Universidade do Minho

Escola de Engenharia

Departamento de Sistemas de Informação

Sistemas de Classificação

Ó Manuel Filipe Vieira Torres dos Santos

2000

Conteúdo

Lista de Definições

Lista de Algoritmos

Lista de Figuras

Notação

Acrónimos

AA	*Aprendizagem Automática.*
AG	*Algoritmo Genético.*
LCS	*Learning Classifier System.*
SC	*Sistema de Classificação.*
SCA	*Sistema de Classificação Antecipatório.*
SCBP	*Sistema de Classificação Baseado na Precisão.*
SCC	*Sistema de Classificação Corporativo.*
SCH	*Sistema de Classificação Heterogéneo.*
SCNZ	*Sistema de Classificação de Nível Zero.*
SCO	*Sistema de Classificação Organizacional.*
SIMD	*Single Instruction Multiple Data processor.*

Símbolos Gerais e Abreviaturas

EA	Conjunto dos estados possíveis do ambiente de um sistema de aprendizagem.
AS	Conjunto das acções que um sistema de aprendizagem é capaz de executar no ambiente.
y	Uma acção de um sistema de aprendizagem executada sobre o ambiente (saída).
M	Função que representa o comportamento de um sistema de aprendizagem

aprendizagem.

f Um estado do ambiente de um sistema de aprendizagem.

t Instante, iteração ou ciclo de tempo.

r Reacção, retorno ou retribuição do ambiente, sob a forma de recompensa ou punição, perante uma acção executada por um sistema de aprendizagem.

REA Função da reacção do ambiente.

c Sistema de Classificação.

C Conjunto (população) dos classificadores de um SC.

n Cardinalidade da população de classificadores de um SC (i.e., $|C|$).

B Lista de mensagens de um SC.

k Cardinalidade da lista de mensagens de um SC (i.e.,$|B|$).

F Função de transição de um SC.

b Uma mensagem da lista de mensagens do SC.

ID Identificador de um classificador.

COND Parte *condição* de um classificador.

$\boldsymbol{COND_c}$ Parte *condição* de um classificador *c*.

ACT Parte a*cção* de um classificador.

TY Tipo associado a um classificador: *interno* e *saída*.

ST Entalpia de um classificador. Medida do grau de utilidade de um classificador.

$\boldsymbol{ST_c}$ Entalpia de um classificador *c*.

$\boldsymbol{ST_{c,t}}$ Entalpia de um classificador *c* no ciclo *t*.

Caracter universal ou "passador" que é adicionado a um alfabeto para a construção das mensagens.

SP Medida da especificidade de um classificador de um SC.

$\boldsymbol{SP_c}$ Medida da especificidade de um classificador *c*.

l Cardinalidade de uma mensagem de um SC.

w Número de símbolos # presentes na parte *condição* de um classificador de um SC.

c Um classificador de um SC.

A Alfabeto de caracteres (símbolos) utilizados na construção das mensagens de um SC.

ca Cardinalidade de um alfabeto.

e	Vazio ou valor nulo.
$f(...), g(...)$	Funções de descodificação de palavras binárias para o correspondente inteiro.
a	Medida do erro associado a intervalos de amplitude variável.
e_T	Erro de codificação.
e_R	Erro de codificação normalizado.
X	Conjunto dos valores reais associado aos intervalos de amplitude variável.
x	Valor exacto de uma mensagem/número a codificar.
$\bar{x}$	Valor aproximado de uma mensagem.
a_i	*Bit* de ordem i numa palavra em binário.
MAG	Magnitude do valor real a representar em binário na codificação por intervalos de valores.
AMP	Amplitude dos intervalos de valores na codificação por intervalos de valores.
Fm	Função de unificação para mensagens e condições de um SC.
fm	Função de unificação entre dois símbolos de um alfabeto de um SC.
fo	Função de conversão de caracteres de um SC.
Fo	Função de conversão de uma acção numa mensagem de um SC.
f_g	Função que determina se um classificador de um SC deve colocar a sua acção na lista de mensagens.
f_z	Função que determina se um classificador de um SC gera uma saída.
F_B	Parte da função de transição F que define a nova lista de mensagens de um SC.
F_g	Parte da função de transição F que determina a saída do SC para o ambiente.
L_c	Valor do lanço de um classificador c de um SC.
b	Constante que determina qual a fracção da entalpia de um classificador de um SC que será usada no cálculo do valor do lanço.
LE_c	Valor do lanço efectivo de um classificador c de um SC.
BM	Função de Box-Muller usada no cálculo do ruído.
$N(\bar{m}, s)$	Valor do ruído tirado da distribuição normal utilizando o método de Box-Muller com média $\bar{m}$ e desvio padrão s.
B_0	População inicial de classificadores de um SC.
i	Uma mensagem do ambiente.

