	*AFRANCI*	*Athena*	*HeuristicLab*	*I-om*	*IT2FLS*	*Optimus*	*CILib*	*Ellen*	*EvA2*	*FOM*	*HSF*	*HOTFRAME*	*iOpt*	*jMetal*
A2.1.1	Diferentes classes de problemas	0,125	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
A2.1.2	Pré-processamento dos dados de entrada	0,125	✓				✓									
A2.1.3	Codificação da solução	0,125		✓	✓	✓			✓		✓	✓	✓	✓	✓	✓
A2.1.4	Definição da estrutura de vizinhança	0,125			✓								✓	✓	✓	
A2.1.5	Construção de heurísticas baseadas em população	0,125		✓	✓	✓			✓		✓		✓	✓	✓	✓
A2.1.6	Estratégias para seleção de soluções	0,125			✓	✓			✓		✓	✓	✓	✓	✓	✓
A2.1.7	Especificação de funções objetivo	0,125		✓	✓	✓			✓				✓	✓	✓	✓
A2.1.8	Definição e controle de restrições	0,125		✓		✓			✓		✓	✓	✓	✓	✓	✓
A2.2.1	Hibridização	0,334	✓	✓	✓				✓	✓		✓	✓		✓	
A2.2.2	Hiper-heurísticas	0,334										✓				
A2.2.3	Execução paralela e distribuída	0,334	✓	✓		✓		✓		✓	✓					
A2.3.1	Número máximo de iterações	0,167	✓	✓	✓	✓		✓	✓		✓	✓	✓	✓	✓	✓
A2.3.2	Valor específico obtido pela função objetivo	0,167		✓	✓	✓			✓		✓	✓	✓	✓		✓
A2.3.3	Tempo de execução	0,167		✓	✓				✓		✓	✓	✓	✓		
A2.3.4	Combinação logica entre condições de parada	0,500							✓		✓	✓				
A2.4.1	Repetição automática de tarefas simples	0,250		✓	✓				✓		✓	✓				
A2.4.2	Automatização de tarefas variando os parâmetros de execução	0,250		✓					✓							
A2.4.3	Repetição automática de diferentes tarefas	0,250			✓											
A2.4.4	Automatização de diferentes tarefas em diferentes problemas	0,250														
A2.5.1	Definição das Hipóteses	0,167										✓				
A2.5.2	Modelagem do experimento (especificação das variáveis)	0,167										✓				
A2.5.3	Desenho do experimento (Fatorial, quadrado latino, etc.)	0,167										✓				
A2.5.4	Auxílio na execução do experimento	0,167										✓				
A2.5.5	Geração do plano de execução	0,167			✓							✓				
A2.5.6	Importar/Exportar experimentos	0,167														
A2.6.1	T-Student	0,100														
A2.6.2	ANOVA	0,100										✓				
A2.6.3	Wilcoxon	0,100										✓				
A2.6.4	Mann-Whitnet	0,100										✓				
A2.6.5	Kolmogorov-Smirnov	0,100														
A2.6.6	Importar/Exportar análises estatísticas	0,500														
A2.7.1	Design	0,167	✓	✓	✓		✓				✓	✓		✓	✓	
A2.7.2	Configuração das técnicas de IA	0,167	✓	✓	✓		✓		✓	✓	✓	✓		✓	✓	
A2.7.3	Modelagem do problema	0,167			✓						✓					
A2.7.4	Auxílio no planejamento de tarefas de otimização	0,167		✓	✓											
A2.7.5	Suporte a definição de diferentes projetos	0,167	✓	✓	✓						✓					
A2.7.6	Representação gráfica dos resultados	0,167	✓	✓	✓		✓				✓	✓				
A2.8.1	Importar dados	0,250	✓	✓	✓		✓				✓					
A2.8.2	Exportar dados	0,250	✓		✓						✓					
A2.8.3	*Web Service*	0,250		✓		✓										
A2.8.4	Manipulação de projetos por meio de XML ou JSON	0,250		✓	✓				✓							

Tabela 36 – Cobertura das funcionalidades pertencentes à área *Técnicas de IA* (2)

ID	Funcionalidade	Peso	*jMetalCpp*	*MAFRA*	*MAGMA*	*MALLBA*	*Nyx*	*ParadisEO*	*PMF*	*Rosemberg*	*Weyland*	*Zha*	*METCO*	*B-Course*	*WEKA*	*RapidMiner*	*Google Prediction API*	*TensorFlow*	*Azure ML Studio*
A1.1.1	Perceptron/Adaline	0,250										✓				✓			
A1.1.2	Perceptron Multicamadas	0,250										✓			✓	✓		✓	✓
A1.1.3	Redes de funções de base radial	0,250													✓			✓	✓
A1.1.4	Redes auto-organizáveis	0,250													✓			✓	
A1.2.1	Algoritmo genético	0,143	✓	✓		✓	✓	✓	✓	✓									
A1.2.2	Estratégia evolutiva	0,143	✓			✓		✓			✓								
A1.2.3	Programação genética	0,143						✓											
A1.2.4	Programação evolutiva	0,143																	
A1.2.5	Evolução diferencial	0,143	✓																
A1.2.6	Evolução cultural	0,143																	
A1.2.7	Coevolução	0,143																	
A1.3.1	PSO	0,250	✓			✓		✓			✓					✓			
A1.3.2	PSO discreto	0,250						✓											
A1.3.3	Ant System	0,125																	
A1.3.4	Ant Colony System	0,125			✓						✓								
A1.3.5	Max-Min Ant System	0,125																	
A1.3.6	Ant System Rank	0,125																	
A1.4.1	CLONALG	0,334							✓										
A1.4.2	optIA	0,334																	
A1.4.3	aiNet	0,334																	
A1.5.1	Sistemas de Inferência Fuzzy	0,500										✓							
A1.5.2	Fuzzy Tipo II	0,167																	
A1.5.3	Fuzzy C-Means	0,167																	
A1.5.4	Algoritmos Neuro-Fuzzy	0,167										✓							
A1.6.1	Steepest descent/hill climbing	0,167				✓		✓											
A1.6.2	Simulated annealing	0,167				✓		✓	✓	✓	✓								
A1.6.3	Tabu search	0,167						✓	✓	✓	✓								
A1.6.4	Scatter search	0,167																	
A1.6.5	GRASP	0,167			✓														
A1.6.6	VNS	0,167						✓											
A1.7.1	Apriori	0,112													✓	✓			
A1.7.2	Tertius	0,112													✓				
A1.7.3	HotSpot	0,112													✓				
A1.7.4	Redes de Bayes	0,055												✓	✓	✓			✓
A1.7.5	C4.5	0,055													✓				✓
A1.7.6	KStar	0,055													✓				
A1.7.7	NaiveBayes	0,055													✓	✓			✓
A1.7.8	RandomForest	0,055													✓	✓			✓
A1.7.9	RandomTree	0,055													✓	✓			✓
A1.7.10	Cobweb	0,167													✓				✓
A1.7.11	K-Means	0,167													✓	✓			✓
A1.8.1	PGA	0,062																	
A1.8.2	MOGA	0,062						✓											
A1.8.3	NSGA	0,062						✓					✓						
A1.8.4	NSGA-II	0,062	✓			✓		✓					✓						
A1.8.5	NPGA	0,062																	
A1.8.6	SPEA	0,062											✓						
A1.8.7	SPEA-II	0,062											✓						
A1.8.8	PAES	0,062	✓																
A1.8.9	PESA	0,062																	
A1.8.10	PESA-II	0,062																	
A1.8.11	MOMGA	0,062																	
A1.8.12	ARMOGA	0,062																	
A1.8.13	*Multobjective AS*	0,062																	
A1.8.14	*Multiobjective PSO*	0,062	✓																
A1.8.15	POSA	0,062																	
A1.8.16	MOSA	0,062																	

