Desenvolvimento de Sistemas Robóticos Inteligentes com Paradigma Orientado à Notificações

Development of Intelligent Robotics Systems with Notification-Oriented Paradigm

Matheus Diniz de Freitas (orientado) *, João Alberto fabro(Orientador)†, Rodrigo Yuske Yamauchi‡

RESUMO

CONTEXTO: Em meio ao desenvolvimento do emergente Paradigma Orientado à Notificações (PON) na Universidade Tecnológica Federal do Paraná (UTFPR) e a Competição Latino-Americana de Robótica na categoria *Very Small Size Soccer* (VSSS). OBJETIVO: Através do estudo de metodologias para o desenvolvimento de robôs autônomos, projetar um software utilizando o Paradigma Orientado a Notificações para uma equipe de robôs VSSS. PROCEDIMENTOS: Como este ano a competição será realizada apenas com simuladores, o sistema de controle PON foi integrado à interface do simulador, mas está preparado para integração perfeita com a equipe de robôs reais que também estão em desenvolvimento na universidade. Além disso, é a primeira vez que o PON está sendo usado em uma competição de robótica. MÉTODOS DE OBSERVAÇÃO: O projeto foi desenvolvido por membros da Universidade Tecnológica Federal do Paraná, Campus Curitiba, de forma remota. O software desenvolvido foi frequentemente comparado a outros de competições anteriores. RESULTADOS: O algoritmo de gerenciamento de robôs mostrou-se funcional e ágil, assim como a estratégia aplicada para coordenação entre atacante e defensor. Em comparação com o programa do ano passado, também há ganho de desempenho.

Palavras-chave: Robótica. PON. VSSS.

ABSTRACT

CONTEXT: Amid the development of the emerging Notification-Oriented Paradigm (NOP) at Universidade Tecnológica Federal do Paraná (UTFPR) and the annual Latin American Robotics Competition in the Very Small Size Soccer (VSSS) category. OBJECTIVE: Through the study on methodologies for the development of autonomous robots, to design a software using the Notification-Oriented Paradigm for a team of VSSS robots. PROCEDURES: Since this year the competition will be held only with simulators, the NOP control system was integrated to the simulator interface, but it is prepared to seamless integration with the team of real robots that are also under development at the university. Furthermore, it is the first time the NOP is being used in a robotic competition. OBSERVATION METHODS: The project was developed by members of Universidade Tecnológica Federal do Paraná, Campus Curitiba, in a remote way. The software developed was often compared to ones made for past competitions. RESULTS: The algorithm for managing robots proved to be functional and agile, as well as the applied strategy for coordination between attacker and defender. Compared to last years' program, there is also performance gain.

Keywords: Robotics. NOP. VSSS.

* Acadêmico de Engenharia de Computação/UTFPR-Campus Curitiba; dfmatheusdf@gmail.com.
† Departamento Acadêmico de Informática(DAINF-CT)/UTFPR-Campus Curitiba; fabro@utfpr.edu.br; https://orcid.org/0000-0001-8975-0323.
‡ Acadêmico de Engenharia de Computação/UTFPR-Campus Curitiba; rodrigoyuske@alunos.utfpr.edu.br.

1 INTRODUÇÃO

A Competição Latino-Americana de Robótica, ou LARC(*Latin American Robotics Competition*), é uma competição anual que apresenta uma variedade de competições relacionadas à robótica [1]. Uma das categorias é a *Very Small Size Soccer* (VSSS), na qual equipes de 3 robôs miniaturizados (tendo que ser menor que um cubo de dimensões máximas de 8 centímetros de altura, largura e comprimento) competem em um pequeno campo de futebol. Por conta da pandemia COVID-19, a exemplo da edição do ano passado, a competição será realizada remotamente, por meio de um simulador, que simula de maneira fisicamente plausível o comportamento dos robôs e da bola. As partidas de futebol são simuladas em um ambiente gráfico, possibilitando a visualização do jogo. Além disso, o simulador é responsável por armazenar as variáveis do estado atual do jogo, que são repassadas às equipes por meio de um protocolo de comunicação em rede denominado Protobuf [2]. Conforme a informação é recebida em um computador que executa o sistema de controle de cada equipe, ela é processada e comandos são enviados de volta ao simulador para execução dos comandos pelos robôs simulados. Há dois computadores, um para cada equipe que competem em cada partida.

Este projeto visa aplicar as técnicas e metodologias relacionadas ao Paradigma Orientado à Notificações (PON), em desenvolvimento na Universidade Tecnológica Federal do Paraná, à competição com futebol de robôs. A aplicação deste novo paradigma em programação de robôs é inédita e de significativa importância para pesquisas com o PON.