T	Parâmetro que indica o número de iterações ou ciclos que o SC deverá executar.
P_t	Conjunto dos classificadores de um SC que colocaram uma mensagem na lista de mensagens no ciclo *t*.
$Q_{c,t}$	Conjunto dos classificadores de um SC que colocaram mensagens na lista de mensagens e que tornaram o classificador *c* activo no ciclo *t*.
t_{vida}	Taxa de permanência ou de vida cobrada a todos os classificadores em cada ciclo de um SC.
$t_{lanço}$	Taxa de licitação cobrada aos classificadores activos num ciclo de um SC.
$R_{c,i}$	Valor total das receitas de um classificador *c* no ciclo de vida *i*.
$R_{c,est}$	Valor estabilizado da receita de um classificador *c*.
$ST_{c,est}$	Entalpia estabilizada (quando é atingido o estado estacionário) de um classificador *c*.
$L_{c,est}$	Lanço estabilizado (quando é atingido o estado estacionário) de um classificador *c*.
lic_1, lic_2	
$elic_1$,e lic_2	Parâmetros utilizados no cálculo do lanço de um classificador do SC.
$POP(t)$	População de cromossomas, indivíduos ou soluções no tempo *t* de um AG.
$Cruz$	Função de cruzamento entre dois indivíduos da população de um AG.
Mut	Função de mutação entre dois indivíduos da população de um AG.
p_m	Probabilidade da mutação.
p_c	Probabilidade do cruzamento.
H	Esquema.
$l(H)$	Comprimento de um esquema *H*.
$o(H)$	Ordem de um esquema *H*.
$m(H,t)$	Número de esquemas *H* presentes numa na população *POP(t)*.
$f(H,t)$	Desempenho médio do esquema *H* na iteração *t*.
$\overline{f(t)}$	Desempenho médio de uma população no ciclo *t*.
T_{ag}	Período de aplicação do AG a um SC.
$P_{purif}(t)$	Subpopulação de purificação para o ciclo de tempo *t*.
n_{purif}	Cardinalidade da subpopulação de purificação.
f_{purif}	Factor de purificação.

$|X|$ Cardinalidade de um conjunto X.

N^+ Conjunto dos números inteiros e positivos.

R^+ Conjunto dos números reais e positivos.

Sistemas de Classificação

Uma das vertentes da AA, porventura a mais associada a uma ciência empírica, decorre do paradigma genético - os Sistemas de Classificação (SC) . Os SC emanam do trabalho pioneiro de Holland e Reitman [Holland & Reitman 1978], sobre sistemas adaptativos. Originalmente, os SC partem da aglutinação dos Sistemas Periciais (SP) com os Algoritmos Genéticos (AG), dando corpo a uma terceira via para a análise e concepção de sistemas - sendo possível, em certas circunstâncias, materializar-lhes o comportamento (e.g., através da criação de regras de produção).

Um SC, é um sistema de aprendizagem reforçada, que combina um sistema de produção simples baseado em regras do tipo *se...então* (os classificadores), um algoritmo de distribuição de créditos, que avalia as regras utilizando princípios da Ciência Económica e das Ciências da Informação, e um AG, com um conjunto de operadores, capaz de gerar novas e possivelmente melhores regras, com base num grupo de regras pré-estabelecido, ou seja, a partir de conhecimento [Goldberg 1989] [Booker et al. 1990] [Dorigo & Maniezzo 1993].

Os SC podem ser considerados como modelos sub-simbólicos de inteligência, no sentido em que os seus componentes apenas têm significado no contexto do universo que descrevem [Forrest 1991] [Holland et al. 1986], uma posição que se situa entre as teses da IA forte [Searle 1990], em que um programa é visto como uma entidade que apenas manipula símbolos, enquanto que um cérebro lhes atribui significado, e a IA fraca, em que os programas, sistemas ou máquinas, são considerados verdadeiras mentes [Fodor 1975]. Estes sistemas podem, assim, vir a ser usados não só como modelos cognitivos, mas também como mecanismos que possibilitam a criação de sistemas computacionais para o tratamento de processos do foro psicológico ou para meros processos de inferência.

Os SC têm sido aplicados em variados domínios, nomeadamente na resolução de problemas em controlo, reconhecimento de escrita, medicina, ciências sociais, robótica, aprendizagem de conceitos, optimização, simulação, aprendizagem latente, modelação do comportamento de organismos adaptativos e vida artificial, sistemas multi-agente, economia computacional, música, e "data mining" .

Os Sistemas de Classificação (SC) são sistemas de aprendizagem que se regem por comportamentos cujo objectivo principal é maximizar o seu conhecimento tendo como fim a resolução de um(ns) problema(s), alcançando ganhos ou reforços através da interacção com um dado universo de discurso, o que passa por um processo de reformulação da sua base de conhecimento materializada num corpo de regras do tipo *condição-acção*, chamadas classificadores. Através de uma evolução selectiva defendida pela teoria *Darwiniana*[1], os classificadores que se revelam úteis para os propósitos do sistema são preservados e podem propagar-se através de um mecanismo genético, substituindo gradativamente os classificadores menos úteis. Este processo vai, aos poucos, melhorando o desempenho do SC no ambiente em que se integra, i.e., adequa a sua programação ao problema que se pretende resolver.

Os SC aplicam uma filosofia de aprendizagem designada como reforçada[2] e é baseada em exemplos. Aos SC são apresentados exemplos por parte de um ambiente, em conjunto com informação que complementa a do exemplo, informação esta que indica se a resposta do sistema é ou não aceitável (sendo apenas fornecida após o sistema ter apresentado uma solução para o problema) - daí o nome de aprendizagem reforçada (Figura 1).

Definição 1 (Sistema de Aprendizagem Reforçada).
Para definir formalmente um sistema de aprendizagem reforçada é necessário definir, por um lado, o sistema de aprendizagem, e por outro, o ambiente onde este se insere [Bart 1994]. Considerem-se dois conjuntos: *EA*, o conjunto dos estados do ambiente e

[1] Charles Darwin (1809-1882). Descreveu na sua obra *Sobre a Origem das Espécies por Meio da Selecção Natural* os princípios da evolução das espécies.

[2] A abordagem proferida pela aprendizagem reforçada situa-se entre outras duas abordagens, a saber: a aprendizagem superisionada e a aprendizagem não-supervisionada [Banzhaf et al. 1998].

AS, o conjunto das acções que o sistema de aprendizagem é susceptível de executar sobre o ambiente. O sistema de aprendizagem pode então ser definido através de uma função (possivelmente estocástica) $Y=M(f,t)$ onde $Y \in EA$ é a acção que o sistema executa, $f \in AS$ é o estado do ambiente e $t \in N^+$ o tempo (instante, iteração ou ciclo) em que tal acção ocorreu.