Tabela 37 – Cobertura das funções pertencentes à área *Suporte ao processo de otimização* (1)

ID	Funcionalidade	Peso	*jMetalCpp*	*MAFRA*	*MAGMA*	*MALLBA*	*Nyx*	*ParadisEO*	*PMF*	*Rosemberg*	*Weyland*	*Zha*	*METCO*	*B-Course*	*WEKA*	*RapidMiner*	*Prediction API*	*TensorFlow*	*Azure ML Studio*
A2.1.1	Diferentes classes de problemas	0,125	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
A2.1.2	Pré-processamento dos dados de entrada	0,125												✓	✓	✓	✓	✓	✓
A2.1.3	Codificação da solução	0,125	✓	✓	✓	✓	✓	✓	✓	✓			✓						
A2.1.4	Definição da estrutura de vizinhança	0,125			✓			✓		✓			✓						
A2.1.5	Construção de heurísticas baseadas em população	0,125	✓	✓		✓	✓	✓	✓	✓			✓						
A2.1.6	Estratégias para seleção de soluções	0,125	✓	✓		✓	✓	✓	✓	✓			✓						
A2.1.7	Especificação de funções objetivo	0,125	✓	✓	✓		✓		✓	✓			✓						
A2.1.8	Definição e controle de restrições	0,125	✓	✓	✓		✓	✓	✓	✓			✓						
A2.2.1	Hibridização	0,334	✓		✓	✓		✓		✓		✓	✓		✓	✓		✓	✓
A2.2.2	Hiper-heurísticas	0,334											✓						
A2.2.3	Execução paralela e distribuída	0,334				✓		✓	✓	✓	✓		✓				✓	✓	✓
A2.3.1	Número máximo de iterações	0,167	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓		✓	✓		✓	
A2.3.2	Valor específico obtido pela função objetivo	0,167	✓	✓	✓	✓	✓	✓	✓	✓			✓						
A2.3.3	Tempo de execução	0,167				✓	✓	✓	✓										
A2.3.4	Combinação logica entre cond. de parada	0,500						✓											
A2.4.1	Repetição automática de tarefas simples	0,250													✓	✓			✓
A2.4.2	Automatização de tarefas variando os parâmetros de execução	0,250													✓	✓			✓
A2.4.3	Repetição automática de diferentes tarefas	0,250																	
A2.4.4	Automatização de diferentes tarefas em diferentes problemas	0,250																	
A2.5.1	Def. das Hipóteses	0,167																	
A2.5.2	Modelagem do experimento	0,167																	
A2.5.3	Desenho do experimento	0,167																	
A2.5.4	Auxílio na execução do experimento	0,167																	
A2.5.5	Geração do plano de execução	0,167																	
A2.5.6	Importar/Exportar experimentos	0,167																	
A2.6.1	T-Student	0,100																	
A2.6.2	ANOVA	0,100																	
A2.6.3	Wilcoxon	0,100																	
A2.6.4	Mann-Whitnet	0,100																	
A2.6.5	Kolmogorov-Smirnov	0,100																	
A2.6.6	Importar/Exportar análises estatísticas	0,500																	
A2.7.1	Design	0,167			✓							✓	✓	✓	✓	✓			✓
A2.7.2	Configuração das técnicas de IA	0,167										✓	✓	✓	✓	✓			✓
A2.7.3	Modelagem do problema	0,167										✓			✓	✓			✓
A2.7.4	Auxílio no planejamento de tarefas de otimização	0,167																	
A2.7.5	Suporte a definição de diferentes projetos	0,167													✓	✓			✓
A2.7.6	Representação gráfica dos resultados	0,167				✓								✓	✓	✓			✓
A2.8.1	Importar dados	0,250						✓						✓	✓	✓	✓	✓	✓
A2.8.2	Exportar dados	0,250						✓							✓	✓			
A2.8.3	*Web Service*	0,250															✓	✓	✓
A2.8.4	Manipulação de proj. com XML ou JSON	0,250															✓	✓	

APÊNDICE C – Materiais do Experimento

Questionário de Caracterização dos Participantes

Prezado participante,

Esse estudo experimental está sendo realizado com o objetivo de analisar o desenvolvimento de sistemas de inteligência computacional com e sem a utilização da ferramenta Athena, com o propósito de avaliar o impacto dessas duas técnicas, com respeito a eficiência (esforço), a eficácia (qualidade dos resultados) e a percepção de qualidade (satisfação dos participantes).

Todas as questões éticas relacionadas aos experimentos com seres humanos foram consideradas durante o planejamento desse estudo. Portanto, todos os participantes serão instruídos quanto ao objetivo do estudo, as implicações de sua participação, o valor científico desse estudo empírico e a confiabilidade dos dados. Por fim, cada participante deve assinar um termo de consentimento livre e esclarecido, onde declara-se a participação voluntária e o total conhecimento dos pontos supracitadas.