A equipe UTBots da UTFPR participa desta categoria desde 2017. Na edição deste ano de 2021, o sistema de controle foi totalmente re-desenvolvido, fim de que os integrantes da equipe aprendam mais sobre o PON, e também para avaliar a aplicação do paradigma a programação de robôs autônomos. O trabalho em questão visou responder: Qual a viabilidade da aplicação do PON em robôs autônomos? Quais os pontos positivos e negativos em relação aos códigos desenvolvidos pela equipe para a participação na competição em anos anteriores?

Tem-se portanto como objetivo do estudo testar e avaliar a implementação da programação dos robôs autônomos para a VSSS utilizando o emergente paradigma orientado à notificações.

2 MÉTODO

2.1 Sistema de Controle Inteligente

A estratégia a ser seguida pelo robô de execução é executada dentro de uma janela de encaixe, e contém desde instruções básicas, como iniciar tarefas para cada robô, até outras mais complexas, como cálculo de detecção de colisão e outras.

Todas as informações necessárias são passadas para o docker pelo simulador, desde a posição da bola e dos jogadores até o tempo de jogo e a pontuação atual.

O presente algoritmo implementado pela equipe está usando um paradigma emergente, que é o Paradigma Orientado à Notificação (PON), que será descrito em detalhes na próxima subseção. Este novo paradigma permite que você desenvolva um código mais otimizado e menos acoplado, facilitando assim a adaptação a mudanças futuras.

Para avaliar o nível de desempenho dos robôs em campo, são realizadas várias simulações de ataque versus

[1] Competição Latino-Americana de Robótica - www.cbrobotica.org
[2] Protocolo de Rede descrito em: https://developers.google.com/protocol-buffers/docs/proto3

defesa, onde é possível verificar os pontos fortes e fracos do ataque e da defesa. Essas simulações podem ser partidas inteiras ou jogadas específicas para ver como os robôs se comportam.

2.2 Desacoplamento e divisão de código

Visando um desacoplamento de código ainda maior do que no ano passado, e também o futuro retorno da competição presencial, todo o código produzido este ano se comunicará com o simulador e o juiz automático através de uma classe intermediária, que ficará responsável pelo recebimento e envio de essenciais informações do simulador. Permitindo assim o reaproveitamento do código para os próximos anos, exigindo apenas modificação na classe intermediária.

Para implementar esta classe, será usada memória compartilhada, que é conhecida por ser o mecanismo de comunicação intermediário mais rápido. Ele também foi escolhido, pois já havia sido usado nos códigos anteriores da equipe.

2.3 PON e tomadas de decisão

O PON foi proposto como um novo paradigma primeiro para software mas posteriormente também para hardware, de forma a fornecer um novo modelo lógico que pretende contribuir para uma maior produtividade e qualidade no desenvolvimento e execução de processos automatizados. Este paradigma visa promover um desempenho mais rápido de execução do sistema e maior facilidade de construção de sistemas complexos, especialmente sistemas paralelos e distribuídos.

Embora o PON herde alguns conceitos dos paradigmas IP (Imperativo) e DP (Declarativo), ele também possui uma nova forma de estruturar e executar a lógica de programas computacionais, o que justificaria sua classificação de paradigma (LINHARES; SIMAO; STADZISZ, 2015). Ao contrário do comportamento dos programas IP, onde a lógica é totalmente dependente da sequência de execução, e dos Sistemas Baseados em Regras, nos quais a sequência de execução é abstraída e dependente do mecanismo de inferência monolítica, o PON permite a expressão da dinâmica do software e sua lógica de causa-e-efeito por meio de notificações, que podem ser executadas em paralelo (LINHARES; SIMAO; STADZISZ, 2015).

Muitos dos paradigmas de programação atuais e mais usados são aplicados para criar programas com forte acoplamento e avaliações recorrentes muitas vezes desnecessárias (SIMÃO, 2005; LINHARES; SIMAO; STADZISZ, 2015). Nesse contexto, o PON busca solucionar esses problemas.

