Adaptatividade: Generalização Conceitual

J. J. Neto

Resumo — **Este material refere-se ao conteúdo do mini-curso tutorial de mesmo nome, apresentado em 29 de janeiro de 2009 em São Paulo, por ocasião do WTA 2009 – Terceiro Workshop de Tecnologia Adaptativa. Faz-se aqui o estudo de alguns dos mais importantes aspectos da realização computacional das abstrações expressas através de formulações adaptativas, com a ajuda de ferramentas recentemente disponibilizadas. Pretende-se com isso proporcionar, aos interessados em utilizar a adaptatividade na prática, uma visão atualizada da situação em que se encontram os recursos teórico e tecnológico hoje disponíveis, e da maneira como se relacionam.**

Palavras-chave — **Adaptividade, Dispositivos Adaptativos, Comportamento Auto-modificável, Sistemas Adaptativos, Tecnologia Adaptativa.**

VIII. *Introdução*

O tema central em torno do qual costumam girar as discussões acerca dos assuntos ligados à adaptatividade é a forma como pode ser realizada a variabilidade funcional dos programas.

Em maior ou menor intensidade, quase todos os programas modernos bem projetados, e de porte razoável, têm sido construídos de tal forma que possam executar, de forma versátil, uma certa variedade de operações distintas.

De um para outro programa, podem-se constatar diferentes graus de visibilidade e de acesso do usuário à diversidade de funções que tais programas oferecem. Mais ainda, de programa para programa, variam muito as técnicas empregadas em sua construção.

Tais técnicas procuram dotar esses programas da possibilidade de proporcionarem a seus usuários o efeito da funcionalidade variável.

Programas pequenos e de uso restrito costumam ser projetados especificamente para apresentarem uma **funcionalidade constante**, sem opções de variação. Seu código, naturalmente estático, costuma ser muito simples. Seu comportamento, em conseqüência, é fixo e inflexível. A grande vantagem de programas dessa categoria é a simplicidade, que proporciona um baixo custo, mas uma eventual exigência posterior de recursos adicionais acarreta a necessidade de reprogramação, que permita incorporar as novas funcionalidades.

Programas ainda de pequeno porte, porém de propósito pouco menos restrito costumam ser projetados de tal forma que proporcionem mais flexibilidades para seus usuários. Apesar de executarem também uma única funcionalidade fixa, esta é materializada na forma de um procedimento paramétrico. Argumentos adequados instanciam os seus parâmetros, personalizando de forma prática seu comportamento a cada ativação dos procedimentos que os implementam.

Todos os que já se serviram de algum programa, procedimento, sub-rotina ou método paramétrico sabem que a parametrização agrega ao programa um potencial enorme, em relação a outro, que não se utilize de parâmetros.

Para que um programa não paramétrico possa executar operações diferentes daquela que ele realiza, torna-se necessário reescrevê-lo, ainda que na forma de uma cópia modificada. Por sua vez, basta fornecer diferentes conjuntos de dados a um programa paramétrico para que ele execute atividades diferentes, em correspondência aos particulares dados recebidos, sem que sejam promovidas quaisquer alterações em seu comportamento original.

Assim, neste caso particular, os parâmetros assumem o papel de agentes de personalização do comportamento do programa, a despeito de não ser variável a funcionalidade dos procedimentos com código estático.

Muitos programas oferecem uma variedade de funcionalidades fixas, freqüentemente parametrizadas, e permitem selecionar e executar uma delas de cada vez.

Uma situação ligeiramente mais complexa é a dos programas que executam diversas funcionalidades. Neste caso, seções diferentes são criadas na estrutura do programa, e se destinam a executar operações diferentes. Através de um seletor, pode-se escolher dentre essas diversas funcionalidades, aquela que se deseja executar.

Cada uma dessas seções pode, por sua vez, ser paramétrica, e com isso torna-se possível combinar as duas formas mencionadas de variação de funcionalidade, e dessa maneira, através de parâmetros e de dados de entrada que sejam especificamente usados para selecionar, torna-se possível uma escolha confortável da exata funcionalidade que se deseja do programa, a cada vez que este é executado.

Dessa forma, obtém-se um programa estático, com um conjunto constante de funcionalidades, mas cujo comportamento pode ser selecionado através de parâmetros, proporcionando assim uma grande gama de combinações.

Aumentando um pouco mais a complexidade do programa, pode-se acrescentar à parametrização a possibilidade de mascaramento de funcionalidades predefinidas.

Escolhendo-se um valor conveniente para a máscara, bloqueia-se ou habilita-se o acesso do usuário a cada uma das funções de um conjunto fixo que o programa potencialmente oferece, controlando assim o acesso do programa às diversas funcionalidades disponíveis.

Isso permite que um mesmo hardware e um mesmo programa básico possam ser utilizados de formas diferentes pelos diversos usuários, conforme a visibilidade que lhes é dada das funcionalidades potenciais do equipamento.

Assim, apesar de todo cliente possuir o software completo, ele poderá utilizar apenas os módulos realmente adquiridos, e aos demais, embora existam, não lhe será dada permissão de uso.

Tem-se aí, portanto, a simulação de uma funcionalidade variável, através do uso conjunto dos recursos da parametrização e da seleção, apresentados nos casos anteriores.

Quando se compra, por exemplo, um eletrodoméstico ou um equipamento complexo, quase sempre se recebe um hardware contendo um programa completo, capaz de executar todas as funções que foram projetadas para ele, mas ao usuário é dado acesso apenas às pertencentes ao subconjunto escolhido na ocasião da compra.

Assim, através da máscara, o fornecedor impede, segundo critérios preestabelecidos, o acesso a determinadas funções e o libera a outras, conforme a configuração a que cada particular cliente tem direito.

Sofisticando um pouco mais a estrutura dos programas, através do emprego do conceito de máquina virtual, com código interpretado, torna-se possível efetuar a simulação de (re)programações dinâmicas de funcionalidade, em programas mais complexos.

Imagine-se um programa como o mencionado anteriormente, mas desta vez equipado com um interpretador (máquina virtual) que é capaz de simular um certo conjunto de operações.

Seu usuário poderá escolher, dentre tais operações, aquelas que o ajudem a configurar o sistema, de forma que fique adequado aos seus propósitos. Esse conjunto de operações pode ser utilizado ativando-se

cada uma delas por meio de um script, organizado na forma de um programa.

Através desse script, o próprio programa se torna assim capaz de especificar quais operações devem ser executadas, bem como a ordem em que elas devem ser efetuadas. É possível, dessa maneira, memorizar esse script em uma área de dados, e na época da execução, o interpretador, operando sobre eles, faz com que tais dados se comportem como se fossem as instruções de um programa.

Resulta desse procedimento a execução interpretada do programa correspondente a esse script, a qual é viabilizada explorando-se as características associadas ao conceito de máquina virtual. Com isso, confere-se um comportamento programável ao software assim constituído, e isto era exatamente o importante passo que faltava ser dado para se dotar esse software da valiosa característica da adaptatividade.

Tornava-se conveniente incorporar um mecanismo que permitisse ao programa alterar o seu comportamento, e que tal modificação pudesse ser feita justamente através de transformações convenientes no script que determina a sua programação. Dispondo-se de tal recurso, o operador ganha a possibilidade de, agindo externamente, modificar a programação do sistema, determinando dessa maneira o seu comportamento imediatamente antes da execução do script.

Até aqui não se trata ainda de adaptatividade verdadeira, mas de uma técnica muito similar, e que proporciona muita versatilidade a um software, cujo comportamento prévio já podia ser parametrizado, simulado e selecionado, e que passa agora a ser também programado.

A seguir, nessa sucessão de complexidades crescentes dos programas, rumo à adaptatividade, o passo natural consiste em implantar nos programas recursos adicionais que proporcionem condições para que a funcionalidade desses programas passe a ser dinamicamente variável.

Isso significa que, durante a sua execução, os programas devem dispor de meios para criarem automaticamente novas funcionalidades através do emprego, direto ou indireto, da técnica da auto-modificação de código.