O questionário a seguir visa caracterizar os participantes para facilitar a compreensão dos resultados. Por favor, responda-o fielmente.

Dados Gerais	
Nome Completo	
E-mail	
IRA ou Cargo	

1. Qual seu nível de formação?

[ ] Graduando
[ ] Mestrando
[ ] Doutorando
[ ] Graduado
[ ] Mestre
[ ] Doutor

2. Em qual setor você atua?

[ ] Acadêmico (ensino) [ ] Estudante [ ] Industrial (empresarial)

3. Qual o nome da empresa/universidade em que você atua?

_____

4. Quanto tempo de experiência você possui na área em que atua?

_____

Conhecimentos em Programação

5. **Como você definiria seus conhecimentos com relação à programação orientada a objetos utilizando a linguagem Java?**

   [ ] 0 - Nunca programei utilizando a linguagem Java.

   [ ] 1 - Minha experiência é básica. Possuo conhecimento teórico aplicado apenas no contexto acadêmico.

   [ ] 2 - Minha experiência é moderada. Possuo conhecimento teórico somado de experiências práticas individuais.

   [ ] 3 – Minha experiência é avançada. Possuo conhecimento teórico somado de experiências práticas em projetos na indústria.

   Comentários (utilizar apenas se necessário):

   _____

   _____

Conhecimentos em Inteligência Computacional

6. **Informe seu nível de conhecimento nas seguintes técnicas de inteligência computacional (algoritmos genéticos, sistemas fuzzy, inteligência de enxames, redes neurais artificiais ou sistemas imunológicos artificiais) de acordo com a legenda abaixo:**

   [ ] 0 - Nunca utilizei.

   [ ] 1 - Minha experiência é básica. Possuo conhecimento teórico aplicado apenas no contexto acadêmico.

   [ ] 2 - Minha experiência é moderada. Possuo conhecimento teórico somado de experiências práticas individuais.

   [ ] 3 – Minha experiência é avançada. Possui conhecimento teórico somado de experiências práticas em projetos na indústria.

   Comentários (utilizar apenas se necessário):

   _____

   _____

7. **Preencha o formulário a seguir:**

Você já cursou a disciplina...	**Opções**	**Período (Ex.: 2014.1)**
Programação OO	[ ] Sim [ ] Não	
Inteligência Artificial	[ ] Sim [ ] Não	
Engenharia de Software	[ ] Sim [ ] Não	
Aprendizagem de Máquina	[ ] Sim [ ] Não	
Redes Neurais Artificiais	[ ] Sim [ ] Não	

8. **Como você definiria seu conhecimento nas seguintes técnicas de inteligência computacional:**

Técnica	**Nível de Conhecimento**
Sistemas Fuzzy	
Algoritmos Genéticos (por exemplo, NSGA-II)	
Redes Neurais Artificiais	
Inteligência de Enxames (como, Ant System)	

Legenda:
0 – Nunca utilizei;
1 – Minha experiência é básica. Possuo conhecimento teórico aplicado apenas no contexto acadêmico;
2 – Minha experiência é moderada. Possuo conhecimento teórico somado de experiências práticas individuais;
3 – Minha experiência é avançada. Possuo conhecimento teórico somado de experiências práticas em projetos na indústria;

Comentários (utilizar apenas se necessário):
_____
_____
_____
_____

9. **Como você definiria seu conhecimento nos seguintes problemas:**

Problemas	**Nível de Conhecimento**
Alocação de equipes	
Priorização de casos de testes	
Análise de sentimentos	
Estimativa do tamanho de nano-partículas de prata	

Legenda:
0 – Desconheço o problema;
1 – Possuo conhecimento apenas teórico;
2 – Possuo conhecimento teórico aplicado no contexto acadêmico;
3 – Possuo conhecimento teórico somado de experiências práticas em projetos na indústria;

Comentários (utilizar apenas se necessário):
_____
_____
_____
_____
_____
_____

Conhecimentos em Ferramentas de Inteligência Computacional

10. Qual seu nível de conhecimento em relação a ferramentas/frameworks de inteligência computacional (IC)?

[ ] 0 – Desconheço ferramentas de IC e suas aplicações.

[ ] 1 – Minha experiência é básica. Possuo conhecimento teórico aplicado apenas no contexto acadêmico.

[ ] 2 – Minha experiência é moderada. Possuo conhecimento teórico somado de experiências práticas individuais.

[ ] 3 – Minha experiência é avançada. Possuo conhecimento teórico somado de experiências práticas em projetos na indústria.

Comentários (utilizar apenas se necessário):

_____

_____

11. De acordo com a legenda a seguir, informe seu nível de conhecimento nas seguintes ferramentas:

Ferramenta	**Nível de Conhecimento**
JMetal – Metaheuristic Algorithms in Java	
JFuzzyLogic – Fuzzy systems in Java	
Neuroph – Java Neural Network framework	
Weka – Data mining algorithms	
MatLab – The language of technical computing	
Vision Blocks MIT	
App Inventor MIT	
Circuit Lab – Circuit simulator	
LabView	

Legenda:
0 – Nunca utilizei;
1 – Minha experiência é básica. Possuo conhecimento teórico sobre a ferramenta;
2 – Minha experiência é moderada. Possuo conhecimento teórico somado de experiências práticas individuais;
3 – Minha experiência é avançada. Possuo conhecimento teórico somado de experiências práticas em projetos na indústria;

Comentários (utilizar apenas se necessário):

_____

_____

_____

_____

Questionário pós-experimento

Prezado participante,

Para finalizar nosso estudo experimental precisamos coletar seu feedback sobre a condução do experimento e sobre a ferramenta avaliada. Todas as sugestões, críticas e comentários serão bem-vindas, portanto fique à vontade.