A reacção do ambiente pode ser agora descrita por uma função (possivelmente estocástica) $r=REA(Y,f,t)$ na qual Y,f e t têm a interpretação dada em epígrafe e $r \in \mathbf{R}^+$ é a recompensa (retorno ou retribuição) do ambiente. Esta é uma medida da utilidade que a acção proposta pelo sistema de aprendizagem teve para com o ambiente (quanto maior for melhor). Um sistema aprende, por conseguinte, quando:

$$\acute{a}REA(M(f_t,t),f_t,t)\tilde{n} > \acute{a}REA(M(f_{t-1},t-1),f_{t-1},t-1)\tilde{n}$$

onde "á" e "ñ" são usados para denotar o valor esperado de uma função. As variáveis f_t e f_{t-1} são estados do ambiente nos tempos t e $t-1$, respectivamente.

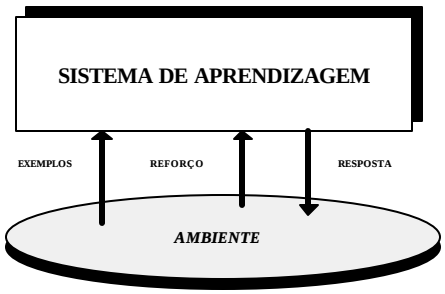


Figura 1 Sistema de aprendizagem reforçada.

Os sistemas de classificação [Goldberg 1989] [Booker et al. 1990] são sistemas de aprendizagem massivamente paralelos[3] baseados em regras sintácticas simples, sendo estas constituídas por sequências de caracteres, os classificadores, que dão corpo ao sistema e que evoluem (adaptam-se) através de um algoritmo de atribuição de

[3] No sentido em que as regras são analisadas e disparadas simultanemante (não existe nenhum grau de dependência entre elas).

créditos (e.g., através do "Bucket Brigade Algorithm"), e da descoberta de novas regras (através do AG).

Os SC baseiam-se nos trabalhos iniciais de [Holland 1975], [Holland & Reitman 1978]. Tendo esta tecnologia surgido no início dos anos setenta, pode dizer-se que está a atingir agora a maioridade. Desde essa altura, porém, a sua arquitectura pouco evoluiu, ao contrário dos algoritmos de aprendizagem e dos métodos e sua integração com os SC [Wilson & Goldberg 1989]. Os SC inserem-se naturalmente na área da programação genética e têm vindo a conhecer, progressivamente, um êxito cada vez mais consolidado nos meios científicos e de engenharia (Anexo G).
Estes sistemas operam, tipicamente, em ambientes que possuem uma ou mais das seguintes características:

- há sempre um grande número de novos acontecimentos os quais são acompanhados de grandes volumes de dados irrelevantes, vulgo ruído;
- obedecem a requisitos de actuação contínua e em tempo real;
- os objectivos a atingir estão definidos implicitamente ou de forma inexacta;
- a recompensa (retorno ou retribuição) do ambiente é obtida de uma forma intermitente, ou só após a realização de longas sequências de acções sobre este por parte do SC.

Os SC são desenhados por forma a absorverem nova informação destes ambientes de uma forma contínua, tratando conjuntos de hipóteses que se apresentam em competição (expressas através das regras) sem perturbarem significativamente as capacidades (conhecimento) já adquiridas.

1 Constituição de um Sistema de Classificação

Um SC (Figura 2) é composto por três componentes básicos:

- Subsistema de regras e mensagens; e

- Subsistema de distribuição de créditos (e.g., o algoritmo "Bucket Brigade"); e
- Subsistema de descoberta de regras (i.e., o AG).

Ao nível mais básico tem-se o motor computacional. Este motor é a componente fundamental do sistema e corresponde a um sistema paralelo de passagem de mensagens baseado em regras. Este subsistema é um caso especial dos sistemas de produção. Contém uma base de conhecimento com regras do tipo:

se condição então acção

O significado de uma regra destas é o de que a *acção* pode ser executada (a regra é disparada) quando a *condição* é satisfeita. Às regras dá-se o nome de classificadores. Os SC partem desta representação do conhecimento restringindo as regras a um tamanho fixo. Esta restrição traz dois benefícios. Primeiro, todas as sequências de caracteres dentro do alfabeto permitido são sintacticamente significativas. Segundo, uma representação fixa das sequências de caracteres permite o uso de operadores do tipo genético. Isto deixa o caminho aberto para o uso do AG no espaço de procura de regras.

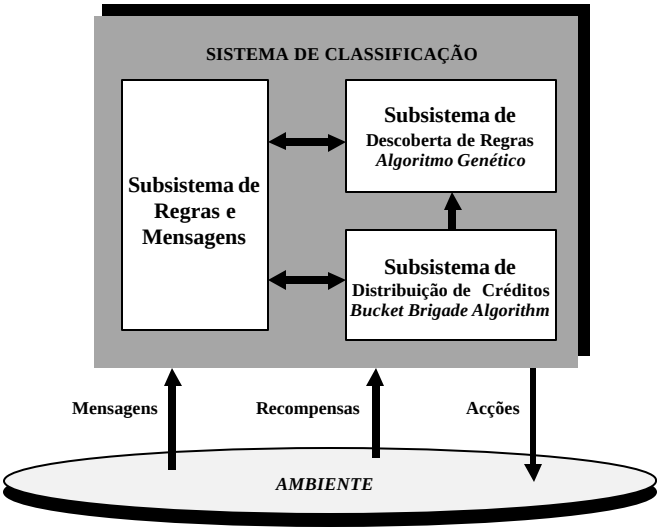


Figura 2 Estrutura funcional de um sistema de classificação.