O caso anteriormente apresentado refere-se a um programa com funcionalidade estaticamente variável: embora seja possível modificar o script, para que este seja em seguida executado, tal script, uma vez alterado pelo usuário, permanecerá invariável durante toda a execução restante do programa.

Se, adicionalmente, for concedida ao próprio programa a possibilidade de acessar e modificar essa área de script, torna-se viável a obtenção de programas capazes de simular um comportamento dinâmico, apesar de, na realidade, seu código não ser obrigatoriamente dinâmico ou auto-modificável.

O esquema aqui apresentado opera da seguinte maneira: tem-se uma máquina virtual que interpreta instruções, e um script, codificado na forma de uma seqüência de dados, os quais representam operações de um programa a ser executado.

Os códigos assim formados vão sendo sucessivamente lidos e interpretados, e as operações correspondentes vão sendo executadas pela máquina virtual, e portanto a atividade que o script codifica acaba sendo efetuada passo a passo, dessa maneira.

É possível ainda disponibilizar no programa, para ser utilizada pelo seu próprio usuário final, uma funcionalidade que propicia uma atividade de reprogramação, efetuada por intervenção do próprio usuário final, o qual se torna assim capaz de alterar esse programa.

O passo decisivo para a adaptatividade pode ser dado em seguida, ao se proporcionar ao próprio programa a possibilidade de acionar essa funcionalidade de reprogramação.

Procedendo desta maneira, a alteração dinâmica de código daí decorrente deixa de ser resultado de ação humana, e portanto poderá ser conseqüência de uma atividade pré-programada ou então pode decorrer de decisões automáticas tomadas por procedimentos auxiliares de inteligência artificial incorporados ao programa.

Logo, através da ação do próprio programa, possivelmente com o auxilio de um procedimento de auto-modificação, auxiliado por um outro de tomada de decisão, ou de mais alguma lógica especial, acaba-se fechando finalmente esse ciclo, permitindo que o programa seja modificado por iniciativa e por ação do próprio programa, sem auxilio externo.

Nesse cenário, tendo-se um script, que o próprio programa possa modificar, script esse que determine integralmente o comportamento do programa, consegue-se finalmente obter, pela primeira vez, softwares que exibem um verdadeiro comportamento dinamicamente variável.

Convém notar, porém, que apesar de o código propriamente dito que o implementa continuar não sendo automodificável, o resultado da prática acima descrita é um comportamento autenticamente automodificável, embora ainda simulado.

É possível, assim, ter um código que seja alternativamente nativo ou virtual, e com isso se obtém um programa dinâmica ou estaticamente variável, ou seja, um software que, em tempo de execução, permita alterar ou não seu próprio código, respectivamente

O estudo apresentado até aqui sobre softwares com funcionalidade dinâmica levou-nos a percorrer uma seqüência gradativa de arquiteturas de programas, que cobre desde programas muito simples, de comportamento totalmente estático, até programas sofisticados, com comportamento totalmente dinâmico.

Toda a pesquisa que vem sendo realizada, tendo como alvo os programas adaptativos, orienta-se pela meta de desenvolver programas cujo comportamento possa ser calibrado em função da necessidade, ganhando-se com isso a possibilidade de ter uma funcionalidade que seja ao mesmo tempo dinâmica e também programável.

EXEMPLOS ILUSTRATIVOS – Por meio de programas que apresentam comportamento variável, podem ser elaboradas inúmeras aplicações de grande utilidade prática.

A modelagem rigorosa de tais aplicações pode empregar formulações matemáticas clássicas, eventualmente transformadas em versões adaptativas, mediante a adição de elementos que tornem o comportamento por elas descrito alterável durante sua operação.

Um exemplo, do dia-a-dia, mostra a situação em que pessoas e veículos modificam seu comportamento em função de alterações no ambiente em que estão mergulhados.

Só para citar uma dessas situações, a simples mudança periódica do estado de um semáforo de trânsito afeta dinamicamente o comportamento dos veículos e dos pedestres que trafegam na via: aqueles bloqueados, para os quais a mudança foi de vermelho para verde passam a poder trafegar livremente, o inverso ocorrendo para os demais, para os quais a mudança foi para vermelho, e cujo tráfego, antes livre, passa a estar bloqueado.

Programas reativos constituem outro exemplo, quando as ações por eles tomadas em resposta a um dado estímulo puderem variar em função dos seus antecedentes e da situação em que eles se encontram no instante em que o estímulo for recebido.

De forma ampla, programas em geral também costumam exibir uma característica similar: eles recebem dados como estímulos de entrada a serem tratados, e em função de sua situação interna, produto de sua história, podendo reagir ao estímulo de acordo com essa situação.

Outro exemplo, significativo e intuitivo, é o de uma simulação típica, como a de um jogo eletrônico, a de um brinquedo eletronicamente controlado de um parque de diversões, a de uma cabine de comando, ou de tantas outras atividades complexas.

Dado algum jogo, cujo comportamento original esteja definido na forma de um conjunto de regras, se à aplicação de uma ou mais dessas regras puder ser associada alguma alteração do conjunto das regras que o definem, tem-se aí criada uma realização concreta de um comportamento dinamicamente variável para o software resultante.

No novo simulador assim obtido, o conceito da adaptatividade pode ser identificado se for constatada a singular característica, assim

adquirida, de poder alterar espontaneamente seu próprio comportamento.

IX. Dispositivos adaptativos – formulação estendida

Discorre-se a seguir acerca de uma extensão à formulação clássica dos dispositivos adaptativos gerais [1], [2], [3], que com ela passam a levar em conta diversos avanços conceituais e estruturais recém-desenvolvidos nessa área.

Inicia-se descrevendo a componente não-adaptativa, presente em todos esses dispositivos na forma de um dispositivo subjacente guiado por regras, o qual serve como suporte para a construção da versão adaptativa.

Em seguida, apresenta-se uma proposta, cuja finalidade é a de dotar de adaptatividade esse dispositivo subjacente, acoplando-se-lhe uma camada adaptativa, a qual difere, em relação à formulação tradicional, essencialmente por apresentar-se estruturada de forma hierárquica, e em vários níveis.

A. Dispositivos Não-adaptativos Guiados por Regras

Os dispositivos abstratos guiados por regras têm seu comportamento totalmente descrito pelo conjunto, finito e fixo, de regras condicionais de movimentação, da forma clássica "**se** ... **então**".

Essas regras representam leis de mudança de configuração para o dispositivo, as quais determinam seu funcionamento. Na formulação aqui proposta, foi preservado seu formato tradicional.

Em operação, o dispositivo percorre sucessivas configurações, enquanto permanecer recebendo estímulos de entrada, exatamente conforme foi anteriormente descrito.

B. Consideração sobre Determinismo

Determinados dispositivos guiados por regras podem exibir uma atraente propriedade, a de apresentarem uma única possibilidade de movimento a partir de qualquer configuração em que se encontrem, ao longo de toda a sua operação.

A essa propriedade dá-se o nome de determinismo, e qualquer dispositivo que a apresente é dito determinístico.

Conseqüentemente, dispositivos determinísticos apresentam um eficiente comportamento dinâmico, pois fluem livremente, de configuração em configuração, até que esgotem os estímulos recebidos, quando encerram então sua operação.

Outros tipos de dispositivos, por seu lado, permitem mais de um movimento válido a partir de alguma de suas configurações, e nesse caso, denominam-se dispositivos não-determinísticos.

A realização prática mais natural de um dispositivo não-determinístico é obviamente aquela em que os possíveis movimentos sejam processados em paralelo, independentemente uns dos outros.

Em computadores nos quais não se disponha de tantos processadores quantos seriam exigidos por esse tipo de implementação, existe a possibilidade de seqüencializar as alternativas, de forma tal que seja processada apenas uma por vez, em cada processador disponível.

No limite, recai-se no tradicional processamento seqüencial, com um só processador, e nessa realidade a busca da seqüência mais adequada em cada caso recai em uma busca exaustiva, por tentativa e erro.

Ao contrário dos dispositivos determinísticos, os não-determinísticos exigem, a cada passo de operação, a execução de uma tomada de decisão, para que seja selecionado, dentre todas as possibilidades válidas de movimentação, um particular movimento, aquele que se mostrar mais adequado, a partir da particular configuração corrente.

