2
MODELO REINFORCEMENT LEARNING NEURO-FUZZY HIERARCHICAL POLITREE (RL-NFHP)

2.1
Introdução

Este capítulo apresenta o modelo *Reinforcement Learning Neuro-Fuzzy Hierarchical Politree* (RL-NFHP) desenvolvido por Figueiredo (2003). Este modelo possui autonomia de aprendizado, através da interação direta com o ambiente, e também a transparência (interpretabilidade) dos sistemas de inferência fuzzy. Estas características já são inerentes aos sistemas neuro-fuzzy, entretanto, muitos destes apresentam aprendizado supervisionado, têm sua estrutura criada de forma limitada ou tem a sua estrutura pré-fixada inicialmente, exigem conhecimento específico sobre a modelagem, ou geram sistemas que não podem ser interpretados. O modelo RL-NFHP implementa a característica inteligência de um agente a partir da interação direta com o ambiente, gerando automaticamente sua própria estrutura e aprendendo, por meio de algoritmos de RL, a ação mais adequada para cada estado identificado nas células que compõem a estrutura. Os particionamentos recursivos, que serão discutidos na seção 2.2, viabilizaram esses objetivos.

O modelo RL-NFHP é uma extensão do modelo *Reinforcement Learning Neuro-Fuzzy Hierarchical Binary Space Partitioning* (RL-NFHB) também desenvolvido por Figueiredo (2003). O modelo RL-NFHP é composto por células padrão chamadas RL Neuro-Fuzzy Politree. Essas células são dispostas numa estrutura hierárquica, na qual a célula de maior hierarquia gera a saída. As saídas das células de menor hierarquia são os consequentes das células de maior hierarquia. No entanto, neste modelo, a designação neuro não se deve ao fato do modelo usar rede neural com treinamento supervisionado e atualização dos pesos baseada em gradiente decrescente, e sim de sua estrutura em forma de árvore.

No modelo híbrido RL-NFHP, o algoritmo de aprendizado SARSA (Sutton & Barto, 1998), explicado na subseção A.5.1, foi mantido. Posteriormente, Flesch (2009) apresentou um estudo comparativo entre este método de aprendizado e o aprendizado por reforço baseado no método *Q-Learning*

(Watkins, 1989), exposto na subseção A.5.2, mostrando a vantagem do primeiro em relação ao menor número de ciclos despendidos por episódio após a etapa de aprendizado.

As regras são geradas por um processo automático de particionamento de regiões do espaço que apresentam informações deficientes ou que exigem um refinamento maior, o que minimiza o problema de explosão combinatória de regras.

A função do componente fuzzy do modelo é agregar estados que têm comportamentos similares, associando-os a uma mesma ação. O componente RL faz com que o modelo aprenda a encontrar a ação mais adequada a ser executada para um determinado estado. O aspecto hierárquico deste modelo refere-se ao fato de que cada partição do espaço de entrada define um subsistema que, por sua vez, pode ter como consequente outro subsistema com a mesma estrutura (recursividade). Esta característica suaviza o processo de generalização das funções de valor (a árvore fuzzy hierárquica como aproximadora de função), não comprometendo os resultados, o que é um ótimo requisito para a convergência do processo (Sutton, 1996).

Similarmente aos Sistemas Fuzzy, os Sistemas Neuro-Fuzzy também mapeiam regiões fuzzy do espaço de entrada em regiões fuzzy do espaço de saída, através das regras fuzzy do sistema. As regiões fuzzy do espaço de entrada/saída são determinadas no processo de identificação da estrutura. Os métodos de particionamento mais utilizados pelos SNF encontrados na literatura são: *Fuzzy Grid*, *Adaptive Fuzzy Grid*, *Fuzzy Boxes* e *Fuzzy Clusters* (Souza et. al., 2002).

O particionamento do espaço de entrada/saída tem grande influência no desempenho de SNF nos aspectos de acurácia, generalização e geração automática de regras. A utilização de métodos recursivos de particionamento das entradas/saídas resultou em uma nova classe de SNF, denominada Sistemas Neuro-Fuzzy Hierárquicos (Souza, 1999).

2.2 Particionamento Politree

O particionamento Politree é uma extensão do particionamento Quadtree (Finkel & Bentley, 1974). No particionamento Quadtree o espaço é sucessivamente dividido em 4 regiões, que, por sua vez podem ser novamente subdivididos em 4 regiões em uma operação recursiva, a fim de detalhar o espaço de estado para melhor descrever a natureza do problema. A limitação do particionamento Quadtree está no fato de este trabalhar apenas em espaços bidimensionais. Isto pode ser contornado pela extensão para casos n-dimensionais. Por exemplo, no caso de dimensão *n*=3, temos o particionamento "Oct-tree" (Tamminen, 1984; Arvo, 1988) que divide o espaço em 8 subespaços. O particionamento utilizado no modelo proposto neste trabalho denominado *Reinforcement Learning Neuro-Fuzzy Hierarchical Politree* é uma generalização desta ideia. Nele hiperespaços de dimensão n são subdividido em 2^n subespaços.

A figura 2.1.a ilustra este tipo de particionamento para o caso de duas dimensões, e a figura 2.1.b mostra a representação da árvore Politree, neste caso igual à Quadtree.

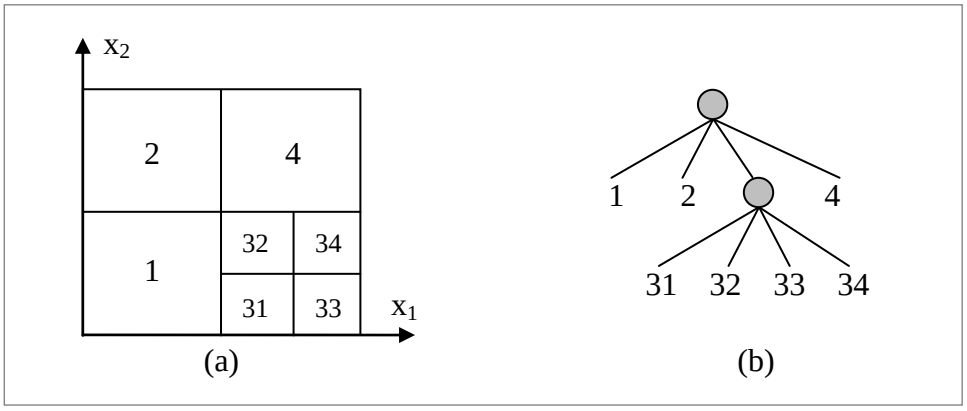

Figura 2.1: (a) Exemplo de particionamento Politree para espaços de dimensão 2 e (b) árvore representativa deste particionamento.

Como dito anteriormente, o modelo RL-NFHP é composto de uma ou várias células padrão chamadas células RL-Neuro-Fuzzy Politree (RL-NFP). Estas células são dispostas numa estrutura hierárquica em forma de árvore. A célula de maior hierarquia gera a saída. As de menor hierarquia trabalham como consequentes das células de maior hierarquia. Estas células são descritas em detalhes na seção 2.3, a seguir.

2.3 Célula Básica RL-Neuro-Fuzzy Politree

Uma célula RL-NFP é um mini-sistema neuro-fuzzy que realiza um particionamento politree em um determinado espaço, utilizando, para cada variável de entrada, as funções de pertinência. A célula RL-NFP gera uma saída exata (*crisp*) após um processo de defuzzificação, conforme será mostrado posteriormente.

As células RL-NFP formam uma estrutura hierárquica que resulta nas regras que compõem o raciocínio do agente. Para estes modelos, os antecedentes das regras são definidos pelas variáveis de entrada que estão associadas a dois conjuntos fuzzy. Os valores das variáveis de entrada são obtidos por meio de manipulação dos dados lidos pelos sensores do agente e são inferidos nos conjuntos fuzzy dos antecedentes. Se o antecedente é verdadeiro, a regra é disparada. Os consequentes são as ações que o agente deve aprender ao longo do processo e são realizadas pelos seus atuadores. Sendo assim, o modelo RL-NFHP cria e determina sua estrutura mapeando estados em ações.

A figura 2.2, a seguir, foi criada para facilitar a compreensão do processo de defuzzificação da célula e o encadeamento dos consequentes. Apenas para efeito de ilustração, na representação da célula serão apresentadas duas entradas (Quadtree) tornando o desenho mais simples do que a forma n-dimensional proposta para o Politree. As entradas x_1 e x_2 geram os antecedentes das quatro regras fuzzy após serem computados os graus de pertinência $\rho_1(x_1)$, $\mu_1(x_1)$, $\rho_2(x_2)$ e $\mu_2(x_2)$, onde: ρ_1 é o conjunto nebuloso *baixo e* μ_1 é o conjunto nebuloso *alto* relativos à entrada x_1; e ρ_2 é o conjunto nebuloso *baixo e* μ_2 é o conjunto nebuloso *alto* relativos à entrada x_2. Os valores definidos como consequentes são conjuntos de ações (a_1, a_2, ... a_t), onde cada ação está associada a uma função de valor *Q* (seção A.3). Através de método de aprendizado baseados em RL, uma ação de cada polipartição (a_k, a_j, a_p e a_q) será definida como aquela que representa o comportamento desejado do sistema quando o mesmo se encontra em um determinado estado.

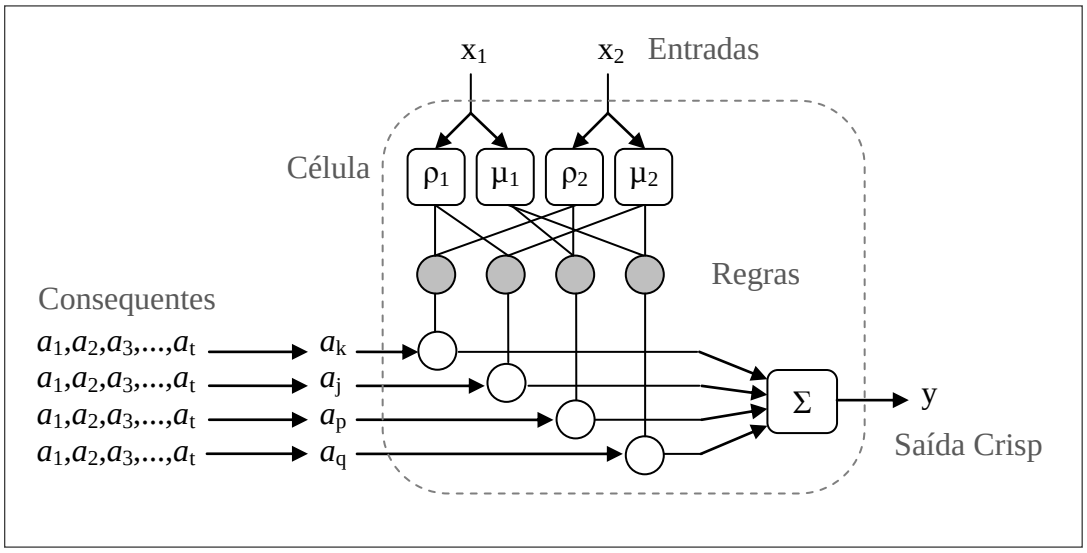

Figura 2.2: Célula Reinforcement Learning Neuro-Fuzzy Quadtree (Politree com *n*=2).