1. **Que nota você atribuiria a execução do experimento?**
   [ ] 0 – Mal executado. Os objetivos não estavam claros e o experimento foi mal conduzido;
   [ ] 1 – Razoável. Os objetivos estavam claros, entretanto ocorreram muitos problemas durante a execução do estudo;
   [ ] 2 – Bem executado. Os objetivos estavam claros, fui bem orientado e não ocorreram problemas durante a execução do estudo;

Comentários:

_____
_____
_____
_____
_____

2. **Que nota você atribuiria a ferramenta Athena?**
   [ ] 0 – Inadequada. A ferramenta não está adequada com a proposta de facilitar a construção de sistemas de inteligência computacional;
   [ ] 1 – Razoável. A ferramenta apresenta ganhos quanto ao custo durante a construção de sistemas de inteligência computacional, entretanto ainda possui muitas falhas e problemas de usabilidade;
   [ ] 2 – Boa. A ferramenta está adequada com a proposta de facilitar a construção de sistemas de inteligência computacional, possui poucas falhas e boa usabilidade, entretanto não parece adequada para utilização em ambiente industrial;
   [ ] 3 – Excelente. A ferramenta está adequada com a proposta de facilitar a construção de sistemas de inteligência computacional, possui poucas falhas, boa usabilidade e pode ser utilizada em ambiente industrial;

Comentários:

_____
_____
_____
_____
_____
_____

3. **Na sua opinião, quais os pontos fortes e fracos da Athena?**

_____
_____
_____
_____
_____
_____

4. **Se você fosse responsável por esse projeto, quais ações você definiria como trabalhos futuros?**

_____
_____
_____
_____
_____
_____

5. **Diante da necessidade de construção de um sistema de inteligência computacional, você utilizaria a ferramenta Athena?**
   [ ] Não utilizaria;
   [ ] Utilizaria apenas em projetos acadêmicos;
   [ ] Utilizaria em projetos acadêmicos e industriais;

Comentários:

_____
_____
_____
_____
_____
_____

6. **Questão aberta. Se ainda há algo que você deseja comentar, fique à vontade para conceder o feedback no espaço a seguir:**

_____
_____
_____
_____
_____
_____
_____

Em nome de toda a equipe de pesquisadores envolvidos nesse projeto, gostaríamos de lhe agradecer pela contribuição concedida a esse estudo experimental. Acreditamos no poder que a ciência tem para transformar e facilitar a vida de todos.

Muito obrigado.

Termo de Consentimento Livre e Esclarecido

Eu, _____, tendo sido convidado(a) a participar como voluntário(a) do estudo experimental do trabalho intitulado "*Athena: uma arquitetura e uma ferramenta para auxiliar o desenvolvimento de sistemas baseados em Inteligência Computacional*", recebi do mestrando Pedro Almir Martins de Oliveira, responsável por sua execução, as seguintes informações que me fizeram entender sem dificuldades e sem dúvidas os seguintes aspectos:

1. Que o estudo se destina a avaliar o desenvolvimento de sistemas de inteligência computacional com e sem a utilização da ferramenta Athena;
2. Que a importância deste estudo é a de possibilitar a disponibilização de uma ferramenta que facilite o processo de desenvolvimento de sistemas de inteligência computacional;
3. Que os resultados que desejam alcançar são os seguintes: i) o esforço necessário para desenvolver sistemas de inteligência computacional com e sem a utilização da ferramenta Athena; ii) a qualidade dos resultados obtidos por sistemas desenvolvidos com e sem a utilização da ferramenta Athena; iii) a satisfação dos usuários ao desenvolverem sistemas com e sem a utilização da ferramenta;
4. Que o estudo começará às ____:____ do dia _____ e está previsto para terminar às ____:____ do dia _____;
5. Que a participação no estudo não trará nenhum risco à minha saúde física ou mental;
6. Que, a qualquer momento, eu poderei recusar a continuar participando do estudo;
7. Que as informações conseguidas através da minha participação não permitirão a identificação da minha pessoa, exceto aos responsáveis pelo estudo, e que a divulgação das mencionadas informações só será feita entre os profissionais e estudiosos do assunto;

Finalmente, tendo eu compreendido perfeitamente tudo o que me foi informado sobre a minha participação no mencionado estudo e estando consciente dos meus direitos, das minhas responsabilidades, dos riscos e dos benefícios que minha participação implicam, concordo em dele participar e para isso dou meu consentimento sem que para isso eu tenha sido forçado ou obrigado.

_____

Assinatura do Participante

Problema da Alocação de Equipe no Desenvolvimento de Software

A alocação equipes de desenvolvimento é essencial para o processo ágil de desenvolvimento de software, entretanto esse problema é NP-difícil, uma vez que compreende a atribuição de equipes multifuncionais e auto organizadas. Muitos pesquisadores têm direcionado esforços para aplicar técnicas de Inteligência Computacional para resolver este problema. Em BRITTO *et al*. é proposta uma abordagem híbrida baseada na metaheurística multiobjectivo NSGA-II e em sistemas *fuzzy* para resolver esse problema. A abordagem proposta é dividida em três etapas:

1. **Estimação da Produtividade:** nesta etapa um sistema *fuzzy* será utilizado para inferir a produtividade dos desenvolvedores com base nas notas de conhecimento, habilidade e atitude de cada desenvolvedor;
2. **Geração das equipes:** nesta etapa serão geradas as equipes buscando maximizar a produtividade total da equipe e minimizar o custo. A produtividade da equipe é obtida somando a produtividade de todos os desenvolvedores selecionados e o custo da equipe é obtido somando o salário dos desenvolvedores pertencentes aquela equipe. Além disso, deve-se obedecer a restrição do tamanho máximo das equipes;
3. **Estimação da qualidade das equipes geradas:** nesta etapa um sistema *fuzzy* será utilizado para estimar a qualidade das equipes geradas na etapa anterior. Esse sistema *fuzzy* possui como entrada a produtividade total da equipe e seu custo;