Tais escolhas nem sempre são triviais, e sua aplicação sistemática durante a operação dos dispositivos não-determinísticos costuma prejudicar, em geral sensivelmente, a eficiência da operação do mesmo, constituindo isso uma das razões mais sérias pelas quais na prática se recomende que, na medida do possível, seja evitado o uso de não-determinismos.

C. Formulação dos Dispositivos Guiados por Regras

Nesta seção é apresentada a formulação adotada para os dispositivos (não-adaptativos) guiados por regras.

Um dispositivo não-adaptativo se caracteriza por não permitir qualquer tipo de alteração dinâmica em suas componentes, estruturais ou funcionais.

DISPOSITIVO NÃO-ADAPTATIVO – FORMULAÇÃO

Um dispositivo D, não-adaptativo, guiado por regras, pode ser descrito como uma quíntupla, conforme a formulação seguinte:

$$D = (C, R, S, c_0, A)$$

onde:

- C é o conjunto de todas as possíveis configurações de D;
- S é o conjunto de todos os estímulos válidos de entrada;
- $c_0 \in C$ é a única configuração inicial de D;
- $A \subseteq C$ é o conjunto de todas as configurações de aceitação de D;
- R é uma relação de mudança de configuração para D:

$$R \subseteq C \times (S \cup \{ \varepsilon \}) \times C,$$

cujos elementos constituem as regras que definem o dispositivo D:

$$r = (c_j, s, c'_j) \in R, \text{ com } c_j, c'_j \in C; s \in S,$$

as quais podem ser denotadas na forma

$$(c_j, s) \rightarrow c'_j,$$

realçando ser s o estímulo que, aplicado ao dispositivo D na configuração c_j, leva-o à sua nova configuração c'_j.

$$c_j \Rightarrow_s c'_j$$

A linguagem definida por D é representada por L(D):

$$L(D) = \{w \in S^* \mid c_j \Rightarrow_w^* c, c \in A\},$$

onde x* denota o fecho de Kleene de um conjunto ou operador x.

Dessa maneira, as regras, denotadas como triplas (c_j, s, c'_j), podem ser facilmente transcritas como cláusulas condicionais, da forma:

se a configuração corrente for c_j e o estímulo recebido for s,
então o dispositivo D poderá alterar sua configuração para c'_j.

Ocorrerá um movimento no dispositivo toda vez que uma dessas regras for aplicada, levando o dispositivo a migrar de sua configuração corrente para alguma outra, de acordo com aquilo que estiver determinado na regra aplicada.

Como a operação do dispositivo D depende total e exclusivamente desse conjunto de regras, ou, conseqüentemente, do conjunto de cláusulas condicionais equivalente, então pode-se dizer que o dispositivo D tem a sua operação determinada (ou guiada) pelo conjunto de regras que o define, ou, simplesmente, que se trata de um dispositivo guiado por regras.

D. Operação do Dispositivo Não-Adaptativo

O algoritmo seguinte descreve o funcionamento de um dispositivo (eventualmente não-determinístico) não-adaptativo.

Com essa descrição, estabelece-se um ponto de partida para a formulação, em seções subseqüentes, da operação de dispositivos mais complexos, com capacidade de auto-modificação.

DISPOSITIVO NÃO-ADAPTATIVO – OPERAÇÃO

1. Atribuir valores iniciais aos elementos:
   - $t = 0$ (o passo de execução corrente inicial é o passo 0)
   - $c_t = c_0$ (configuração inicial, fixa).

- $w_t = w_0 = w = s_1\ s_2\ s_3\ \ldots$ (seqüência inicial, completa, de estímulos de entrada).

2. Se $w_t = \varepsilon$ (seqüência de estímulos esgotada), desviar para o passo 4, caso contrário, extrair de w_t o próximo estímulo de entrada $s = s_{t+1}$, e então, incrementar t.
3. Obter, a partir de R, o sub-conjunto CR de regras compatíveis com o estímulo s e a configuração corrente c_t.
   - Se CR = ∅, rejeitar w.
   - Se CR={(c_t, s, c')}, mover D para a nova configuração $c_{t+1} = c'$, e desviar para o passo 2.
   - Se CR = {$r_k = (c_t, s, c'_k) \mid c'_k \in C, k = 1, \ldots, n, n>1$}, aplicam-se, em paralelo, as regras r_k, levando D simultaneamente às configurações $c'_1, c'_2, \ldots, c'_n$. Para cada um desses n casos, prosseguir no passo 2.
4. Caso $w_t = \varepsilon$ e D esteja nessa ocasião em alguma configuração $c_{t+1} \in F$, então D aceita w, caso contrário, D rejeita w. Em qualquer caso, encerra-se aqui a execução deste algoritmo.

A linguagem aceita pelo dispositivo será, então, o conjunto de todas aquelas seqüências de estímulos que, a partir de uma configuração inicial fixa, possam levar o dispositivo a alguma configuração final de aceitação, numa sucessão de movimentos determinados pela seqüência de regras aplicadas.

E. Adaptatividade Básica

Um dispositivo AD com adaptatividade básica se reduz a um dispositivo não-adaptativo subjacente, acoplado a um mecanismo capaz de alterar o conjunto de regras que define seu comportamento.

O mecanismo que converte um dispositivo não-adaptativo D em um dispositivo AD com adaptatividade básica incorpora um conjunto AA de funções especiais, denominadas funções adaptativas, acionáveis quando da aplicação das regras que definem o dispositivo adaptativo AD.

As funções adaptativas do conjunto AA proporcionam ao dispositivo AD uma forma de alterar o seu próprio conjunto das regras, pois os valores associados aos elementos variáveis que definem dispositivos adaptativos podem ser alterados a cada passo da sua operação.

Assim, resultam dispositivos adaptativos cujos comportamentos são descritos como conjuntos de regras, tendo à disposição uma variedade de funções adaptativas (incluindo a função adaptativa nula), as quais, convenientemente acopladas às regras do dispositivo subjacente, são executadas sempre que for necessário alterar o conjunto de regras.

Dispositivos não-adaptativos podem ser convertidos trivialmente em seus dispositivos adaptativos equivalentes, bastando usá-los como dispositivos subjacentes, e associando a função adaptativa nula a todas as suas regras.

Dispositivos assim obtidos são adaptativos no nome e na formulação, e, embora de fato não se automodifiquem, guardam matematicamente total compatibilidade com os dispositivos estritamente adaptativos, permitindo assim unificá-los no seu tratamento formal.

A introdução do conceito de funções adaptativas torna potencialmente mutável o comportamento de um dispositivo adaptativo, tornando conveniente o uso de um conjunto variável de regras RA, cujo valor RA_t depende do passo t de operação do dispositivo AD, exprimindo assim a dinâmica de seu comportamento. RA_t reflete, pois, o comportamento instantâneo do dispositivo adaptativo, ao seu t-ésimo passo de operação, e representa o conjunto de regras que nessa ocasião definem o comportamento do dispositivo.

Para formular as regras adaptativas do dispositivo AD, cada regra em RA_t associa a uma regra não-adaptativa do dispositivo subjacente duas funções adaptativas, a serem executadas uma antes, e outra, depois da mudança de configuração.

O emprego de uma regra $r \in R$, do dispositivo D subjacente, pela camada adaptativa AM(t) equivale à utilização de regras adaptativas da forma (β, r, α), cuja estrutura explicita o relacionamento entre o dispositivo subjacente e a camada adaptativa básica, aqui discutida.

Uma vez escolhida uma dessas regras adaptativas para ser aplicada, executa-se inicialmente sua função adaptativa anterior β, aplica-se a regra r, e por fim, executa-se sua função adaptativa posterior α.

Existe uma conjuntura particular que merece atenção, na operação dos dispositivos adaptativos: é a que ocorre quando, ao ser utilizada uma regra adaptativa, executando-se a função adaptativa anterior β, esta aplica à própria regra adaptativa uma operação de remoção, que a elimina do conjunto R(t) de regras.