As expressões analíticas das funções de pertinência *alto* e *baixo* são dadas por sigmoides e seu complemento a '1' (eq. 2.1), respectivamente. As constantes *a* e *b* utilizadas na expressão analítica das funções de pertinência (FP) determinam, respectivamente, a inclinação do conjunto fuzzy e o ponto médio de transição entre os valores zero e um. Os perfis das FPs são ilustrados na figura 2.3. Perfis diferentes de sigmoides podem ser usados para essas funções de pertinência, entretanto neste trabalho, assim como no de Figueiredo (2003), este perfil foi escolhido por sua simplicidade.

$$\mu(x) = \frac{1}{1+e^{-(a(x-b))}} \quad e \quad \rho(x) = 1-\mu(x) \tag{2.1}$$

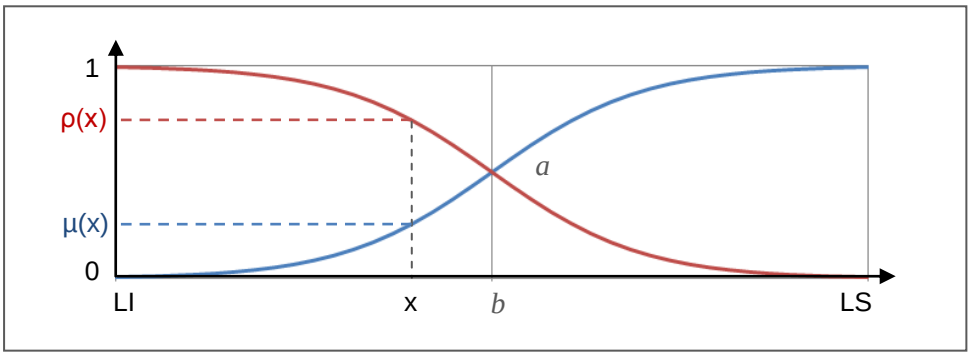

Figura 2.3: Exemplo de perfil das funções de pertinência da célula RL-NFP.

Na figura 2.4, estão mostradas as camadas de fuzzificação, de regras e de defuzzificação.

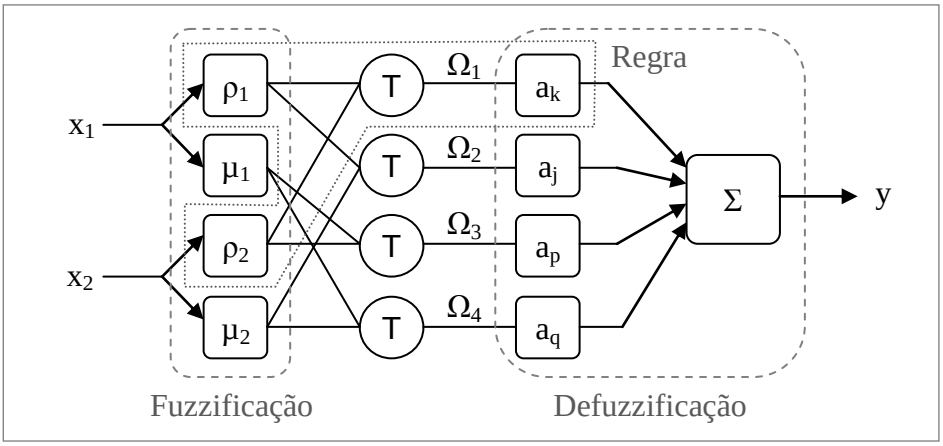

Figura 2.4: Célula RL-NFP representada sob o formato de rede neuro-fuzzy.

Estas funções de pertinência definem os perfis das funções *alto* (μ) e *baixo* (ρ) de cada variável de entrada. Os Ω_i (na figura 2.4) simbolizam os níveis de disparo das regras. Estes níveis de disparo são calculados usando-se uma operação AND (T-norm) sobre os graus de pertinência de ρ_1, μ_1, ρ_2 e μ_2, conforme descrito a seguir na eq. 2.2, onde o símbolo '*' representa a operação AND, que pode ser realizada pela multiplicação ou pela operação de mínimo entre os dois valores. Neste trabalho, tal como proposto por Figueiredo (2003), foi utilizado o operador de multiplicação.

$$\begin{aligned} \Omega_1 &= \rho_1(x_1) * \rho_2(x_2) \\ \Omega_2 &= \rho_1(x_1) * \mu_2(x_2) \\ \Omega_3 &= \mu_1(x_1) * \rho_2(x_2) \\ \Omega_4 &= \mu_1(x_1) * \mu_2(x_2) \end{aligned} \qquad (2.2)$$

A interpretação linguística do mapeamento implementado pela célula RL-NFP da figura 2.4 é dada pelo seguinte conjunto de regras:

- Regra 1: Se $x_1 \in \rho_1$ e $x_2 \in \rho_2$ então y = a_k .
- Regra 2: Se $x_1 \in \rho_1$ e $x_2 \in \mu_2$ então y = a_j .
- Regra 3: Se $x_1 \in \mu_1$ e $x_2 \in \rho_2$ então y = a_p .
- Regra 4: Se $x_1 \in \mu_1$ e $x_2 \in \mu_2$ então y = a_q .

Cada regra corresponde a um quadrante da figura 2.5. Quando o valor das entradas incide sobre o quadrante 1, é a regra 1 que tem maior nível de disparo. Quando a incidência é sobre o quadrante 2, é a regra 2 que tem maior nível de

disparo. No caso das entradas caírem no quadrante 3, é a regra 3 que tem maior nível de disparo e, finalmente, quando a incidência é sobre o quadrante 4, é a regra 4 que tem maior nível de disparo. Cada quadrante por sua vez pode ser subdividido em quatro partes, através de outra célula RL-NFP. É muito importante lembrar que os consequentes a_i não são valores predeterminados; eles fazem parte de um conjunto de ações que deve ser explorado para que se possa determinar, por meio de aprendizado por reforço, a ação mais adequada para cada regra.

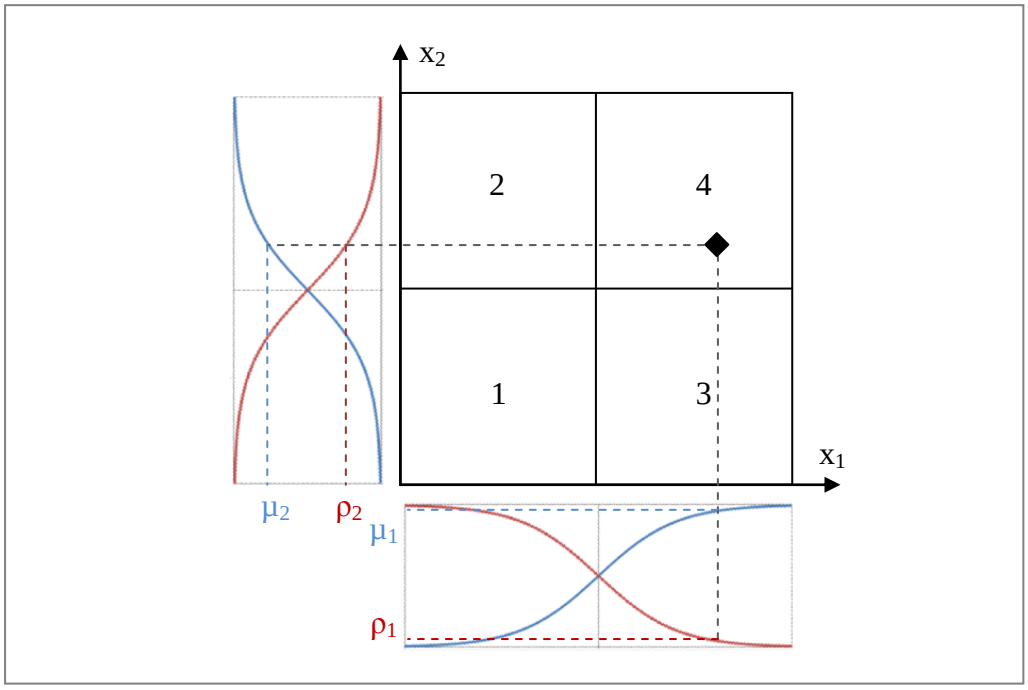

Figura 2.5: Divisão em quadrantes realizada pelas FPs alto e baixo com entrada incidindo sobre o quadrante 4.

A saídas *y* da célula RL-NFP, como mostrada nas figuras 2.2 e 2.4, é um número obtido pela média ponderada mostradas na eq. 2.3:

$$y = \left(\frac{\sum_{i=1}^{2^n} \Omega_i . a_i}{\sum_{i=1}^{2^n} \Omega_i} \right) \tag{2.3}$$

onde *n* é o número de entradas e a_i corresponde a um dos dois consequentes possíveis abaixo:

- *um singleton* (consequente *fuzzy singleton*, ou Sugeno de ordem zero), caso em que a_i = constante;
- *saída de um estágio de nível anterio*r, caso em que $a_i = y_j$, onde y_j representa a saída de uma célula genérica *j*, cujo valor é calculado, também, pela eq. 2.3.

Apesar de o consequente *singleton* ser simples, este não é conhecido previamente. É através do algoritmo RL que será possível determinar o melhor valor *singleton* (ação) para esta regra. Ou seja, os consequentes de cada regra são representados por um conjunto de ações relacionado àquele estado. O estado atual do agente é definido pelos valores das variáveis de entrada, que tornam ativas as células cujos domínios dos conjuntos de pertinência delimitam este estado.

Como mencionado, na célula RL-NFP básica as funções de pertinência são implementadas por sigmoides (ρ e μ) e por seu complemento a um. A utilização dos complementos a um leva a uma simplificação no procedimento de defuzzificação realizado pelo processo de média ponderada (eq. 2.3), pois o somatório do denominador é igual a um para quaisquer valores de entrada x_i. Desta forma, a saída da célula básica fica simplificada, como mostra a eq. 2.4.

$$y = \sum_{i=1}^{2^n} \Omega_i . a_i \qquad (2.4)$$

2.4 Arquitetura RL-NFHP

A arquitetura do modelo RL-NFHP é composta pela interligação entre as células básicas descritas acima. Isto é exemplificado na figura 2.6. A árvore Politree referente ao particionamento da fig. 2.6.a é mostrada na fig. 2.6.b. Cada partição não subdividida é chamada de polipartição.

Na árvore da figura 2.6.b os nós simbolizados com pequenos círculos são nós interiores e representam regiões que foram subdivididas. Os nós simbolizados por pequenos quadrados são nós terminais, ou folhas, e representam as polipartições, isto é, as regiões que não sofreram subdivisões, consequentes. A raiz da árvore simboliza todo o espaço a ser particionado.