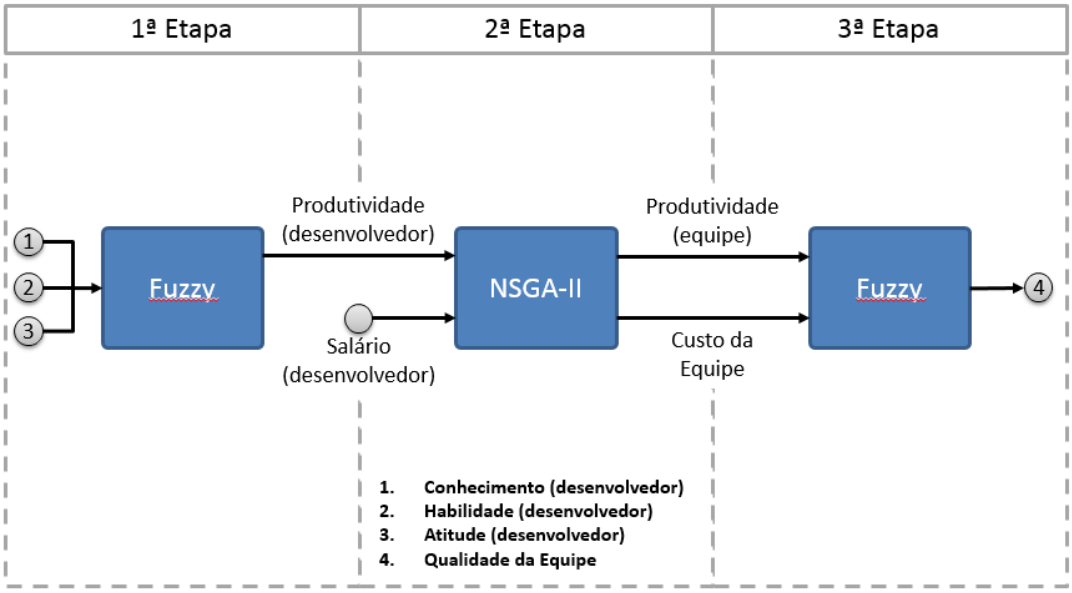


Figura 1. Abordagem de alocação de equipes.

Com base na descrição apresentada acima e de posse das ferramentas disponibilizadas pelos pesquisadores, desenvolva um sistema inteligente capaz de resolver o problema da alocação de equipes usando uma abordagem híbrida (Fuzzy + NSGA-II).

Problema da estimativa do tamanho das Nanopartículas de Prata

A preparação de AgNPs (nanopartículas de prata) pode ser considerado como um processo multivariável, pois uma série de fatores como, por exemplo, a temperatura, o pH do meio reacional, o tempo de síntese, a concentração dos sais de prata, a concentração do agente redutor, bem como, a concentração do agente estabilizante, entre outros, desempenham papel determinante e tendem a afetar as características do produto final.

Na abordagem utilizada neste estudo experimental, uma RNA foi empregada para a predição do tamanho de AgNPs (nanopartículas de prata) reduzidas e estabilizadas com a utilização de um polissacarídeo extraído da alga vermelha do gênero Gracilaria. Foi utilizada uma RNA do tipo MLP com algoritmo de treinamento *Backpropagation* (BP).

Para o problema em questão a melhor topologia encontrada para a RNA foi: 12 neurônios na primeira camada escondida e 5 neurônios na segunda camada escondida, taxa de aprendizado igual a 0.85, máximo de épocas 1500, precisão requerida de 1x10E-06.

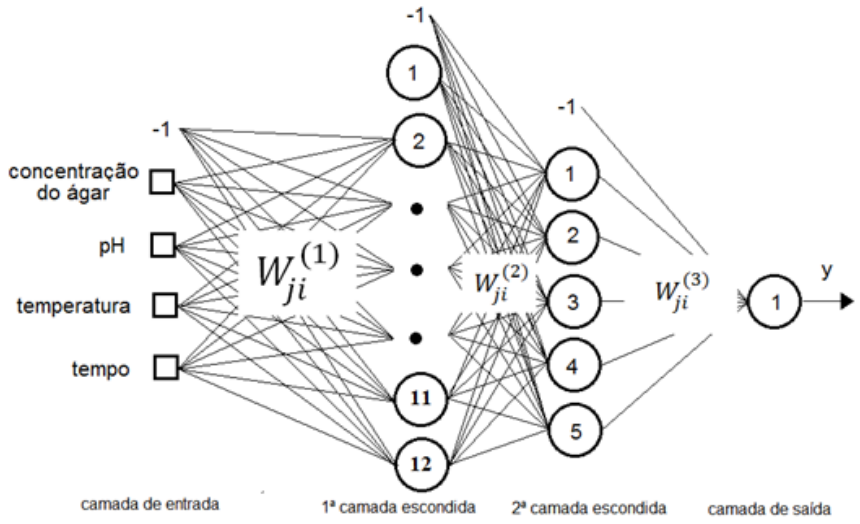


Figura 1. Estrutura da RNA utilizada para o problema.

Com base na descrição apresentada acima e de posse das ferramentas disponibilizadas pelos pesquisadores, desenvolva um sistema inteligente capaz de estimar o tamanho das nanopartículas de prata utilizando uma rede neural do tipo MLP com quatro entradas: concentração de ágar, pH, temperatura e tempo.

APÊNDICE D – Inclusão de Novos Módulos

Um dos princípios arquiteturais que norteia o desenvolvimento da Athena é a extensibilidade. Esse princípio embasou a construção de um mecanismo que permite a inclusão de módulos em tempo de execução (*on-the-fly*), sem a necessidade de reiniciar a aplicação. Isso foi possível com a adoção das especificações da arquitetura OSGi (ALLIANCE, 2003). Entretanto, percebeu-se que a inclusão de módulos *on-the-fly* pode gerar problemas de segurança, uma vez que não há nenhuma verificação desses códigos. Por isso, a versão atual da ferramenta prescreve um processo para a incorporação de novos módulos.