Nesse caso, perde o sentido completar a aplicação da regra, pois esta já deixou de existir, e assim, o mais adequado em tal situação será descontinuar a aplicação daquela regra, reiniciando-se para o dispositivo o ciclo de aplicação de regras, agora escolhidas do conjunto R(t+1), resultante da remoção da regra em questão.

É importante alertar que a incorporação da adaptatividade a um dispositivo, por viabilizar sua automodificação, deve induzir seu usuário a tomar cuidados que garantam ao dispositivo manter-se permanentemente no controle, a despeito das alterações provocadas em seu próprio comportamento.

Cuidados similares se aplicam ao próprio software que implementa o dispositivo adaptativo, pois é real o perigo de perda do controle sobre o comportamento do dispositivo sempre que este, em operação, tiver a liberdade de provocar automodificações irrestritas.

O mecanismo adaptativo deve, portanto, ser projetado de forma tal que, apenas nas ocasiões mais apropriadas, seja promovida a execução de funções adaptativas que não comprometam a integridade operacional do dispositivo.

Do conhecimento do autor, não foi publicado nenhum método, para o desenvolvimento de camadas adaptativas, que garanta a perene preservação dessa integridade na operação de um dispositivo adaptativo, e isso constitui, sem dúvida, um interessantíssimo tema de pesquisa.

F. O Mecanismo Adaptativo Básico

A base da operação dos dispositivos dotados de adaptatividade básica torna-se assim bastante simples. Esse mecanismo conceitual rege o funcionamento dos dispositivos com adaptatividade básica, e se aplica a qualquer dispositivo, bastando para isso instanciar, no particular formato do dispositivo desejado, as regras mencionadas na formulação genérica a seguir:

MECANISMO ADAPTATIVO BÁSICO

Posicionar inicialmente o dispositivo AD em sua configuração inicial $c_0 \in C$, e extrair, da seqüência de entrada $w \in S^*$, o primeiro estímulo s a ser tratado.

1. Extrair, a partir do conjunto corrente de regras RA_t do dispositivo adaptativo, o sub-conjunto CR das regras compatíveis, para o estímulo s na configuração corrente c_t.
   São consideradas regras compatíveis todas as regras da forma $ra = (\beta b, r, \alpha a)$ cuja regra subjacente não-adaptativa $r = (c_t, s, c')$ referencie ao mesmo tempo a configuração corrente c_t e o estímulo corrente s.
   Para aplicar uma regra compatível $ra = (\beta b, (c_t, s, c'), \alpha a) \in CR$:
   - executar primeiro a chamada (*call*) βb da função adaptativa anterior (b). Caso isso faça com que a própria regra ra seja removida do conjunto RA_t, a regra correntemente em aplicação deixará de existir, perdendo portanto o sentido sua execução, devendo-se então voltar ao passo 1 para a escolha de outra regra compatível. Caso contrário, prosseguir.
   - aplicar a correspondente regra não-adaptativa subjacente (c_t, s, c'), o que leva o dispositivo AD à configuração c'.
   - finalizar executando a chamada (call) αa da função adaptativa posterior (a).

2. Aplicar simultaneamente todas as regras ra_j desse sub-conjunto CR (se houver mais de uma):
   - Se CR = ∅, encerrar a tentativa de reconhecimento em curso. Encerrar o algoritmo, rejeitando w se não houver outras tentativas paralelas em andamento.
   - Se CR = { ra = (βb, (c_t, s, c'), αa) }, conjunto unitário, aplicar, deterministicamente, a (única) regra ra, movendo AD para a nova configuração c_{t+1} = c'.
   - Se CR = { ra_k = (βb_k, (c_t, s, c'_k), αa_k) | $c'_k \in C$, sendo βb_k, αa_k chamadas (*calls*) de funções adaptativas pertencentes ao conjunto AA; k = 1, … , n, n>1}, aplicam-se, não-deterministicamente, todas as regras $ra_k \in$ CR.
3. Não havendo novos estímulos, encerrar o algoritmo, aceitando w caso tenha sido atingida uma configuração de aceitação. Caso contrário, extrair da seqüência de entrada um novo estímulo s, incrementar t e voltar ao passo 1 para que seja iniciado mais um passo de operação do dispositivo AD.

É importante notar que um mecanismo dessa natureza pode ser considerado como sendo o alicerce principal da adaptatividade, pois é ele que provê a dinâmica comportamental dos dispositivos, podendo ser identificado com facilidade em todos os fenômenos nos quais se manifestem atividades de automodificação.

G. Formulação de Dispositivos com Adaptatividade Básica

Dispositivos com adaptatividade básica são dispositivos adaptativos cuja capacidade de modificação dinâmica fica restrita à alteração do conjunto de regras adaptativas, que determina seu comportamento. Nesta seção é apresentada a formulação adotada para dispositivos guiados por regras, portadores de adaptatividade básica, os quais se caracterizam por permitirem alterações dinâmicas apenas nos conjuntos de regras que definem seus respectivos comportamentos.

DISPOSITIVO ADAPTATIVO BÁSICO – FORMULAÇÃO

Um dispositivo AD, guiado por regras, dotado do recurso de adaptatividade básica, pode ser descrito como uma sêxtupla, conforme a formulação seguinte:

AD = (C, RA, S, AA, c_0, RA_0, A), onde:

- C é o conjunto de todas as possíveis configurações de AD;
- $c_0 \in C$ é a configuração inicial, única, de AD;
- $A \subseteq C$ é o conjunto das configurações de aceitação de AD;
- S é o conjunto dos estímulos válidos de entrada de AD;
- AA é o conjunto, fixo, de funções adaptativas de AD;
- RA_0 é um conjunto, fixo, de regras adaptativas iniciais de AD;
- RA é o conjunto de regras adaptativas de AD;

Merecem destaque os seguintes detalhes, referentes aos elementos dessa formulação:

- Os elementos do conjunto AA são as funções adaptativas φ, que apresentam os seguintes componentes (no caso mais trivial, φ qualquer pode ser a função adaptativa nula λ):
  - Cabeçalho da função adaptativa φ, contendo:
    - Nome φ, pelo qual a função adaptativa é referenciada.
    - Ênupla ordenada (ρ_1, ρ_2, ..., ρ_p), de p≥0 parâmetros formais, similares às de funções em linguagens de programação.
    - Conjunto {υ_1, υ_2, ..., υ_v} de v ≥ 0 variáveis, cujo valor é preenchido uma só vez por ações adaptativas de consulta.
    - Conjunto {χ_1, χ_2, ..., χ_g} de g ≥ 0 geradores, cujo valor é preenchido com novos valores a cada chamada da função adaptativa.
  - Corpo ($\beta\mu$, Δ, $\alpha\nu$) da função adaptativa φ, compreendendo:
    - Chamada (*call*) $\beta\mu$ de uma função adaptativa anterior (μ), cuja execução esteja prevista para antes da aplicação do conjunto de ações adaptativas elementares que formam o núcleo da função adaptativa φ.
    - O núcleo, a parte essencial da função adaptativa φ, que consiste de um conjunto invariável Δ = {δ_1, δ_2, ..., δ_e}, contendo e ≥ 0 ações adaptativas elementares δ_i, 0≤i<e. Representa a coleção das alterações a serem aplicadas ao dispositivo adaptativo AD toda vez que a função adaptativa φ for executada.
      Cada ação adaptativa elementar δ_i, 0≤i<e, é denotada no formato seguinte:
      $$\delta_i = \otimes [\, r \,].$$
      O símbolo ⊗ representa um operador, aplicado à regra r e ao conjunto RA_t de regras que definem o comportamento do dispositivo AD (conjunto RA, com o conteúdo que nele se encontra no passo t de operação de AD), podendo ser um dos seguintes:
      - \+ (inserção): inclui uma nova regra r em RA_t.
      - – (remoção): elimina a regra r do conjunto RA_t.
      - ? (consulta): pesquisa o conjunto RA_t usando como padrão de busca a regra r indicada, e preenchendo variáveis e parâmetros com os resultados da busca.
    - Chamada (*call*) $\alpha\nu$ de uma função adaptativa posterior (ν), cuja execução esteja prevista para após a aplicação do conjunto das ações adaptativas elementares definidas no conjunto Δ.
  - Para completar esta descrição da formulação das funções adaptativas, resta definir o formato para as chamadas (*calls*) de funções adaptativas φ, (denotadas na presente publicação como $\beta\varphi$ ou $\alpha\varphi$, conforme se trate de ação adaptativa anterior ou posterior, respectivamente) com p ≥ 0 parâmetros formais ρ_1, ρ_2, ..., ρ_p .
    Sendo θ_1, θ_2, ..., θ_p os argumentos correspondentes, onde os θ_i representam valores posicionalmente correspondentes aos parâmetros formais ρ_i (i = 1, ... , p) aos quais se referem, uma chamada (*call*), por exemplo, $\alpha\varphi$, assume o formato seguinte:
    $$\alpha\varphi = \varphi\,(\theta_1, \theta_2, ..., \theta_p).$$
    Obviamente, para p = 0, tem-se uma função sem parâmetros, e neste caso o formato da sua chamada (*call*) se reduz a:
    $$\alpha\varphi = \varphi\,(\,).$$
- O conjunto RA é variável por ação das funções adaptativas (elementos de AA), assumindo sucessivamente os valores RA_0, RA_1, ..., RA_t, (t ≥ 0) a cada passo do processamento de AD.
  Os valores RA_i assumidos por RA (i ≥ 0) representam, na ocasião do passo i do dispositivo AD, a relação de mudança de configuração que define o comportamento do dispositivo nessa oportunidade:
  $$RA_i \subseteq AA \times (C \times (S \cup \{\varepsilon\}) \times C) \times AA,$$
  O produto cartesiano (C × (S∪{ε}) × C) representa o universo das possíveis regras r_j do dispositivo não-adaptativo subjacente.
  Os elementos ra_j são as regras adaptativas cujo conjunto RA_i define o comportamento do dispositivo AD ao passo i de operação, e têm a forma:
  $$ra_j = (\beta b_j, r_j, \alpha a_j),$$