O valor de uma regra relativamente às outras regras é uma das peças de informação que deverá ser aprendida. Para facilitar este tipo de aprendizagem, os SC forçam os classificadores a coexistirem num serviço de economia baseado na informação. A competição dá-se entre os classificadores e tem como objectivo garantirem o direito de responderem às mensagens emanadas do ambiente. Os classificadores que oferecerem melhores lanços terão essa prerrogativa, com o consequente pagamento a servir de fonte de crédito para os classificadores vencedores no ciclo anterior. Deste modo é estabelecida uma cadeia de intermediários entre o fabricante (os detectores), e o consumidor (acções enviadas ao ambiente e retribuição). A natureza competitiva desta economia assegura que as regras que aumentem a entalpia[4] do sistema sobrevivem, sendo as restantes eliminadas.

Este algoritmo de distribuição de créditos introduz uma espécie de circuito monetário dentro do SC. A troca e acumulação de créditos possibilita a aplicação dos AG. Usando o *saldo bancário* dos classificadores como uma função de valorização, os classificadores podem ser reproduzidos, cruzados, e mutados. Logo, o sistema não só aprende pela valorização das regras, como também pode descobrir novas regras, e possivelmente melhores regras por manipulação e combinação das pré-existentes.

Com uma distribuição de créditos entre classificadores a ser feita de uma forma competitiva, e a descoberta de novas regras o ser por utilização do AG, tem-se uma base de sustentação para construir sistemas de aprendizagem alicerçados em classificadores.

Definição 2 (Sistema de Classificação).

Formalmente, um sistema de classificação c é um tupleto da forma $\langle C, B, F \rangle$, no qual:

- C corresponde a um conjunto de classificadores, cuja cardinalidade, dada por $|C|$ (ou n), é um parâmetro do sistema; e
- B corresponde a um conjunto de mensagens, cuja cardinalidade, dada por $|B|$ (ou k), é um parâmetro do sistema; e

[4] Característica de um sistema termodinâmico que é proporcional à sua energia interna. Neste contexto corresponde ao capital (valor económico) acumulado no sistema ou seja, à soma dos valores da entalpia de cada

- *F* é uma função de transição, dada na forma *F(C, B) ® (B', Y)*, que converte um conjunto de classificadores *C* e uma lista de mensagens *B*, na lista de mensagens *B'* e na operação *Y*. A saída *Y* pode eventualmente ser uma saída vazia.

2 Subsistema de Regras e Mensagens

O subsistema de regras e mensagens forma o coração computacional do SC (Figura 3). Os blocos principais deste subsistema são a seguir enumerados:

- um conjunto *C* com *n* regras, denominadas de classificadores; e
- uma lista de mensagens *B* de dimensão *k*, mensagens estas que são enviadas pelos classificadores e pelo ambiente; e
- uma interface com o ambiente composta pelos detectores que recebem mensagens vindas do ambiente e pelos repercussores ou actuadores que enviam mensagens para o ambiente; e
- subsistemas de licitação e de resolução de conflitos, que identificam quais os classificadores que devem ser activados em cada ciclo, e quais de entre estes deverão efectivamente enviar as suas mensagens para o ambiente do sistema.

A informação passa do ambiente para o SC através dos detectores ou sensores (os olhos e ouvidos do SC) onde é descodificada, dando origem a uma ou mais mensagens de tamanho finito. Estas mensagens são então colocadas na lista de mensagens, podendo a partir daí activar os classificadores. A este processo chama-se unificação. Quando activado, um classificador torna-se candidato a colocar a sua mensagem na lista de mensagens, a qual pode então invocar outros classificadores ou levar os repercussores do SC a actuarem sobre o ambiente.

Desta forma os classificadores definem qual a política a seguir pelo SC; i.e., para onde ir e como agir. De uma certa forma o subsistema de regras e mensagens coordena o fluxo de informação desde o ponto onde é detectada (detectores), até ao

um dos classificadores contidos no sistema.

ponto onde é processada (lista de mensagens e conjunto de classificadores), e desde aqui até ao ponto onde se executam as acções sobre o ambiente (repercussores).

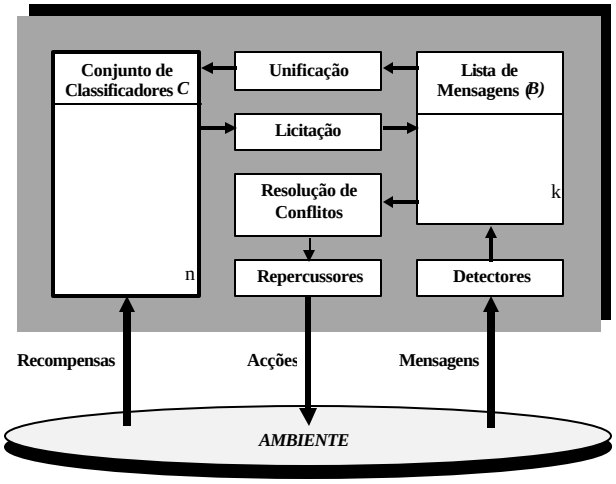


Figura 3 Subsistema de regras e mensagens.

2.1 Mensagens, Classificadores e Estratégias de Codificação

Para uma melhor compreensão do funcionamento do subsistema de regras e mensagens, há que observar as suas fontes de conhecimento (e a forma em que este é processado); ou seja, há que prestar particular atenção às mensagens e aos classificadores.

Definição 3 (Mensagens).

