1
INTRODUÇÃO

1.1
Motivação

Com o avanço tecnológico, problemas de controle cada vez mais complexos têm surgido em diversas áreas como sistemas de produção, automação industrial, e sistemas automotivos, para citar apenas alguns. Neste contexto, o controle tradicional, onde é necessário que o programador anteveja todas as possíveis situações, torna-se inviável. Deste modo, sistemas autônomos com comportamento inteligente, que incorporam a capacidade de decidir que ação tomar para satisfazer seus objetivos, tornam-se uma possível solução para estes problemas.

Agentes são sistemas de controle autônomos capazes de perceber o ambiente por meio de sensores e interagir com este ambiente através de atuadores (Jenning & Wooldridge, 1997). A autonomia é a capacidade de o agente seguir seus objetivos automaticamente, sem interações ou comandos externos.

Para aumentar a capacidade autônoma de um agente, é interessante que este seja dotado de inteligência (Brenner et al., 1998). O comportamento inteligente é dada pela sua capacidade de aprender, criar uma base de conhecimento e raciocinar. O agente deve ser capaz de reunir informações acerca do ambiente no qual está interagindo, tomar decisões baseadas nestas informações e realizar ações em concordância com suas decisões para atingir seus objetivos. No contexto desse trabalho, a inteligência é um pré-requisito da autonomia.

Sistemas como estes têm sido concebidos através do uso de técnicas como Lógica Fuzzy (LF) (Mendel, 1995) e Redes Neurais (RN) (Haykin, 1998), em áreas onde a abordagem convencional não tem conseguido fornecer soluções satisfatórias. Diversos pesquisadores vêm tentando integrar estas duas técnicas, gerando modelos híbridos que associam as vantagens de cada abordagem e minimizam suas limitações. A ideia básica de um sistema híbrido Neuro-Fuzzy é implementar um Sistema de Inferência Fuzzy (SIF) numa arquitetura paralela distribuída de tal forma que os paradigmas de aprendizado comuns às RNs possam ser aproveitados pela arquitetura (Figueiredo, 2003).

Os modelos Neuro-Fuzzy tradicionais, como *Adaptive-Network-Based Fuzzy Inference System* (ANFIS) (Jang, 1993), *Neuro-Fuzzy Classification* (NEFCLASS) (Kruse & Nauk, 1995) e *Fuzzy Self-Organing Map* (FSOM) (Vuorimaa, 1994), ajustam os parâmetros de uma estrutura pré-definida, ou têm uma capacidade muito limitada de ajuste em sua estrutura. De forma a evitar o problema de explosão do número de regras, estes modelos têm restrições quanto ao número máximo possível de entradas do sistema e seu desempenho é prejudicado com o aumento deste (Figueiredo, 2003).

Uma outra técnica bastante utilizada é o Aprendizado por Reforço (*Reinforcement Learning* – RL) (Sutton & Barto, 1998). Este algoritmo de aprendizado permite ao agente fazer a exploração dos estados e ações para determinar qual ação deve ser aplicada em cada estado diretamente a partir da interação com o ambiente. Ou seja, quando a informação usada no aprendizado não está disponível, é possível realizar o processo de aprendizado em sistemas Neuro-Fuzzy por meio do algoritmo de RL. Diversos modelos de agentes utilizam o Aprendizado por Reforço associado a métodos como *Q-Learning* (Watkins, 1989) e SARSA (Sutton & Barto, 1998).

Os modelos RL mais simples são baseados em uma tabela de armazenamento das funções de valor para o espaço de estados que é consultada para a escolha da melhor ação a ser adotada (*lookup table*). Quando o agente está inserido em ambientes pequenos e/ou discretos, os resultados obtidos utilizando *lookup table* são satisfatórios (Weiss, 1993). Entretanto, quando o ambiente é grande e/ou contínuo, a aplicação de métodos desta tabela torna-se inviável, devido à grande dimensão do espaço de estados (Moore, 1991; Jouffe, 1998; Sutton & Barto, 1998). Para contorná-lo, alguma forma de generalização deve ser incorporada à representação de estados. A generalização consiste em produzir modificações nos métodos RL por intermédio de aproximação de funções, permitindo a atualização através de reforços não apenas do estado relacionado à iteração atual, mas também de outros estados correlacionados por alguma característica comum (Brown, 1995; Sutton, 1996; Ribeiro, 1999).

Técnicas inteligentes como Redes Neurais e Lógica Fuzzy são utilizadas como aproximadores universais de funções (Sutton, 1996; Jouffe, 1998). No entanto, a maioria dos modelos atuais necessita de um grande número de informações ou de definições prévias. As principais limitações dos modelos

baseados em RL e LF, tais como *Neuro-Fuzzy Controller* (NEFCON) (Nauck & Kruse, 1994; Nürnberg et al., 1999), *Evolutionary Logic Fuzzy* (ELF) (Bonarini, 1996b), *Fuzzy-Actor-Critic-Learning* (FACL) (Jouffe, 1998) e *Fuzzy-Q-Learning* (FQL) (Glorennec & Jouffe, 1997) estão na necessidade de pré-definição do número de conjuntos fuzzy, das funções de pertinência destes conjuntos (usados nos antecedentes e consequentes das regras fuzzy), da pré-definição do número de regras, dos antecedentes que compõem as regras e de suas limitações quanto ao número de variáveis de entrada no modelo (Figueiredo, 2003).

O modelo *Reinforcement Learning* – Neuro-Fuzzy Hierárquico Politree (*Reinforcement Learning Neuro-Fuzzy Hierarchical Politree* – RL-NFHP) (Figueiredo, 2003; Figueiredo et al., 2004; Figueiredo et al., 2005) estudado neste trabalho ameniza algumas das limitações dos modelos existentes, realizando particionamento recursivo e aprendizado baseado em RL aplicado a ambientes contínuos e/ou de alta dimensão.