onde βb_j, $\alpha a_j \in AA$ são, respectivamente, as chamadas (*calls*) anteriores e posteriores, possivelmente paramétricas, de funções adaptativas b_j, $a_j \in AA$.

A regra não-adaptativa subjacente r_j tem a forma

$$r_j = (c_j, s, c'_j) \in R, \text{ com } c_j, c'_j \in C; s \in S,$$

podendo também ser denotada como

$$(c_j, s) \rightarrow c'_j,$$

realçando ser s o estímulo que, aplicado ao dispositivo na configuração c_j, leva-o à sua nova configuração c'_j.

$$c_j \Rightarrow_s c'_j$$

A linguagem definida por D é representada por L(D):

$$L(D) = \{w \in S^* \mid c_i \Rightarrow_w^* c, c \in A\},$$

onde o x* denota o fecho de Kleene de um conjunto ou operador x.

Sendo AA o conjunto (finito, fixo) de funções adaptativas do dispositivo, e R_t o conjunto das regras (descritas em sua formulação não-adaptativa) que definem seu dispositivo subjacente na ocasião do seu t-ésimo passo de execução ($t \geq 0$), o seu mecanismo adaptativo básico AM_t nesse passo de operação pode ser interpretado como:

$$AM_t \subseteq AA \times R_t \times AA,$$

determinando, naquela ocasião, o conjunto de regras (adaptativas) de operação do dispositivo com adaptatividade básica.

As regras adaptativas obtidas pela utilização desse mecanismo adaptativo básico assumem a forma ar = (βb, r, αa), conforme formulado anteriormente, com $r \in R_t$, sendo que βb e αa, que denotam chamadas (*calls*) das funções adaptativas b, $a \in AA$, representam as chamadas (*calls*) das funções adaptativas previstas para serem executadas respectivamente antes e depois da aplicação da regra $r \in R_t$.

O princípio de operação desse mecanismo adaptativo básico é o que foi apresentado: partindo-se da configuração inicial do dispositivo, aplicam-se ciclicamente as regras adaptativas, até que não haja movimento possível ou até que, esgotados os estímulos fornecidos ao dispositivo, seja atingida alguma particular configuração de término de operação (eventualmente indicativa de sucesso ou fracasso do processamento da seqüência de estímulos do dispositivo).

H. Operação de um Dispositivo com Adaptatividade Básica

Um dispositivo dotado de adaptatividade básica opera de acordo com o algoritmo esboçado a seguir. Sua lógica é relativamente simples, e espelha o funcionamento anteriormente discutido para os dispositivos guiados por regras, aos quais tenha sido acrescentada uma camada adaptativa básica:

DISPOSITIVO ADAPTATIVO BÁSICO – OPERAÇÃO

1. Atribuir valores iniciais aos elementos:
   - $t = 0$ (passo inicial 0).
   - $c_t = c_0$ (configuração inicial, fixa).
   - $w_t = w_0 = s_1 s_2 s_3 \ldots$ (cadeia de estímulos de entrada).
2. Se $w_t = \varepsilon$ (seqüência de estímulos esgotada), desviar para o passo 7, caso contrário, incrementar t e então extrair de w_t o próximo estímulo s_t.
3. Para cada uma das configurações correntes c_t que estejam sendo paralelamente processadas, obter de C o sub-conjunto CA de regras adaptativas a ela compatíveis.
   - Se $CA = \varnothing$, rejeitar w.
   - Se $CA = \{ (\beta b_p, (c_t, s, c'_p), \alpha a_p) \}$, aplicar deterministicamente a única regra nele contida conforme os passos 4, 5 e 6, levando AD a uma nova configuração c_{t+1}.
   - Se $CA = \{ ar_k = (\beta b_k, (c_t, s, c'_k), \alpha a_k) \mid nr_k = (c_t, s, c'_k)$ é uma regra do dispositivo não-adaptativo subjacente, $c'_k \in C, k = 1, \ldots, n, n>1\}$, aplicam-se em paralelo as regras ar_p desse conjunto, conforme os passos 4, 5 e 6, levando AD em paralelo às correspondentes configurações c_{t+1} do conjunto $\{c'_1, \ldots, c'_n\}$.
4. Se $\beta b_p = \lambda(\,)$ (a ação adaptativa nula) desviar para o passo 5; Se não, aplicar inicialmente βb_p. Caso a regra ar_p resulte removida pela aplicação de βb_p, desviar para o passo 3, descontinuando ar_p. Caso contrário AD terá atingido uma configuração estável, e neste caso, desviar para o passo 2.
5. Aplicar a regra nr_p à configuração intermediária corrente, levando AD a uma nova configuração estável $c_{t+1} = c'_p$.
6. Se $\alpha a_p = \lambda(\,)$ (a ação adaptativa nula) desviar para o passo 7; Se não, aplicar finalmente a ação adaptativa αa_p, e então, desviar para o passo 2.
7. Caso algum $c_{t+1} \in F$ tenha sido atingido, então AD aceita w, caso contrário, AD rejeita w; em qualquer caso, encerra-se aqui a execução deste algoritmo.

I. Adaptatividade Multinível Hierárquica

Descreve-se nesta seção o conceito de adaptatividade hierárquica e multinível, pelo qual se acrescenta à adaptatividade básica a possibilidade de alteração controlada das funções adaptativas e também sua hierarquização [4].

Segundo essa nova organização, os mecanismos responsáveis pela automodificação dos dispositivos adaptativos utilizam-se, para isso, de várias camadas hierarquicamente estruturadas, de forma que cada camada adaptativa tem acesso direto de modificação apenas à sua camada vizinha, de hierarquia imediatamente inferior.

Dessa maneira, confinam-se as alterações possíveis, evitando-se modificações indiscriminadas nos mecanismos adaptativos, restringindo o acesso apenas ao nível vizinho inferior, o que reduz significativamente a complexidade do dispositivo como um todo.

A introdução da hierarquia não limita de forma alguma o dispositivo resultante, pois continuam viáveis quaisquer acessos necessários a camadas não vizinhas, desde que se guarde o protocolo de encaminhar convenientemente a informação necessária através das camadas intermediárias.