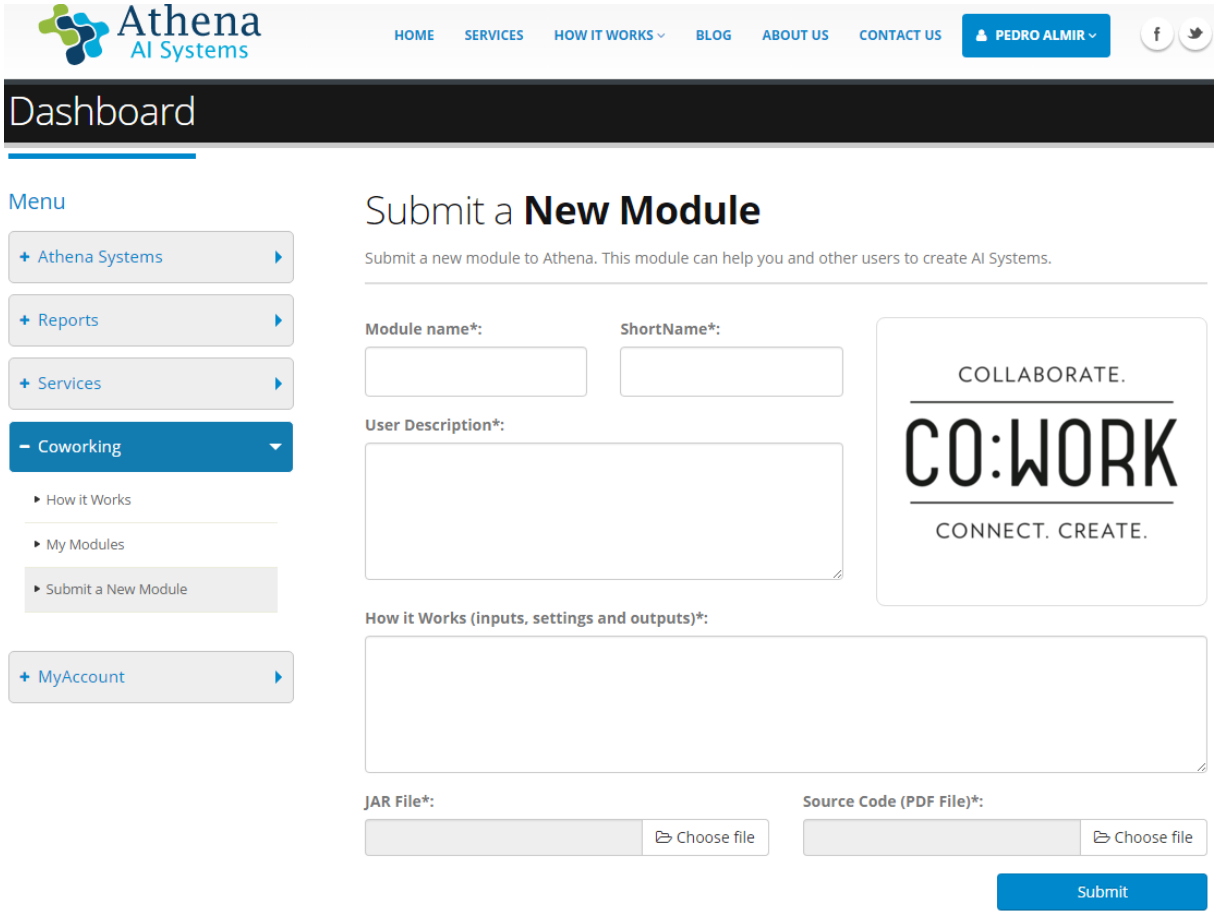


Figura 46 – Tela para submissão de novos módulos.

A Figura 46 apresenta a tela para submissão de novos módulos. O usuário deve informar algumas informações básicas sobre o módulo (nome, descrição e funcionamento), além do código-fonte (PDF) e do pacote executável (JAR). Cada módulo é analisado por um membro da equipe de desenvolvimento da Athena que fica responsável por garantir o comportamento correto da técnica e a ausência de código malicioso.

APÊNDICE E – Avaliação de Performance

Os resultados obtidos no estudo experimental mostraram que o tempo de execução é maior quando as soluções são desenvolvidas na Athena. Esse comportamento motivou a realização de um estudo, com foco no desempenho da ferramenta. Neste apêndice descreve-se uma avaliação de performance preliminar.

E.1 Contexto

A maioria das instituições de pesquisa precisa executar grandes simulações durante suas pesquisas e essa necessidade é comumente resolvida por meio da compra de computadores com grande poder de processamento. No entanto, os custos de aquisição e manutenção desses computadores são elevados; essas máquinas passam a maior parte do tempo ociosas e têm um curto período de validade, tendo em vista que logo ficam obsoletas. Em outras palavras, os custos de um computador dessa natureza não justificam o investimento, pelo menos não para uma parcela dos casos em que eles são utilizados (COFFEY, 2011).

Outra alternativa aos computadores com grande poder de processamento é o uso de infraestrutura remota ou infraestrutura como serviço (IaaS, do inglês *Infrastructure as a Service*), como a Amazon Elastic Compute Cloud (EC2), IBM Cloud, HP Cloud, dentre outras. Esse modelo proporciona grandes vantagens para seus adeptos, porque não requer investimento na compra de máquinas físicas, há a garantia da adequação da tecnologia ao crescimento da entidade que a utiliza, a infraestrutura exigida é garantida e de alta qualidade, existe um maior controle do ambiente computacional e o investimento em capacitação de profissionais é baixo. Além disso, nessa abordagem os usuários pagam apenas pelo recurso computacional consumido (ARMBRUST et al., 2010).

Outra vantagem do modelo IaaS é a capacidade de aumentar ou diminuir os recursos computacionais de acordo com a necessidade. Ademais, existem todas as vantagens associadas à Computação em Nuvem, por exemplo, a mobilidade (“a qualquer hora, em qualquer lugar”), segurança e alta disponibilidade (MARSTON et al., 2011).

Como a Athena foi projetada com base no conceito de Inteligência Computacional como Serviço (CIaaS) e seu *Back-end* foi desenvolvido utilizando a infraestrutura da Amazon EC2, ela pode transmitir a seus usuários algumas vantagens apresentadas acima, por exemplo, pagar apenas quando for consumido, pagar somente pela quantidade de CPU e de memória utilizadas e aumentar ou diminuir os recursos computacionais de acordo com a necessidade.

Neste contexto, decidiu-se avaliar a performance da Athena, considerando a seguinte questão de pesquisa:

- **QP4**: Qual é o impacto do tamanho do problema no tempo de execução dos algoritmos de Inteligência Computacional implementados na Athena?

E.2 Metodologia

Para responder a questão de pesquisa apresentada na Subseção E.1, decidiu-se utilizar um dos objetos detalhados na Seção 4.4 - seleção de caso de teste - mas aumentando exponencialmente o tamanho do problema e usando as mesmas configurações em todas as execuções. A Tabela 38 apresenta os tamanhos do problema utilizados nessa avaliação. Nesse caso, o tamanho do problema é determinado pela quantidade de casos de teste.