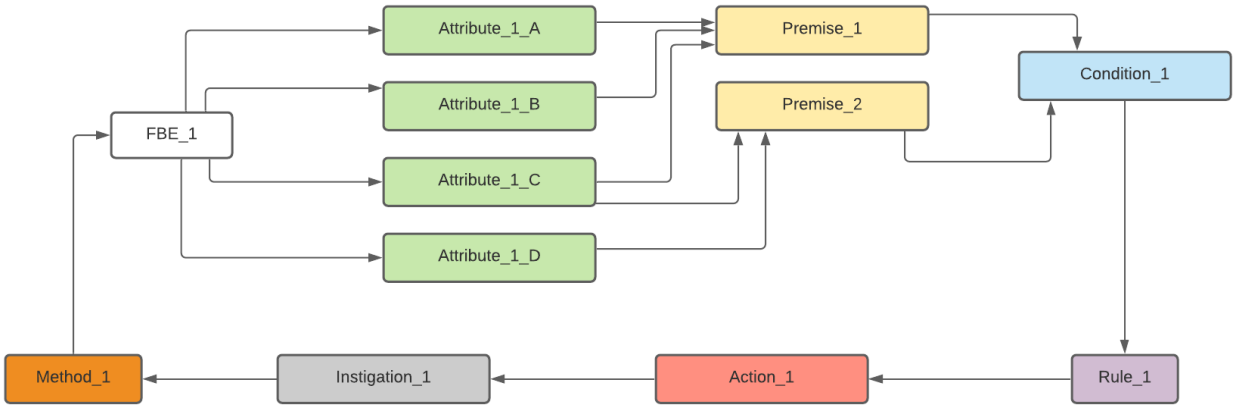
O processo de tomada de decisão com o PON é bastante diferente dos paradigmas comuns. Para explicá-lo resumidamente, apesar do que acontece no IP, incluindo a programação orientada a objetos, em que o loop de iteração é explicitamente criado por meio de comandos como while e for, todas as decisões são executadas por regras, que só são notificadas quando as variáveis observadas têm seu valor alterado. Esta característica possibilita o desenvolvimento de programas computacionais com alto nível de desacoplamento e facilidade de modificações.

Um programa escrito em PON tem quatro seções, a saber, definições FBE, instanciações FBE, estratégia de resolução de conflitos e declarações de regras. FBE significa *Fact Base Element*, a entidade computacional dedicada do PON. Por meio de seus Atributos e Métodos, os FBEs são capazes de correlação lógico-causal através de entidades Regras, os elementos fundamentais do paradigma.

Um FBE contém um conjunto de Atributos, que são avaliados pelas Premissas das Regras (conectados a,

como parte de uma ou mais Condições das Regras). Uma vez que seu valor muda, o Atributo notifica todas as Premissas relevantes para que elas reavaliem seus estados lógicos. Se então a Premissa tem seu valor alterado, ela notifica todas as Condições conectadas. A seguir, cada Condição reportada verifica se todas as Premissas que a integram estão satisfeitas. Em caso afirmativo, sua regra relacionada é aprovada. Quando uma regra é aprovada, ela executa uma Ação, que, por sua vez, executa todas as Instigações às quais está conectada. Cada Instigação dispara a execução de algum serviço de uma instanciação FBE, através de seus Métodos, que contém alguma execução (por exemplo, enviar comando para o robô). Todo este processo é apresentado graficamente com um modelo simples de um programa escrito no PON na Figura 1 e extensivamente explicado em (BANASZEWSKI et al., 2007). Em princípio, a essência da computação no PON é organizada e distribuída entre entidades reativas, que colaboram entre si por meio de notificações pontuais.

Figura 1 – Estrutura de dependências de programa escrito em PON.



Fonte: Autoria própria (2021).

A implementação usada neste projeto foi desenvolvida como um *framework* C ++. Vale ressaltar que não é voltado para a "programação de robôs", e apesar de ser a última versão do *framework*, de forma alguma representa o melhor desempenho. O melhor desempenho está sendo obtido atualmente com LINGPON (RONSZCKA et al., 2017), a linguagem de programação dedicada do paradigma (SIMÃO, s.d.).

2.4 PON aplicado em sistemas autônomos

O principal objetivo de aplicar o paradigma emergente ao código na competição deste ano é verificar se o PON é um bom ajuste para " programação de robôs", comparando seus pontos fortes e fracos com o código do ano passado.

Cada decisão tomada pelo programa que controla os robôs é executada usando o *Framework* do PON. Por exemplo, o seguinte trecho de código escrito em linguagem de pseudo-programação ilustra como o comando para parar os três robôs funciona na versão atual do programa.