Assim sendo, um dispositivo com adaptatividade multinível hierárquica pode ser formulado indutivamente, e é constituído essencialmente dos mesmos elementos que formam os dispositivos com adaptatividade básica:

- Seu dispositivo não-adaptativo subjacente, o qual constitui a base da sua formulação indutiva, e nessa hierarquia, representa uma camada de adaptatividade de nível zero.
- O mecanismo adaptativo hierárquico multi-nível, que permite que sejam causadas alterações no conjunto de regras que define o dispositivo não-adaptativo subjacente, promovendo assim sua adaptatividade por meio de um comportamento dinâmico e hierárquico.
- A camada adaptativa, que se apresenta como uma hierarquia de níveis, na qual cada nível se sobrepõe ao nível imediatamente anterior, e assim, o primeiro deles, de primeiro nível, se sobrepõe diretamente ao dispositivo subjacente não-adaptativo, ou seja, de nível zero.
- As funções adaptativas dos dispositivos com adaptatividade hierárquica multinível, as quais, além do usual acesso à modificação do conjunto de regras adaptativas do próprio nível, podem também reprogramar as funções adaptativas do nível imediatamente inferior, quando existirem.
- A associação das chamadas de funções adaptativas às regras do mesmo nível, para execução sempre que a regra for aplicada, que é efetuada em cada nível, pelo mecanismo adaptativo.
- O nível de adaptatividade: um dispositivo adaptativo é dito de nível N quando suas regras adaptativas apresentam nível máximo N de adaptatividade.

Da mesma forma como foi feito no caso dos dispositivos com adaptatividade básica, os dispositivos subjacentes de nível zero são obtidos a partir dos dispositivos não-adaptativos por um simples ajuste notacional, o mesmo podendo ser feito para outros níveis, compatibilizando a apenas a notação, sem modificar o seu comportamento.

J. Mecanismo Adaptativo Hierárquico de Nível N

Nesta seção, descreve-se um mecanismo adaptativo hierárquico HM de nível N. Trata-se de uma variante do mecanismo adaptativo básico AM, anteriormente proposto, ao qual foram incorporados os elementos necessários à formulação de dispositivos com estrutura adaptativa hierárquica e multinível. Essencialmente, as alterações atingiram a notação, que agora explicita o nível da camada adaptativa a que se refere, e introduziram uma definição indutiva das camadas adaptativas, responsável pela formulação da estrutura hierárquica desejada.

MECANISMO ADAPTATIVO HIERÁRQUICO

- Um mecanismo adaptativo hierárquico HM, de nível $N \geq 0$, no seu passo $t \geq 0$ de operação, é composto de N+1 camadas adaptativas de nível k:
  $$HM(k, t) \mid 0 \leq k \leq N.$$
- Assim, pode-se definir (indutivamente) o mecanismo adaptativo HM caracterizando cada uma de suas camadas adaptativas HM(k, t) de nível k, $0 \leq k \leq N$, no passo t:
  - Para $k = 0$, sendo λ a função adaptativa nula (sem parâmetros), e R_t o conjunto de regras do dispositivo subjacente (não-adaptativo) em seu passo t de operação, define-se a camada de nível 0 do mecanismo adaptativo HM:
    $$HM(0, t) = \{\lambda\} \times R_t \times \{\lambda\}.$$
    Dessa forma, sendo $r \in R_t$ uma regra do dispositivo não-adaptativo HM(0,t) subjacente, pode-se denotar a regra r^0 equivalente, em sua formulação adaptativa de nível zero:
    $$r^0 = (\lambda( ),\ r,\ \lambda( )).$$
  - Para $0 < k \leq N$, sendo HA(k,t) o conjunto de todas as funções adaptativas de nível k, na forma em que se encontrarem no passo t de operação do dispositivo HD (para $0 < k \leq N$):
    $$HM(k, t) \subseteq HA(k, t) \times HM(k-1, t) \times HA(k, t).$$
    Sendo b^k, $a^k \in HA(k,t)$ e $r^{k-1} \in HM(k-1,t)$ uma regra adaptativa de nível k-1, HM(k,t) constrói uma regra r^k de nível k, associando-lhe ações adaptativas anteriores e posteriores βb^k e αa^k, respectivamente chamadas (*calls*) das funções adaptativas b^k e a^k, de nível k (para $0 < k \leq N$):
    $$r^k = (\beta b^k, r^{k-1}, \alpha a^k).$$
    Observe-se que, de acordo com a natureza das funções b^k e a^k (eventualmente vazias), é possível que as correspondentes chamadas βb^k e αa^k sejam paramétricas.
    Note-se ainda que, sendo HD um dispositivo de nível N, não podem existir funções adaptativas de nível $k > N$, logo, tampouco poderão existir as correspondentes camadas adaptativas HM(k, t):
    $$HM(k, t) = \varnothing \times HM(k-1, t) \times \varnothing = \varnothing \text{ , para } k > N.$$
  - As regras que definem um dispositivo adaptativo hierárquico multinível podem, portanto, ser assim denotadas:
    $$(\beta b_N, (..., (\beta b_1, (\lambda( ), (c, s, c'), \lambda( )), \alpha a_1), ...), \alpha a_N)$$
    onde:
    - (c, s, c') se refere a uma regra do dispositivo subjacente não-adaptativo de HD.
    - $r^0 = (\lambda( ), (c, s, c'), \lambda( ))$ é essa mesma regra, expressa na formulação adaptativa equivalente de nível zero.
    - $r^k = (\beta b_k, r^{k-1}, \alpha a_k)$ agrega, à regra r^{k-1} de nível k-1, uma camada adaptativa de nível k ($1 \leq k \leq N$).
- Cada camada HM(k,t) de nível k ($0 < k \leq N$), do mecanismo adaptativo HM, dispõe de um conjunto HA(k,t) de funções adaptativas do nível k, com $\lambda \in HA(k,t)$, com as quais é possível:
  - Alterar para HM(k,t+1) o conjunto HM(k,t) das regras adaptativas de seu próprio nível k ($0 < k \leq N$), através de ações de inclusão e de exclusão de regras.
  - Alterar o funcionamento de alguma função adaptativa $\varphi^{k-1}(t) \in HA(k-1,t)$, do nível imediatamente inferior k-1, resultando assim uma nova versão $\varphi^{k-1}(t+1)$ dessa função, e acarretando, como conseqüência, a obtenção de uma versão modificada desse mesmo conjunto de funções ($1 < k \leq N$):
    $$HA(k-1,t+1) = (HA(k-1,t) - \{\varphi^{k-1}(t)\}) \cup \{\varphi^{k-1}(t+1)\}.$$

  Observe-se que, nunca existindo funções adaptativas em níveis de adaptatividade $k > N$ (por definição), o conjunto HA(N,t+1) será o próprio HA(N,t), ou seja, HA(N,t) não varia, para todo $t \geq 0$.
  Note-se também que, embora a presente reformulação não contemple a criação dinâmica de novas funções adaptativas, limitando-se apenas à alteração do comportamento de funções adaptativas já existentes, não há motivo para que tal restrição seja preservada, podendo incorporar-se, se necessário em futuras reformulações dos dispositivos adaptativos.