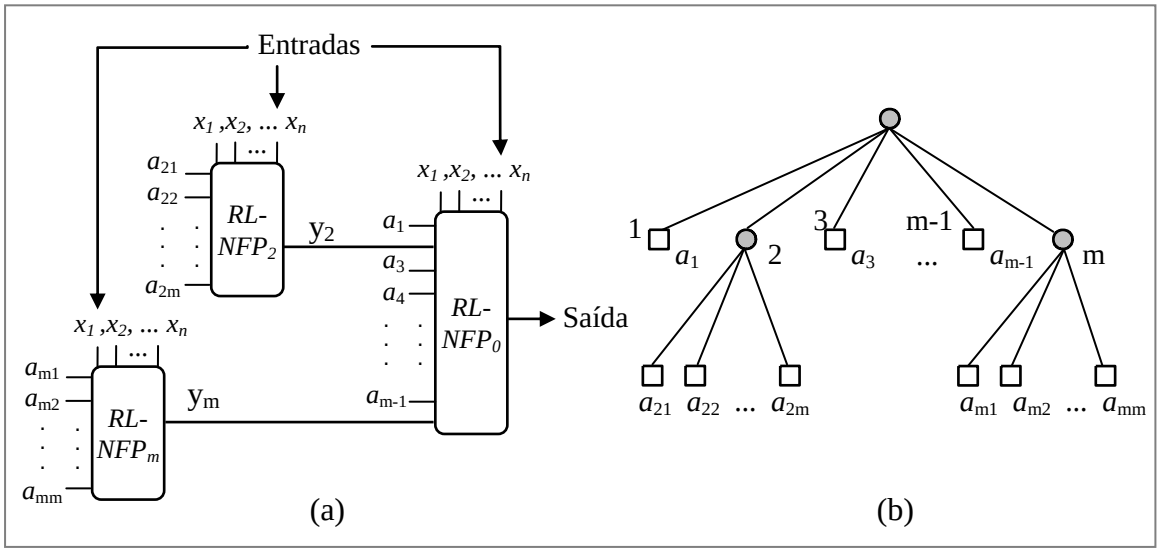

Figura 2.6: (a) Exemplo de arquitetura RL-NFHP e (b) Representação genérica em árvore do modelo RL-NFHP com n entradas e três células (círculos), onde cada célula possui m = 2^n consequentes (quadrados).

No exemplo da figura 2.6 as polipartições 1, 3 e *m*-1 não foram subdivididas, portanto os consequentes de suas respectivas regras são os valores a_1, a_3 e a_{m-1}. As partições 2 e *m* foram subdivididas e os consequentes de suas regras são as saídas (y_2 e y_m) dos subsistemas 2 e *m*. Estes por sua vez têm, como consequentes, os valores a_{21}, a_{22}, ..., a_{2m}, e a_{m1}, a_{m2}, ..., a_{mm}, respectivamente. Cada a_i corresponde a um consequente de Sugeno de ordem '0' (*singleton*), representando a ação que será identificada (dentre as ações possíveis), por meio de aprendizado por reforço, como sendo a mais favorável para um determinado estado do ambiente.

A saída do sistema da figura 2.6 é dada pela eq. 2.5, a seguir.

$$y = \Omega_1.a_1 + \Omega_2 \sum_{i=1}^{m} \Omega_{2i}.a_{2i} + \Omega_3.a_3 + \Omega_4.a_4 + \ldots + \Omega_m \sum_{i=1}^{m} \Omega_{mi}.a_{mi} \qquad (2.5)$$

De uma forma genérica, a equação de saída de um sistema RL-NFHP de dois níveis completos é dada pela eq. 2.6. Neste caso houve necessidade de se incluir as variáveis k_i e k_{ij}. Essas variáveis assumem apenas valores iguais a '0' ou '1', indicando a existência ou não das polipartições de ordem *i* e *ij*, respectivamente.

$$y = \sum_{i=1}^{2^n} \Omega_i k_i a_i + \sum_{i=1}^{2^n} \sum_{j=1}^{2^n} \Omega_i \Omega_{ij} k_{ij} a_{ij} \qquad (2.6)$$

Expandindo a eq. 2.6 para um sistema RL-NFHP de quatro níveis de hierarquia tem-se a eq. 2.7:

$$
\begin{aligned}
y = & \sum_{i=1}^{2^n} \Omega_i k_i a_i + \\
& \sum_{i=1}^{2^n} \sum_{j=1}^{2^n} \Omega_i \Omega_{ij} k_{ij} a_{ij} + \\
& \sum_{i=1}^{2^n} \sum_{j=1}^{2^n} \sum_{p=1}^{2^n} \Omega_i \Omega_{ij} \Omega_{ijp} k_{ijp} a_{ijp} + \\
& \sum_{i=1}^{2^n} \sum_{j=1}^{2^n} \sum_{p=1}^{2^n} \sum_{q=1}^{2^n} \Omega_i \Omega_{ij} \Omega_{ijp} \Omega_{ijpq} k_{ijpq} a_{ijpq}
\end{aligned}
\tag{2.7}
$$

onde:

- Ω_i , Ω_{ij} , Ω_{ijp} , Ω_{ijpq} , são os níveis de disparo das regras de cada polipartição *i*, *ij*, *ijp*, ou *ijpq*, respectivamente;
- k_i , k_{ij} , k_{ijp} , k_{ijpq} , são iguais a ‘1’ se as respectivas partições *i*, *ij*, *ijp* ou *ijpq* existem e ‘0’ caso contrário;
- a_i , a_{ij} , a_{ijp} , a_{ijpq} , são os consequentes (*singletons*) das regras existentes.

Na equação da expressão geral de saída do modelo RL-NFHP, descrita acima, já se levou em consideração a simplificação causada pelo uso das funções de pertinência complementares ($\rho + \mu = 1$) no método de defuzzificação das saídas de cada subsistema neuro-fuzzy.

O conjunto de regras que traduz o conhecimento linguístico do exemplo da figura 2.6 é:

Se $x_1 \in \rho_1$ e $x_2 \in \rho_2 \ldots x_n \in \rho_n$ então y = a_1
Se $x_1 \in \mu_1$ e $x_2 \in \rho_2 \ldots x_n \in \rho_n$ então
{
Se $x_1 \in \rho_{21}$ e $x_2 \in \rho_{22} \ldots x_n \in \rho_{2n}$ então y = a_{21}
Se $x_1 \in \mu_{21}$ e $x_2 \in \rho_{22} \ldots x_n \in \rho_{2n}$ então y = a_{22}
Se $x_1 \in \mu_{21}$ e $x_2 \in \mu_{22} \ldots x_n \in \rho_{2n}$ então y = a_{23}
.
.
.
Se $x_1 \in \mu_{21}$ e $x_2 \in \mu_{22} \ldots x_n \in \mu_{2n}$ então y = a_{2m}
}
Se $x_1 \in \mu_1$ e $x_2 \in \mu_2 \ldots x_n \in \rho_n$ então y = a_3
Se $x_1 \in \mu_1$ e $x_2 \in \mu_2 \ldots x_n \in \rho_n$ então y = a_4
.
.
.
Se $x_1 \in \mu_1$ e $x_2 \in \mu_2 \ldots x_n \in \mu_n$ então
{
Se $x_1 \in \rho_{m1}$ e $x_2 \in \rho_{m2} \ldots x_n \in \rho_{mn}$ então y = a_{m1}
Se $x_1 \in \mu_{m1}$ e $x_2 \in \rho_{m2} \ldots x_n \in \rho_{mn}$ então y = a_{m2}
Se $x_1 \in \mu_{m1}$ e $x_2 \in \mu_{m2} \ldots x_n \in \rho_{mn}$ então y = a_{m3}
.
.
.
Se $x_1 \in \mu_{m1}$ e $x_2 \in \mu_{m2} \ldots x_n \in \mu_{mm}$ então y = a_{mm}
}

onde:

- $\rho_1, \rho_2, \ldots, \rho_n$ e $\mu_1, \mu_2, \ldots, \mu_n$, são as funções de pertinência *baixo* e *alto* que definem a polipartição de nível 1;
- $\rho_{21}, \rho_{22}, \ldots, \rho_{2n}$ e $\mu_{21}, \mu_{22}, \ldots, \mu_{2n}$, são as funções de pertinência que definem as subdivisões da polipartição 2;
- $\rho_{m1}, \rho_{m2}, \ldots, \rho_{mn}, \mu_{m1}, \mu_{m2}, \ldots, \mu_{mn}$, são as funções de pertinência que definem as subdivisões da polipartição *m*.

2.4.1
Antecedentes das Regras do Modelo RL-NFHP

A figura 2.7 mostra a estrutura de aprendizado do agente. A leitura do ambiente (s_1, s_2,..., s_n) é feita pelo agente através de seus sensores e estas leituras podem ser traduzidas em um ou mais valores de entrada (x_1, $x_{2, \ldots}$ x_n) das células. Os valores x_i são avaliados nas células, podendo disparar regras. Sendo assim, toda vez que uma regra é disparada, ou, dito de outra forma, uma célula se torna ativa, o processo de aprendizado identifica que o agente está em um estado definido pelo domínio dos conjuntos fuzzy do antecedente da regra (domínio da entrada da célula).

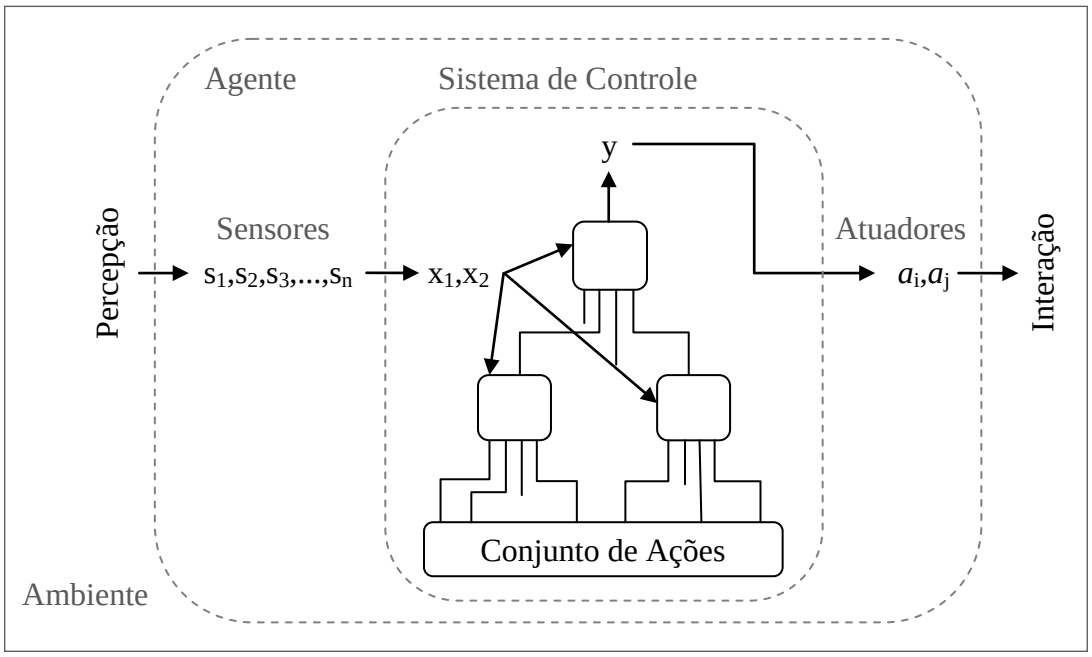

Figura 2.7: Esquema do processo de aprendizado do agente

2.4.2
Consequentes das Regras do Modelo RL-NFHP

Os consequentes das regras fuzzy nas células RL-NFP são as ações que devem ser identificadas por meio de aprendizado por reforço. Quando as células se tornam ativas, as ações são selecionadas, cada uma relativa à combinação *baixo* e *alto* de cada uma das entradas da célula. A seleção ocorre em função de valores atribuídos a cada uma das ações que pertencem ao conjunto de ações disponíveis para cada polipartição *baixa* e *alta*.