Listing 1 – Exemplo completo de código em LINGPON.

```
fbe Game
 attributes
  string atGameState
 end_attributes
```

```
 methods
  method mtStopRobotI (...)
  method mtStopRobotII (...)
  method mtStopRobotIII (...)
 end_methods
end_fbe

inst
 Game game1
end_inst

strategy
 ...
end_strategy

rule rlShouldStop
 condition
  subcondition cnShouldStop
   premise prStop game1.atGameState == "STOP"
  end_subcondition
 end_condition
 action
  instigation inStopRobotI game1.mtStopRobotI (...)
  instigation inStopRobotII game1.mtStopRobotII (...)
  instigation inStopRobotIII game1.mtStopRobotIII (...)
 end_action
end_rule
```

2.5 Ferramentas para o PON

Como o PON é um novo paradigma, as ferramentas disponíveis para auxiliar no desenvolvimento de código extenso são limitadas. Pensando nisso, decidiu-se iniciar o desenvolvimento de um software que terá como objetivo auxiliar na detecção de bugs e na verificação do funcionamento do código.

O seu funcionamento baseia-se na tentativa de ilustrar uma linha do tempo, onde é possível ver todas as ações que modificam as entidades PON, como premissas, condições, entre outras, e também ver as respetivas reações no resto do código. Com isso em mente, foi pensada uma maneira de fazer isso da forma mais simples e direta, não envolvendo nenhuma interface gráfica, tendo como saída três relatórios feitos pelo sistema salvos em tempo real.

Esses três relatórios possuem um formato comum, sendo que cada um registra a hora e o nome de uma entidade PON sempre que uma delas é acionada, assim, é possível visualizar melhor o comportamento do código, de forma geral. Como são três relatórios, as três entidades do PON escolhidas para serem apresentadas são: premissas, condições e métodos.

Para futuras implementações e melhorias desta ferramenta, é possível melhorar sua exibição e expandir sua funcionalidade. Por exemplo, uma interface gráfica estilo linha do tempo mostraria como o código funciona

melhor. E caso você esteja rodando multi-threading, gerar múltiplos timelines, com a execução de cada um dos threads pode ajudar a encontrar bugs e inconsistências no código.

3 RESULTADOS

Com a utilização do novo Paradigma Orientado à Notificações para a elaboração da programação, algumas melhorias de desempenho durante as partidas entre o código do ano passado e este já podem ser observadas. O algoritmo de gerenciamento de robôs e funções mostrou-se funcional e ágil, assim como a estratégia aplicada para coordenação entre atacante e defensor.

4 CONCLUSÕES

A razão para o ganho de desempenho nas partidas ainda é desconhecida, visto que o algoritmo em PON leva alguns milissegundos a mais para processar um quadro, devido à implementação ser pelo *Framework* em C++, possivelmente. Apesar das boas observações, ainda é necessário fazer melhorias na área de detecção de colisões, que no novo paradigma ainda apresentava algumas divergências de funcionamento, e navegação de robôs. Para a parte de navegação do robô, um estudo está sendo realizado pelos membros da equipe para verificar o uso e adaptação dos algoritmos existentes que ajudam a guiar e controlar os motores dos robôs.

AGRADECIMENTOS

Os autores M. D. Freitas e J. A. Fabro agradecem à Fundação Araucária pela bolsa concedida ao primeiro, sob orientação do segundo.

REFERÊNCIAS

BANASZEWSKI, R. F. et al. Notification Oriented Paradigm (NOP): A Software Development Approach based on Artificial Intelligence Concepts. In: CITESEER. PROC. of Logic Applied To Technology. November 21-23 Santos/Brazil. [S.l.: s.n.], 2007. P. 216–222.

LINHARES, Robson Ribeiro; SIMAO, Jean Marcelo; STADZISZ, Paulo Cezar. NOCA - A Notification-Oriented Computer Architecture. **IEEE Latin America Transactions**, IEEE, v. 13, n. 5, p. 1593–1604, 2015.

RONSZCKA, A. F. et al. Notification-Oriented Programming Language and Compiler. In: 2017 VII Brazilian Symposium on Computing Systems Engineering (SBESC). [S.l.: s.n.], nov. 2017. P. 125–131. DOI: 10.1109/SBESC.2017.23.

SIMÃO, J. M. **A Contribution to the Development of a HMS simulation tool and Proposition of a Meta-Model for Holonic Control**. 2005. Tese (Doutorado) – School in Electrical Engineering, Industrial Computer Science (CPGEI) at Federal University of Technology - Paraná (UTFPR, Brazil) e Research Center For Automatic Control of Nancy (CRAN) - Henry Poincaré University (UHP, France).

SIMÃO, J.M. **Paradigma Orientado a Notificações (PON)**. [S.l.: s.n.]. Disponível em: http://www.dainf.ct.utfpr.edu.br/ jeansimao/PON/PON.htm. Acesso em: 23/08/2021.