Tabela 38 – Tamanhos do problema utilizado na avaliação de performance.

Problema	T1	T2	T3	T4	T5	T6	T7
Seleção de Casos de Teste	1024	2048	4096	8192	16384	32768	65536

As medições foram efetuadas na Athena usando diferentes tipos de instâncias disponíveis na Amazon EC2 e na infraestrutura local.

E.3 Ambiente de Avaliação

Athena. Como mencionado anteriormente, o *Back-end* da ferramenta é executado sobre a infraestrutura da Amazon EC2. Essa infraestrutura fornece vários tipos de máquinas, cada uma com diferentes configurações. As máquinas usadas nessa avaliação estão descritas na Tabela 39. Todas as elas foram configuradas com o sistema operacional Linux.

Tabela 39 – Especificação das máquinas da Amazon EC2.

Type	**Grade**	**Architecture**	**CPU***	**Cores**	**Memory****
Micro (t2)	Micro	64 bits	< 3 ECU	1	1 GiB
Standard (m3)	Medium	64 bits	3.0 ECU	1	3.75 GiB
	Large	64 bits	6.5 ECU	2	7.50 GiB
	xLarge	64 bits	13.0 ECU	4	15.0 GiB
	2xLarge	64 bits	26.0 ECU	8	30.0 GiB
High CPU (c4)	4xLarge	64 bits	62.0 ECU	16	30.0 GiB
	8xLarge	64 bits	132.0 ECU	36	60.0 GiB

Nota: * O poder computacional é expresso em termos de EC2 Compute Units (ECUs), que são equivalentes a 1.0-1.2 GHz de um processador 2007 Xeon ou Opteron. ** 1 Gibibyte (GiB) = 2^{30} bytes, de acordo com a International Electrotechnical Commission (IEC).

Infraestrutura local. Foram utilizadas duas máquinas fornecidas pela UFPI. As características desses computadores são apresentadas na Tabela 40.

Tabela 40 – Especificação dos computadores da UFPI utilizados no estudo de performance.

Máquina	**Sistema Operacional**	**Arq.**	**CPU**	**Cores**	**Memória**
HP Compaq 4300 Pro	Windows 7 Professional edition	64 bits	3.30 GHz Intel Core i3 - 3220	2	8 GB
Dell XPS L502X	Windows 7 Professional edition	64 bits	2.20 GHz Intel Core i7 - 2670QM	4	8 GB

Nota: Arq. foi usado como abreviação para arquitetura da máquina.

E.4 Resultados

A Tabela 41 apresenta os resultados obtidos no estudo de performance. Cada avaliação foi executada trinta vezes eliminando o maior e o menor valor. Finalmente, obteve-se o tempo médio de execução nos diferentes contextos (local e Athena) de acordo com o tamanho do problema.

Tabela 41 – Resultados do estudo de performance - Tempo de Execução (em segundos).

	Seleção de Casos de Teste						
Contexto/Tamanho	**1024**	**2048**	**4096**	**8192**	**16384**	**32768**	**65536**
HP Compaq	0,750	1,880	4,090	8,490	17,230	34,680	69,650
Dell XPS	0,740	1,860	4,010	8,230	16,720	33,590	67,020
Athena (t2.micro)	3,217	5,188	6,053	11,622	28,554	67,581	∞
Athena (m3.medium)	2,445	4,522	5,530	10,514	19,394	38,335	77,336
Athena (m3.large)	1,368	1,955	3,361	5,677	10,403	19,525	70,752
Athena (m3.xlarge)	1,201	1,853	3,206	5,476	10,372	19,217	63,518
Athena (m3.2xlarge)	1,110	1,764	3,172	5,347	10,207	18,613	40,713
Athena (c4.4xlarge)	1,089	1,601	2,992	5,222	10,144	18,548	38,668
Athena (c4.8xlarge)	1,022	1,625	3,042	5,309	10,202	18,537	38,298

Nota: ∞ - significa que a execução não foi completada por problemas de memória.

A Figura 47 mostra o tempo de execução (em segundos) necessário para resolver os diferentes tamanhos do problema em cada contexto utilizado nesse estudo. Para simplificar a visualização dos resultados, utilizou-se o tempo médio de execução das máquinas HP Compaq e Dell XPS para representar o tempo no contexto da infraestrutura local e omitiu-se os resultados dos contextos m3.2xlarge e c4.4xlarge porque foram muito semelhantes.

Pode-se perceber (Figura 47) que em todos os contextos o tempo de execução cresce exponencialmente quando aumenta-se o tamanho do problema. No entanto, à medida que o recurso computacional disponível para a Athena aumenta, o crescimento do tempo de execução torna-se menor. Além disso, é possível realizar a seguinte interpretação: por ter a capacidade de ajustar seu poder computacional, a Athena possui uma melhor adequação aos diferentes tamanhos do problema.

A Figura 48 mostra uma comparação do tempo de execução na infraestrutura local e com os contextos nos quais a Athena obteve o pior e o melhor resultado. Pode-se observar que o tempo de execução da Athena começa um pouco mais alto do que a execução local. Isso deve-se à necessidade de processar as requisições provenientes do cliente. No entanto,

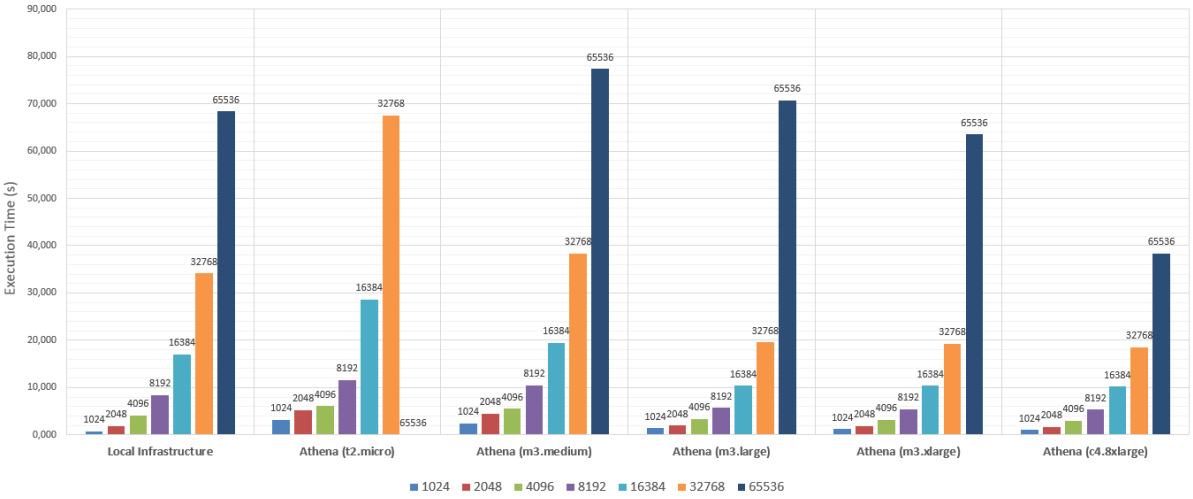


Figura 47 – Tempo de execução em relação ao contexto de avaliação.