Uma mensagem *b* dentro do SC não passa de uma sequência de caracteres de tamanho finito, dada num alfabeto finito *A* de cardinalidade *ca*. Em termos do alfabeto binário (caracteres 0 e 1), *A={0, 1}*, *b* é uma sequência da forma $\{0,1\}^l$ $(l \in N^+)$ onde *l* é o comprimento da mensagem (número de caracteres), e N^+ o conjunto dos números inteiros e positivos. "*{*" e "*}*" é a notação usada para tipificar conjuntos. As mensagens correspondem à unidade atómica usada na troca de informação num SC e ocorrem em vários pontos do sistema e em diferentes formas: como uma entrada

proveniente do ambiente do sistema; como uma saída do sistema para o seu ambiente; como a parte *condição* de um classificador; e como a parte *acção* de um classificador. Convém ainda referir que as mensagens na lista de mensagens podem unificar com um ou mais classificadores.

Definição 4 (Classificadores).

Um classificador *c* é um tupleto *áID, COND, ACT, TY, ST, SPñ,* no qual:

- *ID* é o identificador do classificador (identifica univocamente um classificador).
- *COND* é a parte *condição*. Trata-se de uma sequência de caracteres na forma $(A \cup \{\#\})^{l} (l \in \mathbb{N}^{+})$. O símbolo # é chamado de caracter universal, pois unifica com qualquer caracter do alfabeto *A*. Por exemplo, a sequência 00##11 unifica com as sequências de caracteres 000011, 000111, 001011 e 001111.
- *ACT* é a parte *acção*. Corresponde a uma acção a ser executada pelo SC sobre o ambiente ou a ser processada no próximo ciclo. É dada por uma sequência de caracteres na forma $(A \cup \{\#\})^{l} (l \in \mathbb{N}^{+})$. Neste caso o símbolo # é chamado de símbolo *passador*, pois permite a passagem de caracteres da mensagem unificadora para a componente *acção* de um classificador. Para ilustrar este mecanismo atenda-se à Figura 4.

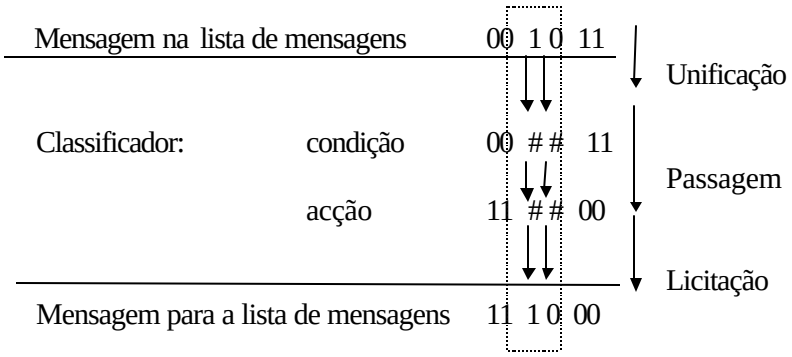


Figura 4 Relação entre uma mensagem e um classificador.

- *TY* $\hat{I}$ *{interno, saída},* o tipo do classificador. *saída* denota que a acção a executar pelo SC é sobre o ambiente, *interno* denota que a acção a executar pelo SC é sobre o próprio SC.
- *ST* $\hat{I}$ $\boldsymbol{R}^+$, uma medida do grau de utilidade do classificador para o sistema.
- *SP* $\hat{I}$ $\boldsymbol{R}^+$, dá uma medida da especificidade ou grau de generalidade do classificador. Este parâmetro pode ser definido de diferentes formas, variando contudo na razão inversa do número de mensagens com que unifica um dado classificador. Uma possível definição para *SP* é dada por:

$$SP = \frac{l - w}{l}$$

em que *w* é o número de símbolos *#* na parte *condição* do classificador, e *l* é igual ao seu número total de caracteres. A especificidade do classificador fornece informação acerca do grau de generalidade da regra. Um classificador com o valor (máximo) de especificidade *SP* = 1 só será activado quando na lista de mensagens existir uma mensagem exactamente igual à parte *condição* do classificador; pelo contrário, um classificador com *SP* = 0 é activado, quaisquer que sejam as mensagens na lista de mensagens.

Note-se que em todas estas definições, *l*, a cardinalidade das mensagens, das condições e acções dos classificadores, tem o mesmo valor. É um parâmetro do sistema. Os símbolos 0,1, e # são caracteres. O caracter # confere generalidade aos classificadores. Considerando que *ca* é a cardinalidade do alfabeto *A*, então o número de mensagens distintas que poderão ajustar-se a um classificador que contenha *l* caracteres # será de ca^l.

Segundo Goldberg [Goldberg 1989], devem-se ter em conta dois princípios básicos na escolha do sistema de codificação e do alfabeto a utilizar na aplicação dos SC à resolução de problemas: o princípio de "meaningful building blocks" e o princípio de "minimal alphabets", para cuja compreensão é necessário recorrer ao conceito de esquema, cuja definição formal será introduzida na Secção 4.5.

Informalmente, um esquema *H,* é uma representação de cada uma das sequências de caracteres obtidas a partir de *H*, substituindo o símbolo #, pelos símbolos do alfabeto escolhido, segundo todas as combinações possíveis. Dado um esquema *H,* o comprimento de *H*, *l(H),* é dado pela distância, em *H*, entre os dois símbolos mais afastados e diferentes do símbolo #. A ordem de *E*, *o(H),* é dada pelo número total de símbolos em *H*, subtraído do número de símbolos #, nessa mesma sequência. O número de esquemas depende da cardinalidade do alfabeto considerado. Para uma dada mensagem, de comprimento *l*, o número de esquemas possíveis, considerando o alfabeto binário *A={0,1}*, será 3^l, e $(ca+1)^l$ para o caso de um alfabeto *A* de cardinalidade *ca*. Por exemplo, o esquema *H=00#0#1* tem como comprimento *l(E)=1*, uma ordem de *o(E)=4*, e um número de esquemas possíveis de $6^3=216$.