- Está faltando ainda definir melhor as funções φ adaptativas de nível k, que compõem os conjuntos HA(k,t). Como essas funções podem ser alteradas dinamicamente pelo mecanismo adaptativo, será usada a notação φ^k_t para designar uma função adaptativa φ, tal como se encontra no passo t da operação de HD.
  - Uma função adaptativa qualquer, utilizada pelo dispositivo HD, apresenta os seguintes elementos:
    - Nome φ.
    - Nível k da camada adaptativa HM(k,t) a que pertence.
    - Passo t da operação do dispositivo HD, que a utiliza.
    - Denota-se φ^k_t, como é referenciada doravante neste texto.
    - Seqüência $(\rho_1, \rho_2, ..., \rho_p)$ de $p \geq 0$ parâmetros formais, similares aos utilizados pelos procedimentos e funções encontrados nas linguagens usuais de programação.
    - Conjunto $\{\upsilon_1, \upsilon_2, ..., \upsilon_v\}$ de $v \geq 0$ variáveis, componentes básicos da formulação, aos quais são associados valores, uma só vez, como resultado da execução de operações elementares de consulta ou de remoção especificadas na função adaptativa φ^k_t.
    - Conjunto $\{\chi_1, \chi_2, ..., \chi_g\}$ de $g \geq 0$ geradores, componentes básicos da formulação, aos quais são associados valores, uma só vez, toda vez que a função adaptativa φ^k_t é chamada.
    - Chamada (*call*) βb^k de uma função adaptativa anterior b^k, do mesmo nível k de adaptatividade, para ser executada antes do conjunto das ações adaptativas elementares que constituem a parte essencial da função adaptativa φ^k_t.
    - Conjunto $\Delta^k = \{\delta^k_1, \delta^k_2, ..., \delta^k_{ek}\}$ contendo $e^k \geq 0$ ações adaptativas elementares δ^k_i, $1 \leq i < e^k$, todas do mesmo nível k de adaptatividade. Indica a coleção das alterações específicas a serem aplicadas a cada execução da função adaptativa φ^k_t pela camada adaptativa HM(k,t) de nível k, às regras do dispositivo adaptativo hierárquico HD.
      Note-se que este conjunto Δ^k não é obrigatoriamente constante, podendo ser modificado durante a operação de HD pela eventual iniciativa de funções adaptativas do nível k+1 (para $1 \leq k < N$).
      Nas fórmulas a seguir, o símbolo $\otimes$ denota um dos operadores seguintes, aplicado ao conjunto apropriado –

conjunto CA de regras, definido em HM(k,t), ou então, conjunto Δ^{k-1} de ações adaptativas elementares, que caracteriza uma função adaptativa $\varphi^{k-1}{}_t$:

- \+ inserção: inclui um novo elemento no conjunto.
- – remoção: elimina do conjunto o elemento indicado.
- ? consulta: pesquisa o conjunto usando como padrão de busca o elemento indicado, e preenchendo variáveis e parâmetros com os resultados da busca.

Cada uma das ações adaptativas elementares, $\delta^k{}_i$, $0 \le i < e^k$, de nível k, pode apresentar-se em uma das duas formas seguintes:

- As que aplicam o operador $\otimes$ diretamente à regra r^k do conjunto CA de regras do mesmo nível k, do dispositivo HD (para $0 < k \le N$):
$$\delta^k{}_i = \otimes [\, r^k \,].$$
- As que operam sobre o comportamento das funções adaptativas de nível k-1 (para $1 < k \le N$):
$$\delta^k{}_i = \otimes \varphi^{k-1}{}_t [\, \delta^{k-1}{}_j \,],$$
onde $\delta^k{}_i \in \Delta^k$ e $\delta^{k-1}{}_j \in \Delta^{k-1}$, e se refere a $\varphi^{k-1}{}_t$.
Neste caso, uma função adaptativa de nível k, em execução, promove a aplicação da ação adaptativa elementar $\delta^k{}_i \in \Delta^k$, a qual, por sua vez, aplica o operador $\otimes$ à ação adaptativa elementar $\delta^{k-1}{}_j$ destacada entre colchetes nesta notação, e referente à definição da função adaptativa $\varphi^{k-1}{}_t$, de nível k-1.

- Chamada (*call*) $\alpha a^k{}_t$ de uma função adaptativa posterior $a^k{}_t$, do mesmo nível k de adaptatividade, para ser executada após o conjunto Δ^k das ações adaptativas elementares que compõem a parte essencial da função adaptativa $\varphi^k{}_t$.

- No caso mais trivial, uma função adaptativa qualquer $\varphi^k{}_t$, de nível $0 < k \le N$, pode eventualmente ser a função adaptativa sem parâmetros nula λ, cuja execução não exerce qualquer alteração sobre o dispositivo HD.
- Para completar a formulação, apresenta-se uma definição de formato para as chamadas (*calls*) $\alpha a^k{}_t$ das funções adaptativas $a^k{}_t$, com $p \ge 0$ parâmetros formais $\rho_1, \rho_2, ..., \rho_p$.
Sendo $\theta_1, \theta_2, ..., \theta_p$ os argumentos correspondentes, onde os θ_i denotam valores (que podem referir-se a quaisquer instâncias de elementos componentes da formulação do dispositivo HD em questão) posicionalmente associados aos respectivos parâmetros formais ρ_i (i = 1, ... , p), uma chamada (*call*) αa^k assume o formato seguinte:
$$\alpha a^k{}_t = a^k{}_t (\theta_1, \theta_2, ..., \theta_p).$$
Obviamente, quando p = 0, isso indica que $a^k{}_t$ é uma função adaptativa sem parâmetros.
Por exemplo, quando $a^k{}_t$ é a função adaptativa nula λ, o formato da sua chamada (*call*) $\alpha\lambda$ se reduz a:
$$\alpha\lambda = \lambda( ).$$

K. Dispositivo com Adaptatividade Hierárquica de Nível N

A exemplo do que foi visto para dispositivos com adaptatividade básica, é possível descrever um dispositivo HD, guiado por regras, dotado do recurso de adaptatividade hierárquica, como uma sêxtupla, conforme a formulação anteriormente utilizada:

(C, RA, S, AA, c_0, RA_0, A),

onde, embora todos os elementos tenham a mesma conotação conceitual anteriormente estudada, os elementos que envolvem as funções adaptativas devem ser reinterpretados:

- RA e RA_0 são constituídos de regras que apresentam agora formato hierárquico multinível.
- AA passa a conter funções adaptativas capazes de modificar o funcionamento de outras funções adaptativas.

Por questão de clareza, não será usada a sêxtupla acima nesta formulação, pois torna-se conveniente evidenciar a camada adaptativa do novo dispositivo, realçando o caráter estratificado do seu formalismo hierárquico.

DISPOSITIVO ADAPTATIVO HIERÁRQUICO – FORMULAÇÃO

Formula-se a seguir um dispositivo guiado por regras com adaptatividade hierárquica:

HD = (ND_0, HM, N), onde:

- $N \ge 0$ representa o nível de adaptatividade do dispositivo HD. Note-se que:
  - para N=0, esta formulação pode ser vista como uma denotação alternativa geral para dispositivos adaptativos, aplicável até mesmo quando HD não é adaptativo.
  - Para N=1, essa formulação oferece uma notação alternativa que se aplica ao caso particular em que HD se refere a um simples dispositivo com adaptatividade básica.
- ND_0 representa um dispositivo subjacente inicial (em formulação não-adaptativa) ao qual se está incorporando a adaptatividade:
ND_0 = (C, R_0, S, c_0, A), onde:
  - C é o conjunto de todas as possíveis configurações do dispositivo ND_0
  - R_0 representa o conjunto das regras (em formulação não-adaptativa) que definem, no passo de operação 0 de HD, o comportamento do dispositivo subjacente ND_0, cujas alterações são promovidas pelo mecanismo adaptativo HM.
  - S é o conjunto dos estímulos que geram as movimentações (mudanças de configuração) do dispositivo ND_0.
  - $c_0 \in C$ é a configuração inicial (única) do dispositivo ND_0.
  - $A \subseteq C$ é o conjunto de todas as configurações finais (de aceitação) do dispositivo ND_0.
- HM é o mecanismo adaptativo hierárquico multinível, conforme definido anteriormente, o qual, atrelando em camadas estratificadas, chamadas de funções adaptativas às regras do dispositivo subjacente ND_0, capacita HD a efetuar auto-modificações comportamentais por dois diferentes caminhos: pela modificação do conjunto de regras ou então pela alteração do comportamento de funções adaptativas existentes.