à medida que o tamanho do problema cresce, essa situação é revertida e a infraestrutura local obteve piores resultados.

Finalmente, a resposta à quarta questão de pesquisa (QP4) está diretamente ligada à Lei de Amdahl (AMDAHL, 1967), que diz que o *speedup* de um programa usando múltiplos processadores em computação paralela é limitado pelo tempo necessário para a fração sequencial de um programa. A fração sequencial associada à Athena está na transferência de dados entre os componentes da arquitetura. A partir desse ponto, a máquina entra em execução e, quanto mais robusta a máquina, menor o tempo total.

A infraestrutura local obteve bons resultados quando a transferência de dados representava uma grande parcela do tempo de execução dos algoritmos, mas quando o problema cresce, esse tempo de transferência torna-se irrelevante diante do tempo total de processamento. Isso faz com que a diferença nas execuções das tarefas sequenciais

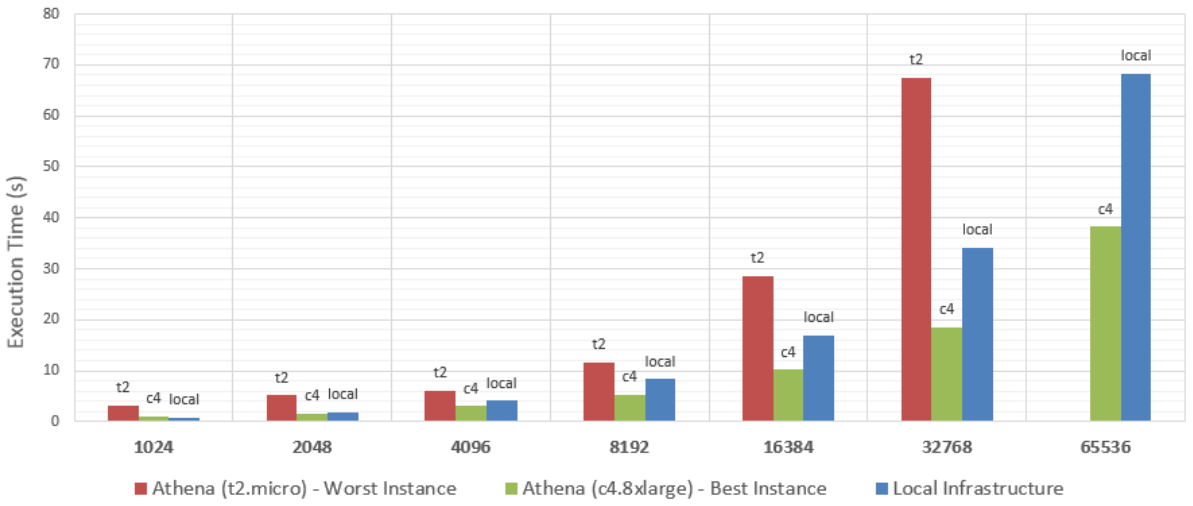


Figura 48 – Tempo de execução local comparado com o melhor e pior resultados da Athena.

seja desconsiderada para o resultado geral. A Athena não obteve bons resultados no contexto inicial (*t2.micro*), uma vez que a configuração dessa máquina não permitiu uma recuperação do tempo de transferência e melhora do tempo total, entretanto a partir da execução com máquinas mais robustas, os resultados foram melhorando até superar as máquinas locais.

Esse resultado era esperado, mas essa avaliação foi importante para esclarecer que o custo-benefício da execução de algoritmos de IC na Athena é melhor do que as execuções em infraestrutura local. Isso deve-se ao fato de que à medida que aumenta o tamanho do problema é necessário obter uma máquina mais robusta para encontrar uma solução em tempo hábil. Segundo os estudos apresentados por Kondo *et al.* (KONDO et al., 2009) e por Coffey (COFFEY, 2011), o custo de aquisição e manutenção de máquinas locais é muito maior quando comparado ao uso de computadores remotos (IaaS).

E.5 Ameaças à Validade

Muitos fatores podem ter influência sobre a validade interna e externa desta avaliação de desempenho. Ameaças à validade interna dizem respeito a questões que podem indicar uma relação causal entre o tratamento e os resultados, mesmo quando essa relação não existe (WOHLIN et al., 2012). Nesta avaliação, a implementação dos algoritmos é uma ameaça à validade interna, pois um erro pode levar a um tempo de execução excessivo e com isso direcionar para conclusões erradas. No entanto, as implementações utilizadas foram avaliadas pelos autores desta pesquisa e não mostraram erros estruturais que comprometessem seu tempo de execução.

Ameaças à validade externa dizem respeito a questões que limitam a capacidade de generalizar os resultados (WOHLIN et al., 2012). O estudo aqui descrito, embora também seja preliminar, demonstrou aquilo que era esperado: a escalabilidade da Athena provê melhores resultados sempre que o tempo de execução for considerável em relação ao tempo de transferência de dados para a execução remota dos algoritmos.

E.6 Considerações Finais

Este apêndice apresenta uma avaliação de performance para investigar o impacto do tamanho do problema no desempenho da Athena. Esse estudo não foi realizado com o rigor exigido para tal avaliação, portanto não foi possível obter conclusões concretas sobre a performance da proposta. Faltou a definição protocolo rígido para avaliação, além da utilização de testes estatísticos sobre os resultados. Entretanto, essa avaliação preliminar serviu como preparação para uma investigação mais formal.