O modelo RL-NFHP apresenta as seguintes características: aprendizado automático da estrutura do modelo; auto-ajuste dos parâmetros associados à estrutura; capacidade de aprender a ação a ser tomada quando o agente está em um determinado estado do ambiente; possibilidade de lidar com um número maior de entradas do que os sistemas neuro-fuzzy tradicionais; e geração automática de regras linguísticas com hierarquia.

A principal desvantagem deste modelo é o fato do processo de aprendizado ser extremamente lento, uma vez que o agente não possui qualquer conhecimento prévio a respeito do ambiente nem qualquer pré-definição a respeito das características do modelo.

A motivação deste trabalho foi, portanto, desenvolver novas técnicas que contribuíssem para que o modelo RL-NFHP aprendesse mais rapidamente, servindo como mais uma ferramenta no rol de modelos que equipam agentes com comportamento inteligente. O uso deste modelo de aprendizado por reforço em ambientes grandes ou contínuos constitui uma grande vantagem sobre os demais: é desnecessário o conhecimento da dinâmica do ambiente, precisando-se apenas de um mínimo de definições prévias.

1.2 Objetivo

O trabalho destinou-se a aprimorar o modelo *Reinforcement Learning Neuro-Fuzzy Hierarchical Politree* (RL-NFHP) proposto por Figueiredo (2003) através de novas técnicas que acelerem o processo de aprendizado. A análise de desempenho do modelo modificado foi realizada em aplicações simuladas, para verificar o ganho real das novas estratégias propostas.

Outro objetivo deste trabalho, foi a implementação e análise de desempenho do modelo modificado em um agente real. O comportamento deste agente é importante para analisar a influência de incertezas e ruídos nas leituras dos sensores, variação entre os comandos enviados e executados pelos atuadores e possibilidade de treinamento simulado.

1.3 Descrição do Trabalho e Contribuições

Com o intuito de acelerar o processo de aprendizado, foram desenvolvidos quatro métodos de aceleração:

- interrupção do processo de aprendizado através de *early stopping*;
- nova política de seleção de ação *Q-DC-roulette*;
- *elegibility trace* cumulativo;
- poda da estrutura RL-NFHP.

O método *early stopping* é um procedimento de parada automática do aprendizado, por meio da avaliação da convergência do algoritmo.

A política de seleção de ação *Q-DC-roulette* visa mesclar os benefícios do aprendizado através das funções de valor *Q* com a possibilidade de exploração de acordo com o quanto cada ação foi visitada.

O método de rastros de eligibilidade (*eligibility trace*) cumulativo, faz com que a ação *n* passos a frente influencie os passos antigos.

Já o método de poda da estrutura consiste em remover células criadas e não visitadas por meio de um conceito de tempo de vida, diminuindo o tamanho da estrutura do modelo.

A fim de validar estes métodos de aceleração, foram realizados diversos experimentos em três estudos de casos:

- *benchmark* simulado carro na montanha;
- *benchmark* simulado Khepera;
- robô real *MindStorms*.

Foram desenvolvidos dois ambientes computacionais adequados de simulação. Implementou-se o *benchmark* carro na montanha para a avaliação das alterações do modelo, permitindo a realização de diversas experiências de forma a avaliar a potencialidade e a aplicabilidade destas modificações. Outro *benchmark* desenvolvido foi o Khepera, onde as melhores combinações de parâmetros foram testadas em um ambiente mais complexo.

Foi implementado o modelo RL-NFHP no comportamento inteligente de um robô explorador real Lego *MindStorms* NXT para que o mesmo navegasse em certo ambiente a procura de uma posição final desejada, desviando de obstáculos. A utilização do robô real foi importante para estabelecer um paralelo entre o ambiente simulado e o mundo real. Ao longo deste texto, quando da citação Lego, entenda-se como LEGO®.

No decorrer do desenvolvimento dos métodos de aceleração, o código computacional de Figueiredo (2003) foi inteiramente reescrito, reduzindo estruturas cíclicas e recursivas. A concepção da organização dos objetos na linguagem Java foi repensada de modo a facilitar a generalização do modelo.

1.4
Estrutura do Trabalho

Este trabalho está estruturado em cinco capítulos, conforme descrito a seguir:

O atual Capítulo 1, introdutório, descreve primordialmente a motivação e o objetivo.

O Capítulo 2 apresenta em detalhes o modelo de *Reinforcement Learning Neuro-Fuzzy Hierarchical Politree* (RL-NFHP), descrevendo o mecanismo de particionamento, a célula básica RL-NFP, a arquitetura e o algoritmo de aprendizado utilizado.

No Capítulo 3, são discutidas as melhorias sugeridas neste trabalho para o modelo RL-NFHP, como a interrupção do processo de aprendizado através de *early stopping*, nova política de seleção de ação (*Q-DC-roulette*), *elegibility trace* cumulativo, poda da estrutura, e reestruturação do código computacional.

O Capítulo 4 mostra os resultados obtidos em dois benchmarks simulados e a implementação deste algoritmo em um robô real. Inicialmente são realizados diversos testes no benchmark carro na montanha a fim de verificar a sensibilidade do modelo mediante variação de alguns parâmetros e validar as mudanças sugeridas. Posteriormente as melhores combinações de parâmetros são aplicadas no benchmark simulado mais complexo Khepera. Finalmente o melhor modelo obtido é testado em um robô Lego *MindStorms* NXT.

No Capítulo 5 são apresentadas as conclusões do trabalho e sugestões para estudos futuros.