Novamente, apenas para efeito de ilustração, na figura 2.8 são apresentadas duas entradas (Quadtree), tornando o desenho mais simples do que a forma n-dimensional proposta para o Politree. Esta figura mostra a célula RL-NFP com 4 conjuntos de ações – associados aos suas respectivas funções de valor *Q* (seção A.3) –, onde cada conjunto está relacionado às polipartições de cada célula. Cada conjunto pode possuir um número de ações *t*, independentemente do número de entradas.

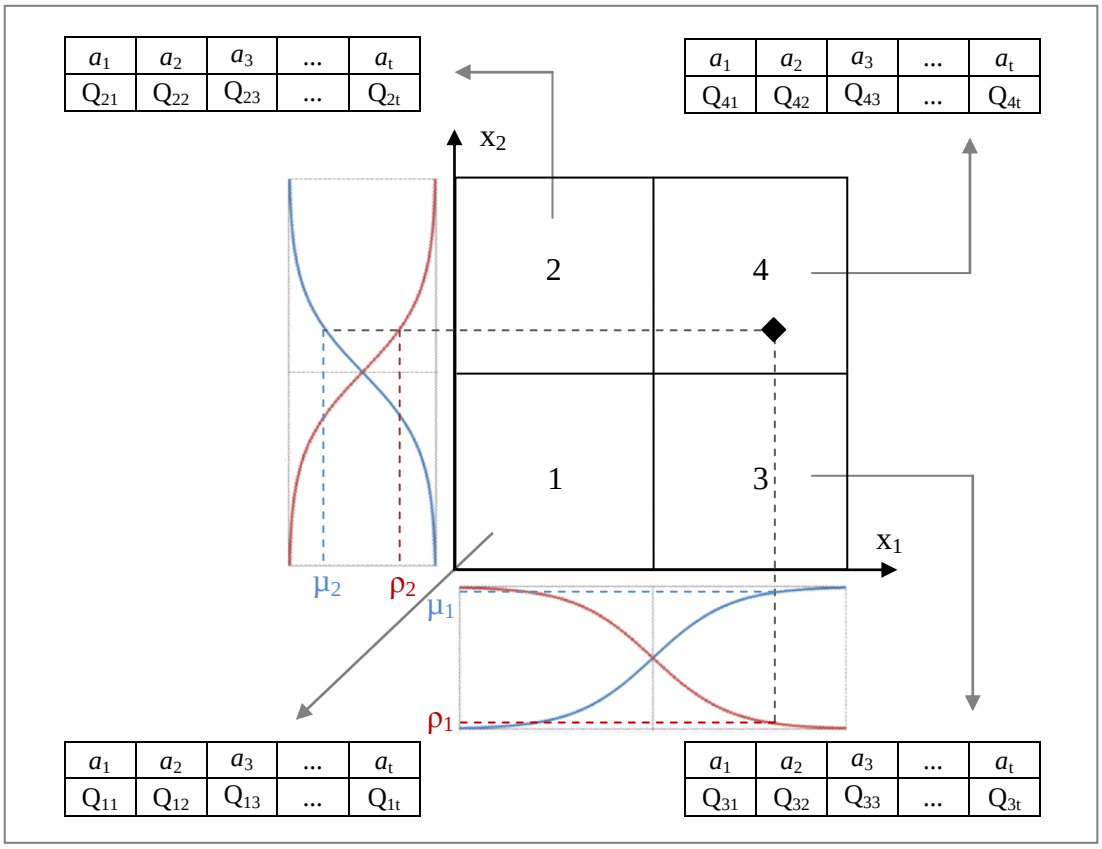

Figura 2.8: Interior da célula RL-NFP com duas entradas (Quadtree).

Os consequentes serão do tipo ‘1’ se a célula for uma folha da estrutura e serão do tipo ‘2’ se forem células intermediárias.

2.4.2.1 Consequente Tipo 1: *Singleton*

É o tipo mais simples de consequente de uma regra fuzzy. As regras fuzzy com este consequente são mostradas abaixo.

Se $x_1 \in \rho_1$ e $x_2 \in \rho_2$ então $y = a_k$

Se $x_1 \in \rho_1$ e $x_2 \in \mu_2$ então $y = a_j$

Se $x_1 \in \mu_1$ e $x_2 \in \rho_2$ então $y = a_p$

Se $x_1 \in \mu_1$ e $x_2 \in \mu_2$ então $y = a_q$

A vantagem do uso deste tipo de consequente está na facilidade do cálculo da saída que, neste caso, é geralmente efetuado através da média ponderada (eq. 2.3). Este tipo de consequente também é conhecido como consequente de *Sugeno* de ordem ‘0’.

No modelo RL-NFHP, apesar do método de cálculo para este tipo de consequente ser simples, o valor do consequente não é conhecido a priori. Um dos objetivos do modelo é, portanto, aprender, por meio do algoritmo baseado no SARSA (Sutton, 1998), a identificar as ações associadas aos conjuntos fuzzy *baixo* e *alto* da célula RL-NFP que melhor respondam ao estado atual do agente.

2.4.2.1 Consequente Tipo 2: Célula RL-NFHP

Este tipo de consequente é, na verdade, a saída de outro mini-sistema RL- Neuro-Fuzzy Hierárquico Politree implementado por uma célula RL-NFP. Isto gera a hierarquia inerente aos sistemas neuro-fuzzy hierárquicos. As regras fuzzy com este consequente são como exposto na seção 2.4.

2.5 Algoritmo de Aprendizado

O processo de aprendizado começa com a definição das entradas relevantes, para o sistema/ambiente no qual o agente está inserido, e dos conjuntos de ações que ele pode dispor para atingir seus objetivos. O fluxograma exibido na figura 2.9 descreve o algoritmo de treinamento do modelo RL-NFHP. As nove fases correspondentes à numeração no fluxograma são descritas nas subseções seguintes.

2.5.1 Criação do Controle RL-NFHP

Uma célula raiz é criada, tendo como domínios dos seus conjuntos fuzzy relativos a cada uma das entradas, os valores mínimo e máximo destas entradas (Limite Inferior – LI e Limite Superior – LS). Com o objetivo de generalidade, os valores das variáveis de entrada são normalizados. Os valores correspondentes às variáveis de entrada da célula são lidos do ambiente por meio dos sensores, normalizados e podem ser aplicados diretamente às entradas da célula ou ser modificados segundo uma função que os torne adequados às variáveis das entradas da célula.

As funções de valor *Q* iniciais associadas às ações das polipartições também devem ser definidas. Normalmente, as aplicações que utilizam SARSA ou *Q-Learning* iniciam suas funções de valor *Q* com zero ou de maneira aleatória. Neste trabalho funções de valor *Q* foram iniciadas com valores iguais positivos próximos de zero para facilitar na construção da política *Q-roulette* que será explicada no capítulo 3.

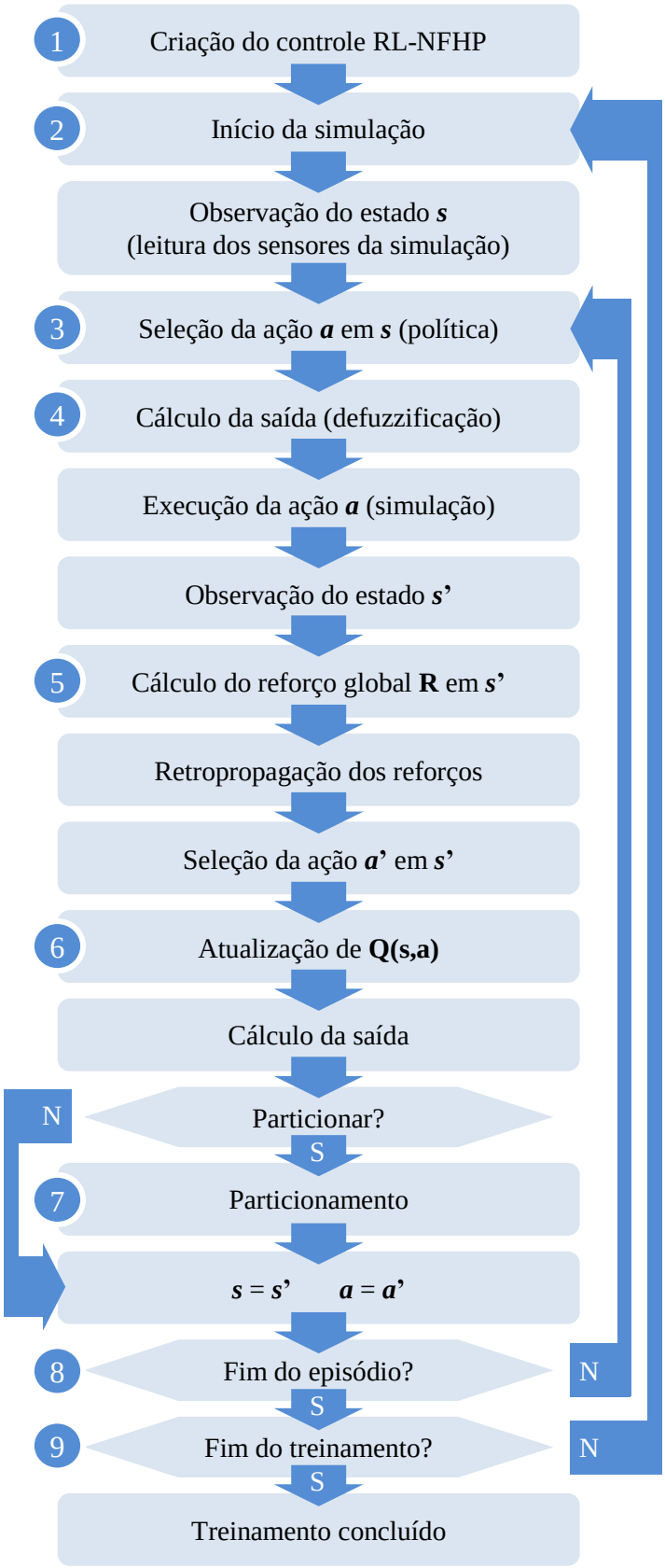

Figura 2.9: Fluxograma de treinamento.

2.5.2 Início da Simulação

A simulação deve ser capaz de emular a física do ambiente real, recebendo como entrada ações (comandos para os atuadores) e gerando como saída percepções do ambiente (leitura dos sensores).

O agente é colocado em uma posição inicial definida pelo programador. Esta posição inicial pode e deve ser alterada depois de alguns episódios de modo que o agente seja submetido a partes do espaço diferentes durante o treinamento.

2.5.3 Seleção das Ações

As ações estão associadas às funções de valor *Q* e compõem um conjunto de comandos que são selecionados e experimentados durante o aprendizado RL. A exploração do espaço de estados é fundamental à descoberta de ações que correspondam à melhor resposta do agente (que visa atingir um objetivo associado a maiores valores *Q*) quando este se encontra em um determinado estado do ambiente.

Idealmente, no início do processo de aprendizado ou treinamento, o parâmetro ε associado à política de escolha de ação deveria ter um valor que permitisse mais exploração (*exploration*) e menos aproveitamento (*exploitation*) e, à medida que o sistema aprendesse, deveria permitir menos exploração e mais aproveitamento (seção A.4). No entanto, como ocorrem inclusões de células na estrutura ao longo do processo de aprendizado, o que seria indicado, nestas circunstâncias, é que estas novas células também tivessem oportunidade de explorar suas ações. Sendo assim, foi definido, para cada polipartição da célula, um parâmetro denominado ε, cujo valor deve estar em [0,1], possibilitando variar a política de escolha da ação deste modelo.