Voltando ao primeiro princípio, este postula que se devem escolher códigos, de tal modo curtos, que os esquemas de menor ordem sejam os mais relevantes para a resolução do problema em causa e, não necessariamente condicionados pelos restantes esquemas. É evidente que o alfabeto binário é o alfabeto que possibilita o maior número de esquemas por *bit* de informação na codificação de mensagens.

O segundo princípio postula que se deve escolher um alfabeto que não só permita uma caracterização formal e natural do problema considerado, mas que apresente também a menor cardinalidade. Para ilustrar este princípio, considere-se que se pretende codificar um vector de valores inteiros usando um alfabeto binário. Cada elemento desse vector é codificado usando o mesmo número de *bits*, ou alternativamente, usando alfabetos definidos para uma dada gama de valores inteiros ou reais (cada elemento do vector é codificado por um valor inteiro ou real). Por exemplo, o vector <1,0,4,5> será codificado em binário com sub-sequências de 4 bits, ou usando inteiros, nas mensagens [0001 0000 0100 0101] e [1 0 4 5], respectivamente. Como se pode verificar o número de esquemas em binário é superior ao número de esquemas com inteiros, tal como já tinha sido mencionado. Note-se, porém, que com quaisquer das bases utilizadas representaram-se os mesmos valores; no entanto, alfabetos com diferentes cardinalidades requerem, mensagens de diferentes comprimentos.

A construção de mensagens e de classificadores pode ser feita, assim, com diferentes alfabetos com recurso a diferentes mecanismos de codificação. A qualidade das

mensagens e dos classificadores depende do mecanismo de codificação utilizado, assim como do comprimento das referidas representações, pelo que irão ser analisados diferentes mecanismos de codificação em termos de cardinalidade das representações e dos erros associados aos valores representados.

O desempenho de um SC depende do tamanho das mensagens e dos classificadores utilizados. Quanto mais longas forem as mensagens e os classificadores, maior será o peso computacional associado e, consequentemente, menor o desempenho do sistema. Por outro lado, quando a informação que se pretende codificar é finita, é possível representá-la mediante mensagens com comprimento fixo. No entanto, quando a informação a representar não se encontra completamente tipificada, não é possível a representação exacta de todos estes valores usando uma mensagem de tamanho fixo, pelo que as representações terão sempre associadas uma medida de erro, que pode tomar a forma:

$$e_T = x - \overline{x}$$

onde e_T é usado para designar a medida do erro (o valor do erro), x é um valor exacto, e $\overline{x}$ é o valor aproximado do parâmetro a representar. Esta definição não tem em linha de conta a ordem de grandeza das variáveis em jogo, pelo que há que proceder a uma correcção da medida de erro, feita na forma (normalização) :

$$e_R = \frac{e_T}{x} = \frac{x - \overline{x}}{x}$$

No que se segue, são apresentadas algumas estratégias de codificação de mensagens, em termos de alfabetos binários [Booker 1991], estratégias estas, contudo, que podem ser alteradas de forma a poderem ser usadas com outros alfabetos. Para cada uma destas estratégias é descrita a codificação de valores inteiros.

Definição 5 (Codificação por Contagem Simples).
Este é um dos algoritmos menos complexos, em que se procede por simples contagem dos símbolos presentes nos esquemas. O valor codificado corresponde ao número de

ocorrências do símbolo 1 na respectiva representação binária, e aplica-se quando estão em jogo valores inteiros e positivos, tendo em atenção que:

- usando l *bits*, podem-se representar l+1 valores distintos;
- usando l *bits*, $a_1,...,a_l$, podem-se representar os inteiros e positivos dados pelo conjunto $\{0,...,l\}$;
- para se obter a representação em binário, $a_1,...,a_l$, de um valor inteiro e positivo x, atende-se a que $a_i = \begin{cases} 1 \text{ se } i > l - x \\ 0 \text{ se } i \leq l - x \end{cases}$ com $1 \leq i \leq l$.

A função de descodificação de um valor em binário para o correspondente valor inteiro e positivo, é dada na forma:

$$f(a_1,...,a_l) = \sum_{i=1}^{l} a_i$$

Por exemplo, sendo l=4, *0011* e *1111* sequências em binário, obtém-se $f(0011)$=2 e $f(1111)$=4.

Para codificar valores inteiros, positivos ou negativos, tem-se em atenção que:

- usando $2l$ *bits*, $a_{-l+1},...,a_0, a_1,...,a_l$, podem-se representar os inteiros dados pelo conjunto $\{-l,...,0,...,l\}$;
- para se obter a representação em binário, $a_{-l+1},...,a_0, a_1,...,a_l$, do inteiro x, atende-se a que:

$$a_i = \begin{cases} 1 \text{ se } (i \geq -x+1 \wedge x > 0) \vee (i \geq -x \wedge x < 0) \\ 0 \text{ se } (i < -x+1 \wedge x > 0) \vee (i < -x \wedge x < 0) \vee x = 0 \end{cases}$$

com $-l+1 \leq i \leq l$.

Por exemplo, seja l=6, e {4,-2,6,0,-1,3} o conjunto de valores a codificar. Para a presente situação, obtém-se a representação em binário *001111111111* para o 4, *000000011111* para o –2, *111111111111* para o 6, *000000000000* para o 0, *000000111111* para o –1 e *000111111111* para o 3.