L. Operação de um Dispositivo com Adaptatividade Hierárquica de nível N

Um dispositivo dotado de adaptatividade hierárquica de nível N opera de acordo com o algoritmo esboçado a seguir:

DISPOSITIVO ADAPTATIVO HIERÁRQUICO – OPERAÇÃO

1. Posicionar em seus valores iniciais os elementos:
   - $t = 0$ (passo inicial 0).
   - $c_t = c_0$ (configuração inicial, fixa).
   - $w_t = w_0 = s_1 s_2 s_3 \ldots$ (cadeia de estímulos de entrada).
   - o conjunto inicial de regras adaptativas hierárquicas
   - os conjuntos de ações adaptativas elementares associadas às funções adaptativas, em cada nível

2. Se $w_t = \varepsilon$ (seqüência de estímulos esgotada), desviar para o passo 7, caso contrário, incrementar t e então extrair de w_t o próximo estímulo s_t.
3. Para cada uma das configurações correntes c_t que estejam sendo paralelamente processadas, determinar, a partir dos conjuntos HM(k, t) ($0 < k \leq N$), os sub-conjuntos CA correspondentes de regras adaptativas de nível k, a elas compatíveis. Em cada caso,
   - Se CA = ∅, rejeitar w.
   - Se CA = { $(\beta_p, (c_t, s, c'_p), \alpha_p)$ }, aplicar essa única regra deterministicamente de acordo com os passos 4, 5 e 6, levando HD a uma nova configuração c_{t+1}.
   - Se CA = { $ar_k = (\beta_k, (c_t, s, c'_k), \alpha_k) \mid nr_k = (c_t, s, c'_k)$ é uma regra do dispositivo não-adaptativo subjacente, $c'_k \in C$, k = 1, … , n, n>1}, aplicam-se em paralelo as regras ar_p desse conjunto, conforme os passos 4, 5 e 6, levando HD em paralelo às configurações c_{t+1} do conjunto { $c'_1, c'_2, \ldots, c'_n$ }.
4. Se $\beta_p = \lambda(\,)$ (a ação adaptativa nula) desviar para o passo 5; Se não, aplicar inicialmente β_p. Caso a regra ar_p resulte removida pela aplicação de β_p, desviar para o passo 3, descontinuando ar_p. Caso contrário HD terá atingido uma configuração estável, e neste caso, desviar para o passo 2.
5. Aplicar a regra nr_p à configuração intermediária corrente, levando HD a uma nova configuração estável $c_{t+1} = c'_p$.
6. Se $\alpha_p = \lambda(\,)$ (a ação adaptativa nula) desviar para o passo 7; Se não, aplicar finalmente a ação adaptativa α_p, e então, desviar para o passo 2.
7. Caso algum $c_{t+1} \in F$ tenha sido atingido, então HD aceita w, caso contrário, HD rejeita w; em qualquer caso, encerra-se aqui a execução deste algoritmo.

EXEMPLO

Apresenta-se a seguir um pequeno caso particular de autômato finito adaptativo de nível 3. Destina-se apenas a ilustrar o que foi apresentado, e não constitui aplicação prática, que seria muito mais complexa, tornando proibitivo apresentá-la neste texto. Neste exemplo, nenhuma das funções adaptativas utilizadas é paramétrica:

R_0 = { (λ(), (1, a, 2), λ()), (λ(), (2, b, 1), λ()), (λ(), (2, ε, 3), λ()) }
$A_0 = \varnothing$
$B_0 = \varnothing$

R_1 = { (λ(), (λ(), (1, a, 2), λ()), a_1()),
(b_1(), (λ(), (2, b, 1), λ()), λ()),
(λ(), (λ(), (2, ε, 3), λ()), λ()) }
A_1 = { a_1 }
B_1 = { b_1 }

ações adaptativas elementares da função adaptativa a_1:
{ - [(λ(), (λ(), (1, a, 2), λ()), a_1 ())],
+ [(λ(), (λ(), (2, a, 1), λ()), λ())],
+ [(λ(), (λ(), (1, b, 2), λ()), λ())] }

ações adaptativas elementares da função b_1:
{ - [(b_1(), (λ(), (2, b, 1), λ()), λ ())],
+ [(λ(), (λ(), (2, b, 1), λ()), λ())],
+ [(λ(), (λ(), (1, a, 2), λ()), λ())] }

R_2 = { (λ(), (λ(), (λ(), (1, a, 2), λ()), a_1()), a_2()),
(λ(), (b_1(), (λ(), (2, b, 1), λ()), λ()), λ()),
(λ(), (λ(), (λ(), (2, ε, 3), λ()), λ()), λ()) }
A_2 = { a_2 }
$B_2 = \varnothing$

ações adaptativas elementares da função adaptativa a_2:
{ - b_1 [+ [(λ(), (λ(), (2, b, 1), λ()), λ())]],
+ a_1[- [(λ(), (λ(), (1, b, 2), λ()), λ ())]],
- [(λ(), (b_1(), (λ(), (2, b, 1), λ()), λ()), λ())] }

R_3 = { (λ(), (λ(), (λ(), (λ(), (1, a, 2), λ()), a_1()), a_2()), λ()),
(b_3(), (λ(), (b_1(), (λ(), (2, b, 1), λ()), λ()), λ()), λ()),
(λ(), (λ(), (λ(), (λ(), (2, ε, 3), λ()), λ()), λ()), λ()) }
$A_3 = \varnothing$
B_3 = { b_3 }

ações adaptativas elementares da função adaptativa b_3:
{ - b_2 [+ a_1 [- [(λ(), (λ(), (1, b, 2), λ()), λ())]]],
- [(λ(), (λ(), (λ(), (λ(), (1, a, 2), λ()), a_1()), a_2 ()), λ())]
}

X. Observações Finais

As formulações apresentadas nesta publicação, para os dispositivos guiados por regras – não-adaptativos, com adaptatividade básica e com adaptatividade hierárquica – permitem exprimir e tratar de maneira uniforme os elementos dinâmicos das formulações referentes a dispositivos automodificáveis.

Correspondem a generalizações de ampliação das formulações que têm sido tradicionalmente utilizadas, permitindo reutilizá-las sem que se torne necessário aplicar-lhes modificações significativas, mantendo portanto uma relativa compatibilidade com as notações em uso ao mesmo tempo que se oferecem meios para a expressão de notações estratificadas, voltadas para formulações hierárquicas que podem servir-se de funções adaptativas com comportamento variável.

Maiores informações sobre trabalhos desenvolvidos sobre os formalismos adaptativos se encontram na página web do Laboratório de Linguagens e Técnicas Adaptativas [5].

XI. Referências

[1] Neto, J.J. - Contribuição à metodologia de construção de compiladores. São Paulo, 1993, 272p. Tese (Livre-Docência) Escola Politécnica, Universidade de São Paulo.

[2] Neto, J.J. - Adaptive automata for context-dependent languages. ACM SIGPLAN Notices, v.29, n.9, p.115-24, 1994.

[3] Neto, J.J – Adaptive devices – general formulation and case study – CIAA 2001 - 6th International Conference on Implementation and Application of Automata - Lecture Notes In Computer Science; Vol. 2494, pp. 234-250 – Pretoria, South Africa, 2001

[4] Neto, J.J. - Adaptive Technology and its Applications - in Dopico, J.R.R., J.D. de la Calle, A.P. Sierra (editors) - Encyclopedia of Artificial Intelligence - IGI Global, 2009, pp. 37-44 ISBN 978-1-59904-849-9 (hardcover) and 978-1-59904-850-5 (ebook)

[5] http://www.pcs.usp.br/~lta/ Página web do LTA – Laboratório de Linguagens e Técnicas Adaptativas.

João José Neto graduado em Engenharia de Eletricidade (1971), mestrado em Engenharia Elétrica (1975) e doutorado em Engenharia Elétrica (1980), e livre-docência (1993) pela Escola Politécnica da Universidade de São Paulo. Atualmente é professor associado da Escola Politécnica da Universidade de São Paulo, e coordena o LTA - Laboratório de Linguagens e Tecnologia Adaptativa do PCS - Departamento de Engenharia de Computação e Sistemas Digitais da EPUSP. Tem experiência na área de Ciência da Computação, com ênfase nos Fundamentos da Engenharia da Computação, atuando principalmente nos seguintes temas: dispositivos adaptativos, tecnologia adaptativa, autômatos adaptativos, e em suas aplicações à Engenharia de Computação, particularmente em sistemas de tomada de decisão adaptativa, análise e processamento de linguagens naturais, construção de compiladores, robótica, ensino assistido por computador, modelagem de sistemas inteligentes, processos de aprendizagem automática e inferências baseadas em tecnologia adaptativa.