Com o objetivo de melhorar ainda mais o desempenho de aprendizado do modelo, o parâmetro ε foi definido como adaptativo. Este procedimento adaptativo para o parâmetro ε é usado no método de aprendizado por reforço AHC (Sutton & Barto, 1998). Neste procedimento, quando a ação resultante é boa, o

valor do parâmetro ε desta polipartição é reduzido, aumentando a possibilidade de esta polipartição usar uma política de aproveitamento (seleção da ação associada a maior função de valor *Q* ou roleta baseada em *Q*). Quando a ação resultante não apresenta um bom desempenho, as partições das células ativas têm os seus parâmetros ε aumentados, permitindo que estas partições, nos passos seguintes, tenham maiores chances de usarem um método de exploração (seleção da ação aleatória ou através de roletas baseadas em visita).

Figueirero (2003) utiliza no aprendizado do modelo RL-NFHP a política de *ε-greedy* (Sutton & Barto, 1998), que seleciona a ação associada a maior função de valor *Q* esperada com probabilidade p = 1-ε, seleção gulosa; e com probabilidade p = ε seleciona aleatoriamente uma ação qualquer.

2.5.4 Cálculo da Saída

A cada passo, a ação é escolhida (subseção 2.5.3) em cada polipartição de todas as células ativas. As ações são combinadas, a partir das células folhas até a célula raiz, gerando uma ação resultante que alimentará o simulador. A ação é combinada conforme o exemplo para uma dada entrada (x_1,x_2) no modelo RL-NFHP da figura 2.10. A figura 2.11 mostra a principal polipartição 33.

A figura 2.10 ilustra duas células com duas entradas. Sendo assim, cada célula possui 4 polipartições, cada uma relativa à combinação *baixo*/*alto* dos graus de pertinência avaliados pelas entradas x_1 e x_2. Os valores correspondentes às ações escolhidas em cada polipartição da célula RL-NFP$_0$ são $a_{01,}$ a_{02} , y_{1} , a_{04} , e são calculados mediante os graus de pertinências correspondentes aos Ω_i relativos à célula RL-NFP$_0$. $a_{11,}$ a_{12} , a_{13} , a_{14} são os valores das ações da célula RL-NFP$_1$ que possui como saída y_1. A saída do modelo RL-NFHP é dada por y.

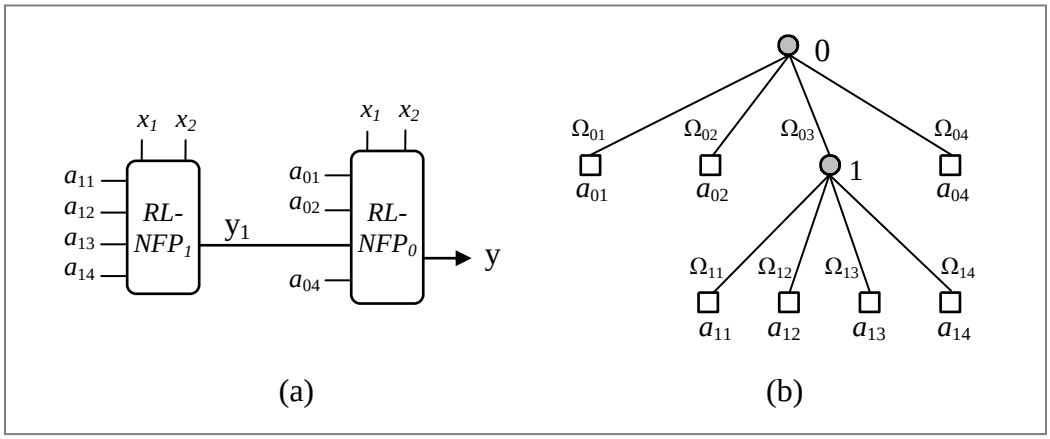

Figura 2.10: (a) Exemplo de arquitetura RL-NFHP com 2 entradas e
(b) sua representação em árvore. Esta estrutura possui duas células (círculos),
onde cada célula tem 4 consequentes (quadrados).

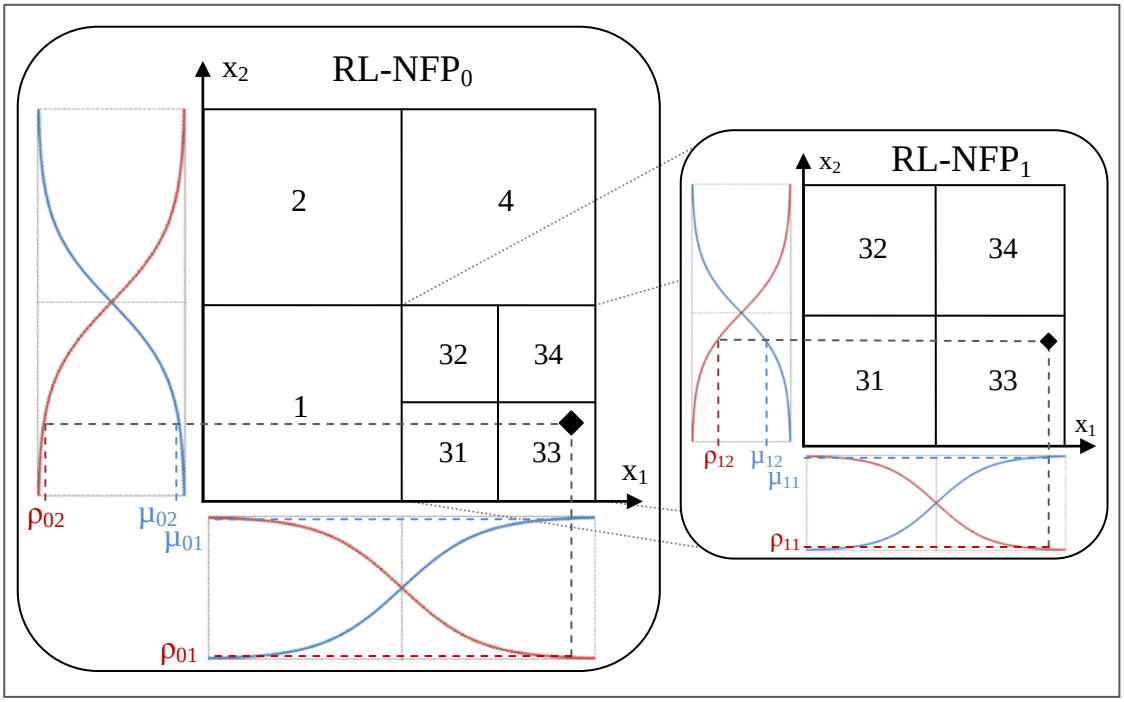

Figura 2.11: Exemplo de arquitetura RL-NFHP de 2 entradas, mostrando os
graus de pertinência dos conjuntos *fuzzy* das duas células que compõe a estrutura.
Os valores Ω_i são calculados usando-se uma operação AND (T-norm).

Os cálculos dos Ω_i, mostrados na figura 2.10, são realizados pela eq. 2.12:

$$\begin{aligned}
\Omega_{01} &= \rho_{01}(x_1) \,.\, \rho_{02}(x_2) \\
\Omega_{02} &= \rho_{01}(x_1) \,.\, \mu_{02}(x_2) \\
\Omega_{03} &= \mu_{01}(x_1) \,.\, \rho_{02}(x_2) \\
\Omega_{04} &= \mu_{01}(x_1) \,.\, \mu_{02}(x_2) \\
\Omega_{11} &= \rho_{11}(x_1) \,.\, \rho_{12}(x_2) \\
\Omega_{12} &= \rho_{11}(x_1) \,.\, \mu_{12}(x_2) \\
\Omega_{13} &= \mu_{11}(x_1) \,.\, \rho_{12}(x_2) \\
\Omega_{14} &= \mu_{11}(x_1) \,.\, \mu_{12}(x_2)
\end{aligned} \tag{2.12}$$

O cálculo da saída y_1 da célula RL-NFP$_1$ é dada pela média ponderada pelos Ω_i's das ações escolhidas em cada polipartição.

$$y_1 = \sum_{i=1}^{4} \Omega_{1i} . a_{1i} \tag{2.13}$$

A saída y do modelo RL-NFHP é a saída da célula raiz (RL-NFP$_0$), dada pela soma ponderada pelos Ω_i's dos consequentes e do mini-sistema formado pela célula RL-NFP$_1$.

$$y = \Omega_{01} . a_{01} + \Omega_{02} . a_{02} + \Omega_{03} . y_1 + \Omega_{04} . a_{04} \tag{2.14}$$

Para que fique claro, é mostrado outro exemplo de entrada (figura 2.12) para o mesmo modelo RL-NFHP da figura 2.10 onde a polipartição principal é a 1. Agora a célula RL-NFP$_1$ não está ativa.

Neste caso o que muda são os valores de Ω_i, que são corrigidos para que sua soma seja igual a 1 (eq. 2.15).

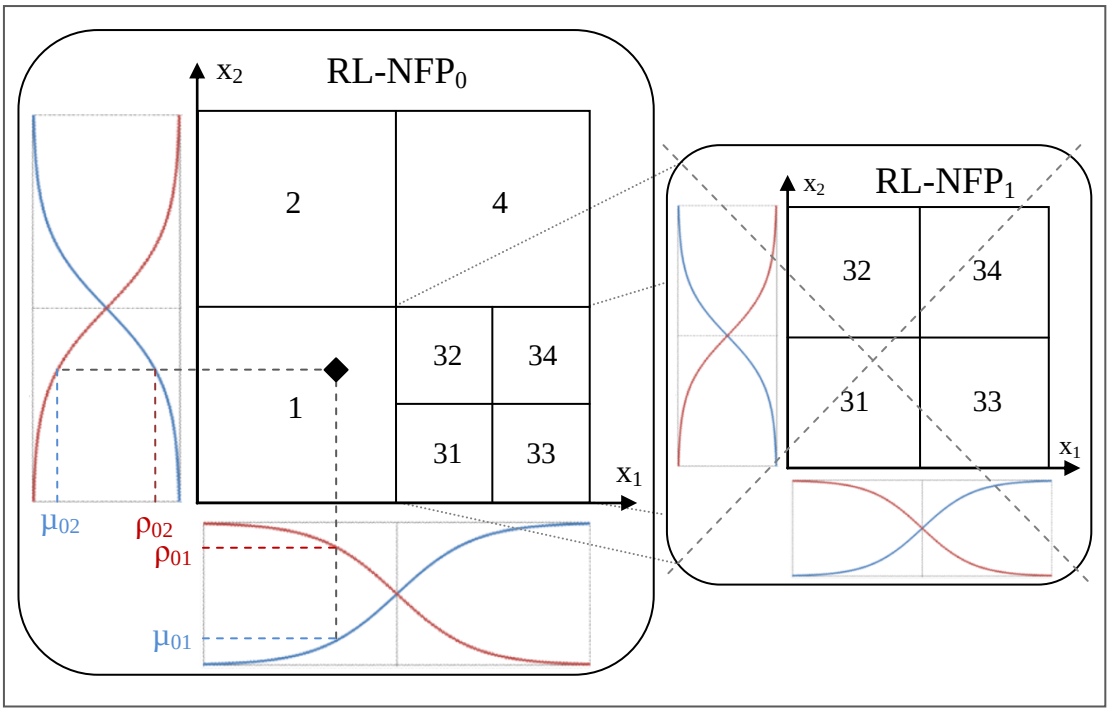

Figura 2.12: Exemplo de arquitetura RL-NFHP de 2 entradas, mostrando os graus de pertinência dos conjuntos fuzzy das duas células que compõe a estrutura.

$$\Omega_{01} = \frac{\rho_{01}(x_1).\rho_{02}(x_2)}{\sum_{i=1}^{4}\Omega_{0i}}$$

$$\Omega_{02} = \frac{\rho_{01}(x_1).\mu_{02}(x_2)}{\sum_{i=1}^{4}\Omega_{0i}} \qquad (2.15)$$

$$\Omega_{03} = 0$$

$$\Omega_{04} = \frac{\mu_{01}(x_1).\mu_{02}(x_2)}{\sum_{i=1}^{4}\Omega_{0i}}$$

Toda vez que existirem células filhas localizadas em partições não principais elas não estarão ativas e, por conseguinte, terão valores de Ω nulos.