A função de descodificação de um valor em binário para o correspondente valor inteiro, é dado na forma:

$$g(a_{-l+1},\ldots,a_0,a_1,\ldots,a_l)=\begin{cases}0 \text{ se } S=0\\ S-l-1 \text{ se } S\neq 0 \wedge S\leq l\\ S-l \text{ se } S\neq 0 \wedge S>l\end{cases}$$

$$\text{com } S=\sum_{i=-l+1}^{l} a_i \ .$$

Por exemplo, sendo l=2, e *0011* e *1111* as sequências em binário, obtém-se g*(0011)*=-1 e *g(1111)*=2.

Note-se que estes algoritmos de codificação utilizam apenas um número fixo de *bits* sendo por conseguinte passíveis de erros de arredondamento. Utilizando um número limitado de *bits*, existem valores inteiros positivos e negativos que não podem ser representados (apenas se podem representar l+1 valores distintos). Para representar um maior número de valores inteiros há que aumentar consideravelmente o número de *bits* utilizados.

Definição 6 (Codificação Binário Padrão).

Este algoritmo baseia-se na codificação binário padrão. Para a codificação de valores inteiros e positivos, tem-se em atenção que:

- usando l *bits*, podem-se representar 2^l valores distintos;
- usando l *bits*, $a_1,\ldots,a_l$, podem-se representar os inteiros e positivos dados pelo conjunto $\{0,\ldots, 2^l-1\}$;
- para se obter a representação em binário $a_1,\ldots,a_l$, de um valor inteiro e positivo x, atende-se a que:

$$a_i = (x / 2^{i-1}) \,//\, 2^i$$

com $1\leq i\leq l$, onde / representa o quociente inteiro da divisão e // representa o resto inteiro dessa mesma divisão.

A função de descodificação de um valor em binário para o correspondente valor inteiro e positivo, é dada na forma:

$$f(a_1,\ldots,a_l)=\sum_{i=0}^{l-1} a_{l-i}\cdot 2^i$$

Por exemplo, sendo l=4, e 0111 e 1111 as sequências em binário, obtém-se $f(0011)$=3 e $f(1111)$=15.

Para codificar valores inteiros, positivos ou negativos, tem-se em atenção que:

- usando l+1 *bits* $(a_0,a_1,\ldots,a_l)$ onde a_0 representa o *bit* de sinal, podem-se representar os inteiros dado pelo conjunto $\{-2^l,\ldots,0,\ldots,2^l-1\}$;
- para se obter a representação em binário, $a_0,a_1,\ldots,a_l$ de um valor inteiro x, atende-se a que:

$$a_i=\begin{cases}(x/2^{i-1})//2^i & \text{se } x\geq 0\\ ((2^b-x)/2^{i-1})//2^i & \text{se } x<0\end{cases}$$

$$\text{com } 1\leq i\leq l \text{ e } a_0=\begin{cases}0 \text{ se } x\geq 0\\ 1 \text{ se } x<0\end{cases}$$

Por exemplo, seja l=3, e {4,-2,6,0,-1,3} o conjunto de valores a codificar. Para a presente situação, obtém-se a representação em binário *0011* para o 4, *1110* para o –2, *0110* para o 6, *0000* para o 0, *1111* para o –1 e *0011* para o 3.

A função de descodificação de um valor em binário para o correspondente valor inteiro, é dada na forma:

$$g(a_0,a_1,\ldots,a_l)=\begin{cases}\displaystyle\sum_{i=0}^{l-1} a_{l-i}\cdot 2^i & \text{se } a_0\geq 0\\ \displaystyle -2^l+\sum_{i=0}^{l-1} a_{l-i}\cdot 2^i & \text{se } a_0<0\end{cases}$$

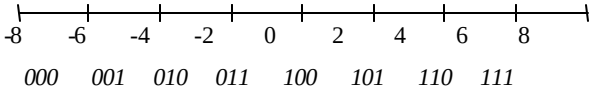
Por exemplo, sendo l=3, e *0011* e *1111* as sequências, obtém-se $g(0011)$=3 e $g(1111)$=-1. Note-se que estes algoritmos, tal como os anteriores, utilizam apenas um número fixo de *bits*, originando por conseguinte problemas com operações de arredondamento. Utilizando um número limitado de *bits*, existem valores inteiros, positivos e negativos, que não podem ser representados (podem-se representar apenas 2^l valores distintos). Para representar um maior número de valores inteiros há que aumentar consideravelmente o número de *bits* utilizados.

Definição 7 (Codificação por Intervalos de Valores).

Este algoritmo permite representar números reais, utilizando um número fixo de *bits*. Obviamente, com este algoritmo não é possível representar exactamente um dado valor, no entanto permite dar uma aproximação com uma certa margem de erro. Assim, para codificar números reais, positivos ou negativos, tem-se em atenção que:

- usando l *bits*, pode-se representar, em princípio, qualquer número, a menos de um certo erro;
- usando l *bits* $a_1,\ldots,a_l$, existem 2^l representações binárias distintas;
- para se obter a representação em binário $a_1,\ldots,a_l$, de um número real x de magnitude $-MAG \leq x < MAG$, calcula-se a amplitude dos intervalos $AMP = \frac{MAG}{2^{l-1}}$;
- é possível definir a sequência de intervalos de valores de amplitude AMP na forma $]MAG-AMP, MAG]$, $]MAG-2AMP, MAG-AMP]$, ... , $]0, AMP]$, $[-AMP, 0[$, ..., $[-MAG+AMP, -MAG+2AMP[$, $[-MAG, -MAG+AMP[$;
- a cada intervalo de valores de amplitude AMP corresponde uma única representação em binário;
- se um número x pertence a um intervalo da sequência de intervalos definida anteriormente, então a correspondente representação em binário é a representação em binário do número x.