Com a utilização de mais entradas, pode ocorrer que os valores Ω_i (resultado T-norm sobre os graus de pertinência relativos às entradas da célula) se tornem muito pequenos, o que faria o algoritmo ter um gasto maior de tempo computacional executando cálculos para a saída e atualizações das funções de valor que não teriam um peso significativo para o algoritmo. Sendo assim, foi acrescentado o conceito de *alfa-cut* com o objetivo de inibir na saída ações

relacionadas a polipartições cujos Ω_i sejam muito pequenos. Neste caso, o valor Ω_i desta polipartição torna-se igual a zero. Isso significa que, nesta iteração, esta polipartição (mesmo quando ativa) não contribuirá na saída com sua ação, nem terá o sua função de valor *Q* (associada à ação selecionada) atualizada.

2.5.5 Cálculo do Reforço e Retropropagação

Após a execução da ação, uma nova leitura do ambiente é realizada. Esta leitura permite que seja calculado o valor de reforço do ambiente e se avalie a ação tomada pelo agente. Este valor deve ser calculado por meio de uma função de avaliação definida segundo os objetivos do agente, sendo fundamental para a orientação do agente ao longo do processo de aprendizado.

A cada passo, no processo de aprendizado, o reforço é calculado para cada polipartição de todas as células ativas, mediante a sua participação na ação resultante. Dessa forma, o reforço do ambiente é retropropagado a partir da célula raiz até as células folhas. A figura 2.13 ilustra a retropropagação para o exemplo de modelo RL-NFHP de duas células adotado na subseção 5.5.4 para a entrada dada da figura 2.11.

A figura 2.13.a mostra duas células com duas entradas. Sendo assim, cada célula possui 4 polipartições, cada uma relativa à combinação *baixo/alto* dos graus de pertinência avaliados pelas entradas x_1 e x_2. A figura 2.13.b mostra o reforço global do sistema representado pela letra R no topo da árvore. Os valores correspondentes aos reforços de cada polipartição da célula RL-NFP$_0$ são $R_{01,}$ R_{02} , R_{03} , R_{04} , e são calculados mediante os graus de pertinências correspondentes aos Ω_i relativos à célula RL-NFP$_0$. $R_{11,}$ R_{12} , R_{13} , R_{14} são os valores dos reforços locais da célula RL-NFP$_1$.

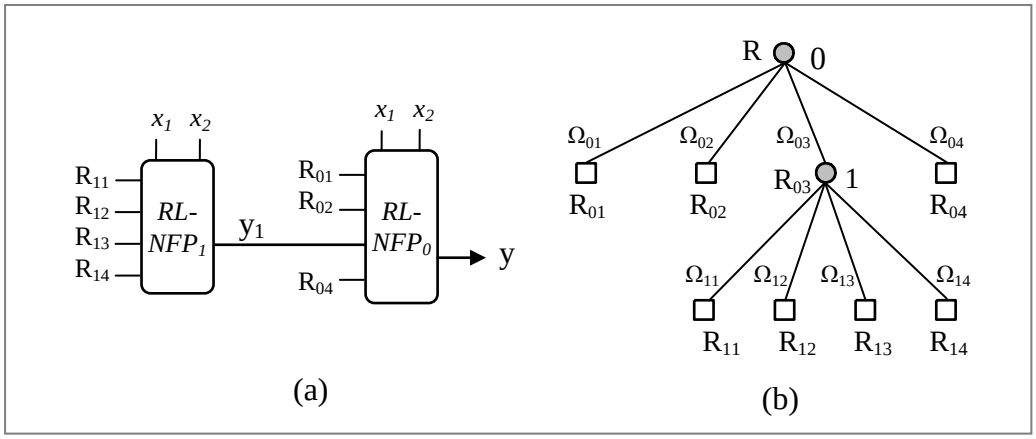

Figura 2.13: (a) Exemplo de arquitetura RL-NFHP com 2 entradas e (b) sua representação em árvore mostrando a retropropagação do reforço do ambiente. Esta estrutura possui duas células (círculos), onde cada célula tem 4 consequentes (quadrados).

Os cálculos dos reforços das partições da célula RL-NFP$_0$ são calculados a partir do reforço global e dos valores de Ω_i (subseção 2.5.4):

$$\begin{aligned} R_{01} &= \Omega_{01} \cdot R \\ R_{02} &= \Omega_{02} \cdot R \\ R_{03} &= \Omega_{03} \cdot R \\ R_{04} &= \Omega_{04} \cdot R \\ R_{11} &= \Omega_{11} \cdot R_{03} \\ R_{12} &= \Omega_{12} \cdot R_{03} \\ R_{13} &= \Omega_{13} \cdot R_{03} \\ R_{14} &= \Omega_{14} \cdot R_{03} \end{aligned} \qquad (2.16)$$

2.5.6 Atualização da Função Valor *Q*

Com os valores de reforços calculados para cada célula da estrutura, as funções de valor *Q* associadas às ações que tenham contribuído para a ação resultante executada pelo agente devem ser atualizadas.

Esta atualização é feita a partir da avaliação entre os reforços globais atual e anterior. A atualização das funções de valor *Q* ocorre de duas formas distintas: para o caso do valor do reforço global atual ser maior que o reforço global anterior ($R_{t+1} > R_t$) e para o caso do reforço global atual ser menor ou igual ao reforço global anterior ($R_{t+1} \leq R_t$).

2.5.6.1 Forma de Atualização 1: $R_{t+1} > R_t$

Caso o reforço global atual seja maior que o reforço global anterior, então as ações atuais (ações que foram executadas quando o agente estava no estado s_t) têm maiores chances de compor a melhor resposta do agente ao sistema quando o mesmo se encontrar neste estado. Assim, se $R_{t+1} > R_t$ deve-se premiar as ações selecionadas neste passo *t*, atualizando suas respectivas funções de valor *Q* conforme a equação do algoritmo SARSA (Sutton, 1996):

$$Q(s_t, a_t) = (1 - \alpha_t) \cdot Q(s_t, a_t) + \alpha_t [r_t + \gamma Q(s_{t+1}, a_{t+1})] \quad (2.17)$$

onde: o valor $Q(s_t, a_t)$ é atualizado a partir do seu valor atual; r_t é o reforço local imediato (este é o reforço da polipartição definido na subseção 2.5.5); o γ é um parâmetro que fixa um percentual da contribuição da função de valor *Q* associada à próxima ação a_{t+1} escolhida (termo $Q(s_{t+1}, a_{t+1})$) quando o sistema está no estado s_{t+1}; e α_t é o parâmetro proporcional à contribuição relativa desta ação local na ação global. A seguir o parâmetro α e a definição do valor $Q(s_{t+1}, a_{t+1})$ usados na eq. 2.17 serão detalhados.

- Parâmetro α

O parâmetro α está compreendido entre [0,1]. Na maioria das aplicações ele tem seu valor inicial igual a '1' e, à medida que o aprendizado evolui, seu valor é reduzido. No início do processo de aprendizado sua função é estimular os novos valores aprendidos a partir do reforço r_t e de $Q(s_{t+1}, a_{t+1})$ (Sutton & Barto, 1998). À medida que o processo de aprendizado evolui, o valor de α é reduzido, aumentando o peso da parcela relativa aos valores $Q(s_t, a_t)$ já aprendidos. No caso do modelo RL-NFHP, baseado em aproximação de funções, a redução deste parâmetro ao longo do processo resultou em graves problemas de aprendizado (Figueiredo, 2003).

Após vários testes, a avaliação que gerou melhores resultados foi a que definiu o parâmetro α como sendo proporcional à contribuição relativa desta ação local na ação global (Figueiredo, 2003). Como a punição e a premiação são

realizadas em condições distintas, a atualização da função de valor *Q* da polipartição que estiver contribuindo mais naquele passo também terá seu valor atualizado segundo esta proporção e a que possuir participação minoritária terá seu valor alterado na proporção desta participação.

A ação de saída da célula é resultado da contribuição das ações de seus dois consequentes. Caso a polipartição que está sendo atualizada tenha um grau de pertinência muito pequeno, mesmo que a ação não seja uma ação ideal, essa influência é minimizada. À medida que o agente se desloca no espaço de estados e cresce o grau de pertinência desta polipartição, a ação "não ideal" que antes tinha seu peso reduzido graças ao grau de pertinência menor, passa a acarretar uma saída "ruim". Por isso a atualização da função de valor *Q* (no que diz respeito ao reforço e ao valor de $Q(s_{t+1}, a_{t+1})$) deve depender do grau de importância que esta ação tem na saída.

Em outras aplicações, nas quais as modelagens diferem do tradicional RL (como a *lookup table*), os autores também ajustaram este parâmetro segundo as necessidades de seus modelos e obtiveram bons resultados. O próprio Sutton (1998), para a aplicação do carro da montanha usando o modelo CMAC, define o α como uma constante. Jouffe (1998); também utilizou uma forma adaptativa para α, na qual o parâmetro pode crescer ou decrescer segundo sua heurística definida para o aprendizado. Neste trabalho também será testado um valor de α fixo (seção 4.1).

- Valor de $Q(s_{t+1}, a_{t+1})$

Após a execução da ação resultante, o agente passa ao estado s_{t+1} do ambiente. Isso significa que pelo menos duas funções de valor *Q* (correspondentes às ações selecionadas a_{t+1} para as bipartições *baixo* e *alto*), no caso de modelos com apenas uma entrada, devem ser consideradas para $Q(s_{t+1}, a_{t+1})$ da eq. 2.17.

A figura 2.14 exemplifica esta situação. No estado s_t, a célula ativa apresenta duas funções de valor a serem atualizadas: Q_1 no conjunto de ações relativo à bipartição *baixo* e Q_2 no conjunto de ações relativo a bipartição *alto*. Quando o agente passa ao estado seguinte, s_{t+1}, após a execução da ação resultante, a célula que é ativada também seleciona duas ações a_{t+1}, cada uma

associada a sua função de valor *Q* (neste caso, Q_2 e Q_3). Para a atualização dos valores de Q_1 e Q_2 relativos ao estado s_t, existem dois métodos:

- no primeiro método, o valor $Q(s_{t+1}, a_{t+1})$ do estado s_{t+1} considerado na atualização é aquele que corresponder ao ramo da estrutura que apresentar maior peso na ação de saída; neste caso seria o valor de Q_2, se $\Omega_1 > \Omega_2$; ou Q_3, se $\Omega_2 > \Omega_1$;
- no segundo, o valor $Q(s_{t+1}, a_{t+1})$ é calculado a partir da soma ponderada de Q_2 e Q_3 com relação aos graus de pertinência da variável de entrada da célula $(\Omega_1.Q_2 + \Omega_2.Q_3)$.

Apesar dos resultados não diferirem significativamente (Figueiredo, 2003), o segundo método foi o adotado por apresentar resultados, em média, ligeiramente superior ao primeiro.

O valor $Q(s_{t+1}, a_{t+1})$ da ação escolhida para o estado s_{t+1}, que será usado para atualizar as funções de valor *Q* (relativos aos conjuntos *baixo* e *alto*) quando o agente está no estado s_t , também considera o peso que cada ação escolhida a_t no estado s_t teve na ação resultante.

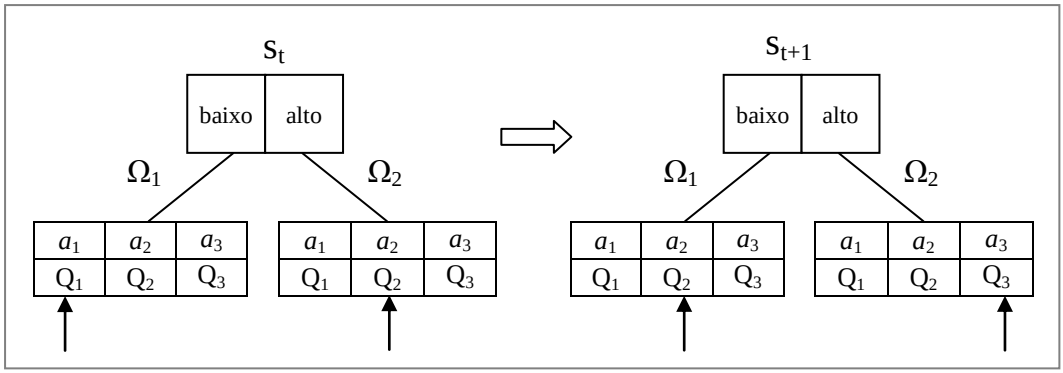

Figura 2.14: Caso exemplo de atualização da função de valor Q_{t+1} relativa à eq. 2.17.

Neste passo do algoritmo também é atualizada a taxa de exploração/aproveitamento já mencionada anteriormente dada pelo parâmetro ε. Cada polipartição deve ter o seu parâmetro ε reduzido/aumentado segundo um pequeno percentual (por exemplo, 5% do seu valor). Este procedimento segue as recomendações feitas por Sutton (1996). O parâmetro ε não deve ser superior a 40%, nem inferior a 5%.

2.5.6.1 Forma de Atualização 2: $R_{t+1} \leq R_t$

Caso o reforço global atual seja menor ou igual ao reforço global anterior, isto significa que as ações atuais não maximizam a função de reforço do estado em que o agente se encontra. Sendo assim, essas ações devem se tornar menos propensas a serem selecionadas nas próximas vezes em que as células, às quais elas pertencem, estiverem ativas.

Se $R_{t+1} \leq R_t$, as ações são punidas, reduzindo suas funções de valor *Q* na proporção da contribuição do reforço local desta polipartição da célula e o reforço global. A explicação para esta medida é a mesma já descrita antes. Caso a influência de uma ação boa seja minimizada pelo seu grau de pertinência neste momento, não se deseja que sua função de valor *Q* seja drasticamente reduzida, devido à má influência da ação da polipartição irmã. A função de valor *Q* é atualizada como na eq. 2.18:

$$Q(s_t, a_t) = (1 - fp) \,.\, Q(s_t, a_t) \tag{2.18}$$

onde *fp* é o fator de punição, que varia entre [0,1] e é definido como a relação entre o reforço local da polipartição e o reforço global.

Caso as ações escolhidas sejam mal sucedidas para um determinado estado, os parâmetros ε das partições envolvidas terão suas taxas aumentadas, permitindo que, nas próximas vezes que esta polipartição estiver ativa, outras ações diferentes tenham mais chances de serem escolhidas. Isso se aplica a qualquer dos métodos de seleção descritos.

Desta forma, é feito o aprendizado das ações que serão executadas quando o agente se encontra em um determinado estado. Este estado é definido pelas células que estão ativas a cada passo.

2.5.7 Particionamento

O crescimento da estrutura do modelo RL-NFHP, aumento do número de células, ocorre por meio do particionamento do espaço no qual o agente está inserido. Quando uma polipartição possuiu todos os requisitos necessários para o particionamento, uma célula filha é criada e conectada àquela polipartição. Seu domínio será o subdomínio correspondente à polipartição do seu ancestral. As células filhas herdam da polipartição ancestral o conjunto de ações e têm suas funções valor *Q* inicializadas com valores iguais próximos a zero.

Na figura 2.15, a célula raiz RL-NFP$_0$, ou célula pai, possui os domínios definidos pelo intervalo [LI,LS], limite inferior e superior, para as entradas x_1 , x_2 e x_3. A célula filha RL-NFP$_1$ descende da polipartição referente à composição do conjunto *alto* relativo à entrada x_1, ao conjunto *alto* da entrada x_2 e ao *alto* da entrada x_3 da célula raiz. Seus domínios são, portanto, diretamente relacionados ao subdomínio da polipartição *alto/alto/alto* da célula pai e são definidos por [(LI+LS)/2,LS] para as entradas x_1 , x_2 e x_3.

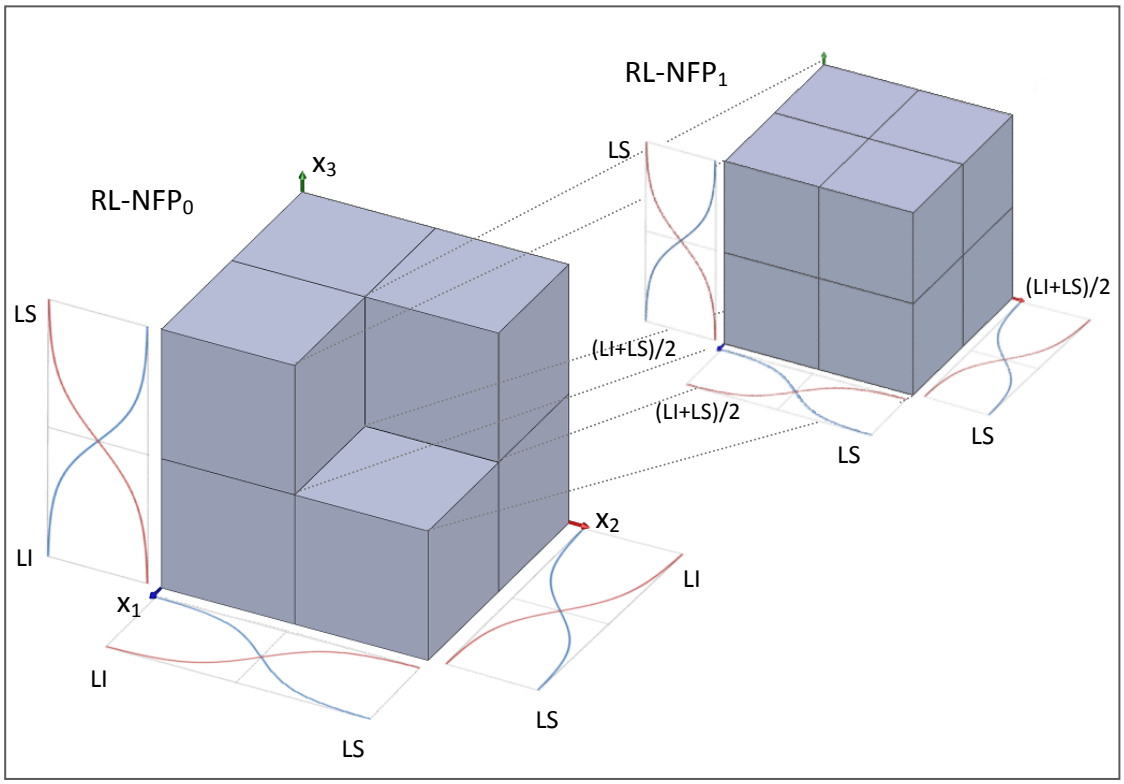

Figura 2.15: Particionamento da célula RL-NFP$_0$.

Quando uma célula (raiz ou pai) possuiu todas as células descendentes (todas as polipartições são células), a ação da célula pai torna-se a ação de saída da célula filha ativa. O domínio da célula pai passa a ser representado na célula filha com uma acurácia maior.

Com relação ao crescimento da estrutura, particionamento das células, algumas avaliações tornam-se necessárias para se garantir o aprendizado. Para isso, foram criados alguns critérios que devem ser atendidos conjuntamente para que seja permitido o particionamento:

- função de crescimento;
- desvio padrão de ΔQ;
- desvio padrão de visita.

Sendo assim, quanto mais rapidamente a polipartição da célula ultrapassar os limites de capacidade de aprendizado (por não conseguir aprender adequadamente), mais rapidamente ela será particionada, de forma a especializar o domínio da célula que não conseguiu atingir seus objetivos.

2.5.7.1 Função de Crescimento

O critério de função de crescimento é composto de duas partes: o parâmetro variável de crescimento β e a função de crescimento Φ propriamente dita. Esta variável e esta função têm como objetivo permitir ou limitar o crescimento da estrutura. À medida que as células são atualizadas, verifica-se o percentual de variação da função de valor *Q* (ΔQ) das ações associadas às partições das células. Quando a variação da função de valor *Q* associada à ação atualizada é maior que um percentual da maior variação já ocorrida para esta polipartição da célula, considera-se que esta polipartição apresenta potencial de crescimento, ou seja, esta variação pode indicar que as ações que estão sendo tomadas nesta polipartição não estão adequadas para o subdomínio relativo a esta polipartição. Logo:

- Se $\Delta Q > \psi . \Delta Q_{max}$, então β é incrementada em $\Delta\beta_1$.
- Se $\Delta Q < \psi . \Delta Q_{max}$, então β é decrementada em $\Delta\beta_2$.

onde: ΔQ é a variação da função de valor *Q* neste passo; $\psi \in [0,1]$ representa o percentual de variação permitido na atualização da função de valor *Q* em relação a maior variação já ocorrida até o momento; ΔQ_{max} registra a maior variação da função de valor *Q* da polipartição de uma célula ocorrida ao longo do aprendizado; β variável de crescimento para esta polipartição da célula; $\Delta\beta_1$ o valor de incremento; e $\Delta\beta_2$ o valor de decremento de β.

A definição do percentual ψ está associada diretamente à taxa de crescimento desta estrutura: quanto maior for o valor de ψ, maior variação de ΔQ será permitida nesta polipartição e menor o particionamento da estrutura.

O objetivo deste percentual é permitir que haja alguma variação na atualização da função de valor *Q* desta polipartição, mas sem prejudicar o aprendizado. Pequenas variações ocorrem principalmente no início do aprendizado (já que a função de valor *Q* inicial é próxima a zero) ou quando o processo de exploração escolhe uma ação diferente (com sua função de valor *Q* associada) que passa a maximizar o valor de reforço para aquele estado. No entanto, variações muito grandes indicam que a ação tomada nesta polipartição não responde adequadamente ao comportamento desejado (definido através do reforço do ambiente) às exigências deste estado (domínio), ou seja, esta ação não consegue atender ao comportamento desejado para o agente em todo este domínio, indicando que ele deve ser particionado.

Associada à variável critério de aprendizado deve-se definir a função de crescimento Φ (eq. 2.19), cujo objetivo é limitar o crescimento ao longo do processo de aprendizado. Idealmente, ela deve ser menos exigente com relação às células iniciais do sistema, ou seja, deve permitir que no início do aprendizado as células se multipliquem mais rapidamente e, à medida que o sistema evolui, deve crescer o grau de exigência da função, ou seja, aumentar o efeito de *exploration/exploitation* sobre as ações das células da estrutura. Isso se deve ao fato de que as ações das células criadas no início do aprendizado não são tão efetivas para domínios ainda muito abrangentes.

$$\Phi = \log(n \times \text{número de passos} \times \text{número de ciclos}) \qquad (2.19)$$

Conforme sugerido por Figueiredo (2003), a função de crescimento é função do número de passos, do número de ciclos e de um parâmetro de ajuste n ($n > 1$), como mostra a figura 2.16. O eixo y representa o valor limite para variável de crescimento.

A cada ciclo avalia-se se a variável de crescimento da polipartição de uma célula tornou-se maior do que a função de crescimento. Caso isso seja verdade, então esta polipartição apresenta a primeira condição para gerar uma célula filha.

Ao longo dos ciclos, a função de crescimento Φ aumenta o valor que a variável critério de aprendizado β deve atingir para que ocorra um novo particionamento. No entanto, a cada novo ciclo, a função de crescimento apresenta valor maior do que o valor do início do ciclo anterior e menor do que o valor do final do ciclo deste ciclo. Isso permite que as novas células criadas a cada ciclo também tenham chances de ser particionadas.

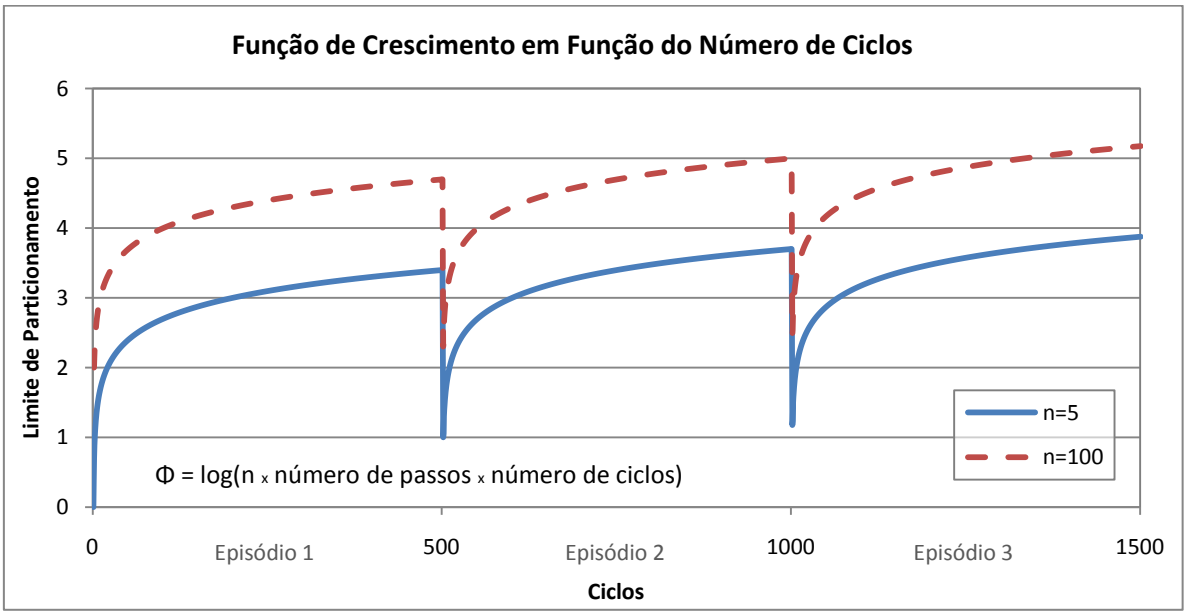

Figura 2.16: Função de crescimento.

Caso o aprendizado das ações de cada uma das polipartições das células não seja efetivo, ou seja, caso a variável de crescimento de qualquer uma das partições (ou mesmo das duas) da célula atinja o valor definido pela função de crescimento, a polipartição apresenta o primeiro requisito para o particionamento.

A função de crescimento apresenta um valor pequeno no início do aprendizado (reduzindo o grau de exigência para realizar o particionamento), quando os conjuntos fuzzy das células ainda são abrangentes, com relação ao

domínio, e maior ao final do processo (aumentando o grau de exigência para realizar o particionamento), quando os conjuntos fuzzy já se especializaram o suficiente para garantir o aprendizado.

2.5.7.2 Desvio Padrão de ΔQ

Neste critério a dispersão dos valores de variação da função valor *Q* (ΔQ) é avaliada para que se permita o particionamento das células, juntamente com os outros critérios adotados.

Os valores de ΔQ da polipartição da célula são armazenados em uma lista e seu desvio padrão σ e sua média $\overline{\Delta Q}$ são avaliados a cada visita a esta polipartição. Caso a média seja menor que dois desvios padrões ($\overline{\Delta Q} < 2\sigma$) o critério de particionamento está atendido.

A lista deve possuir um tamanho mínimo (neste trabalho fixado em 50 elementos) para que tenha sentido estatístico e um tamanho máximo (neste trabalho de 200 elementos) para evitar problemas de memória computacional.

Mecanismos semelhantes ao aqui descritos também foram usados em outros modelos por outros pesquisadores em sistemas de aprendizado baseados em árvores (Pyatt & Howe, 1998; Uther & Veloso, 1998).

2.5.7.3 Desvio Padrão de Visita

Para evitar que uma ou mais das possíveis ações da polipartição ainda não tenham sido escolhidas, ou tenham sido escolhidas poucas vezes, foi criado um critério baseado na visita.

Neste caso, há a possibilidade de a célula não ter sido bem explorada e ter um particionamento prematuro.

Este critério utiliza contadores de visita que guardam o número de vezes que determinada ação foi selecionada em cada polipartição. Estes contadores de visita são os mesmos utilizados para a política *DC-roulette* que será apresentada no capítulo 3.

A ideia é impedir o particionamento prematuro, se houver uma ou mais ações que não foram bem exploradas. Em outras palavras, se houver um desequilíbrio significativo na visita das ações. Se uma ou mais ações foram selecionadas com muito menos frequência do que outras, presume-se que a polipartição ainda não está desenvolvida o suficiente para justificar o particionamento. Deve-se então impedir que a estrutura RL-NFHP cresça desnecessariamente.

Flesch (2009) propõe duas abordagens para verificar se há uma distribuição desequilibrada das ações. A primeira versão do critério de igualdade de seleção SEC1 (*Selection Equality Criterion 1*) compara a visita normalizada com um valor determinado e pode ser formulada da seguinte forma:

$$\frac{C(s,a_i)}{\sum_i C(s,a_i)} > f \tag{2.20}$$

onde, C(s,a_i) é um contador de seleção da ação a_i na polipartição; $\sum_i C(s,a_i)$ é o número de visitas da polipartição; e *f* um fator determinado heuristicamente (por exemplo *f* = 0,01).

A segunda versão do critério de seleção SEC2 compara o desvio padrão do contador de visitas normalizado com outro valor determinado.

$$\frac{\sigma_C}{\sum_i C(s,a_i)} > g \tag{2.21}$$

onde σ_C é o desvio padrão das visitas às ações dentro da polipartição; $\sum_i C(s,a_i)$ é o número de visitas da polipartição; e *g* um fator determinado heuristicamente (por exemplo *g* = 0,1).

No RL-NFHP, um ou mais critérios de particionamento podem ser usados ao mesmo tempo a fim de agregar diferentes visões. Porém não há sentido em combinar as duas versões do critério de visita, uma vez que ambas as versões realizaram trabalho semelhante. Como sugerido por Flesch (2009), este trabalho utiliza a segunda versão do critério, pois o fator *g* é mais facilmente determinado heuristicamente que o *f*. Para determinar o valore de *g*, é necessário realizar alguns experimentos, sem nenhum dos critérios de seleção da igualdade e verificar o

valor do desvio padrão normalizado da seleção de ações. O resultado será um intervalo de valores relativamente pequeno, no qual é possível selecionar o desvio padrão máximo tolerado de acordo com a tarefa, quanto mais baixo for, menor será a estrutura Politree resultante (menor o número de células).

2.5.8 Fim do Episódio

O episódio termina quando se chega ao número máximo de ciclos ou o objetivo da simulação é alcançado.

Neste trabalho o número máximo de ciclos, ou passos, para alcance do objetivo em cada episódio, ou época, foi fixado em 10000, ou seja, na prática, ilimitado.

O objetivo varia de acordo com a simulação adotada e em geral é dado uma tolerância. A título de exemplo, pode-se mencionar a posição final determinada no ambiente na qual um robô deve chegar.

Outra forma de término de episódio pode ocorrer quando o agente não consegue sair de determinado estado: *lock position*. Esta forma é semelhante a se chegar ao número máximo de ciclos, com um benefício de tempo computacional. Neste caso ocorre o fim do episódio sem a conclusão da tarefa desejada.

2.5.9 Fim do Treinamento

O fim do treinamento ocorre quando a estrutura RL-NFHP deixa de mudar significativamente, ou seja, o agente aprende. Na prática, é difícil determinar o fim do treinamento. Pode-se treinar demasiadamente a estrutura, perdendo-se generalização e tempo, ou, ao contrário, treiná-la insuficientemente, impossibilitando o agente de cumprir seu objetivo ou de cumpri-lo de maneira insatisfatória.

Foi criado um procedimento automático de determinação de fim de treinamento: *early stopping* que será apresentado na seção 3.2.

2.6 Teste e Uso do Modelo RL-NFHP

Depois de realizado o treinamento do modelo RL-NFHP, ele deve ser testado com valores iniciais diferentes dos de treinamento para que se comprove seu real aprendizado e capacidade de generalização. Uma vez que o agente alcance o objetivo nestes testes, este modelo estará pronto para o uso.

No caso de utilização do modelo em agentes reais, geralmente é aconselhável seu início de treinamento em ambiente simulado. No mundo real existe a inviabilidade de realização de tantos episódios. A título de exemplo, no caso do aprendizado de um robô, é complicado reiniciar o episódio através da colocação do agente nas condições iniciais milhares de vezes. Após o aprendizado completo em ambiente simulado, deve-se ajustar as entradas e saídas do modelo RL-NFHP para o mundo real. Novamente deve-se testar o agente neste ambiente real em condições iniciais diferentes, para depois utilizar o modelo. Caso seja necessário, pode-se continuar o treinamento, que foi concluído na simulação, no ambiente real.