Por exemplo, seja l=3, MAG=8 e {4,-2,6,0,-1,3} o conjunto de valores a codificar. Atendendo a que AMP=2 é então possível definir a sequência de oito intervalos:



Para a presente situação, obtém-se a representação em binário *110, 011, 111, 100, 011* e *101*.

A função de descodificação de um valor em binário para o equivalente real, é dada na forma:

$$f(a_1,\ldots,a_l) = MAG\cdot\left(\frac{1}{2^{l-1}}\cdot x' - 1\right) = AMP\cdot x' - MAG$$

$$\text{com } x' = \sum_{i=0}^{l-1} a_{l-i}\cdot 2^i \,.$$

Por exemplo, sendo l=4 e MAG=8, obtém-se $f(0011)$=-5 e $f(1111)$=7.

As medidas de erro associadas a este tipo de codificação são limitadas por (números reais):

$$|e_T| = |x - \overline{x}| \le \frac{MAG}{2^{l-1}}$$

$$|e_R| = \frac{|x - \overline{x}|}{|x|} \le \frac{MAG}{2^{l-1}\cdot|x|}$$

Definição 8 (Codificação por Intervalos de Amplitude Variável).

Com este algoritmo é possível representar e comparar pares de reais, em que a medida de erro é codificada através de um número fixo de *bits*. Obviamente, não se está a falar de representações em valores absolutos, mas na possibilidade de estabelecer uma relação de ordem entre os valores considerados para codificação. Ter-se-á, por conseguinte, que atender a que:

- usando l *bits*, pode-se representar, em princípio, qualquer número, a menos de um certo erro;
- usando l *bits*, $a_1,\ldots,a_l$, existem 2^l representações binárias distintas;

$$|e_R| = \frac{|x - \vec{x}|}{|x|} \leq \frac{MAG}{2^{l-1} \cdot m}$$

em que $|x| \geq m, \forall x \in X$.

2.2 Lista de Mensagens e Procedimento para o Subsistema de Regras e Mensagens

Após uma viagem pelo mundo das mensagens e métodos para a sua codificação, é altura de lançar um olhar pelos restantes componentes e processos envolvidos no subsistema de regras e mensagens do SC.

Definição 9 (Lista de Mensagens).

A lista de mensagens, *B*, de um SC é um conjunto de tupletos da forma *áb, cñ*, em que *b* é uma mensagem e *c* o classificador responsável pela sua colocação em *B* (*c Î C È {e}*; *C* é o conjunto dos classificadores); i.e., *c* pode ser o classificador ε (nulo), o que ocorre quando a mensagem é proveniente do ambiente. A cardinalidade de *B* é *k*, um parâmetro do sistema.

Definição 10 (Unificação).

É possível definir uma função *fm* para a unificação entre dois símbolos de um alfabeto. Para o caso particular do alfabeto binário, tem-se que *fm:{0, 1}* x *{1, 0, #}* ® *{falso, verdadeiro}*, dada na forma:

fm(0, 0) = verdadeiro
fm(0, 1) = falso
fm(0, #) = verdadeiro
fm(1, 0) = falso
fm(1, 1) = verdadeiro
fm(1, #) = verdadeiro

em que x denota o produto cartesiano de dois conjuntos. O caracter # na parte da condição da regra (*COND*) é chamado de caracter universal uma vez que unifica com qualquer outro caracter. É possível então estabelecer a função de unificação de uma mensagem *X* com uma condição *Y*, ambas em binário, ou seja $Fm: \{0, 1\}^l \times \{0, 1, \#\}^l \rightarrow \{verdadeiro, falso\}$, dada na forma:

$$Fm(X, Y) = \bigwedge_{i=1}^{l} fm(x_i, y_i)$$

em que $l \in \mathbf{N}^+$ e (x_i,y_i) denota o elemento de ordem i de (X,Y). Uma condição y é satisfeita por uma mensagem x, se $Fm(x, y) = verdadeiro$. Um classificador c é activado se $\exists_b \in B \wedge Fm(b, COND_c)=verdadeiro$.

A parte *condição* de um classificador unifica com o conteúdo de uma mensagem *sse* todas as posições em *0* na condição casem com um *0* na mensagem; as posições em *1* casem com um *1*; e as posições em *#* casem com um *1* ou um *0* (e.g., uma condição de quatro posições, *COND=#01#*, unifica com a mensagem *b=0010*, mas não unifica com a mensagem *b=0000*). Uma vez que a operação de unificação esteja concluída o classificador torna-se num candidato a colocar a sua parte *acção* na lista de mensagens, no próximo ciclo. O sucesso desta operação é, contudo, condicionado pelo resultado de uma operação de licitação, que por sua vez depende do cálculo do valor ou peso do classificador, e que ocorre entre os classificadores que foram seleccionados durante o processo de unificação.

Para uma melhor compreensão do funcionamento do subsistema de regras e mensagens, passar-se-á a simular a operação de unificação num SC, tendo apenas em linha de conta a lista de mensagens e o conjunto de classificadores (Figura 5). Suponha-se que logo no primeiro ciclo, a mensagem recebida do ambiente, *0111*, está contida na lista de mensagens. Esta mensagem unificará, então, com a parte *condição* do classificador 1, que então coloca a mensagem *0000* na lista de mensagens (i.e., a sua componente *acção*) para ser processada no próximo ciclo. Esta mensagem, por seu turno, unifica, no segundo ciclo, com as regras 2 e 4, que colocam as suas

mensagens, respectivamente *1100* e *0001*, na lista de mensagens. A mensagem *1100*, por seu lado, unifica, no terceiro ciclo, com os classificadores 3 e 4. Destes dois classificadores, apenas a mensagem enviada pelo classificador 3, *1000*, unifica com o classificador 4, no quarto ciclo.

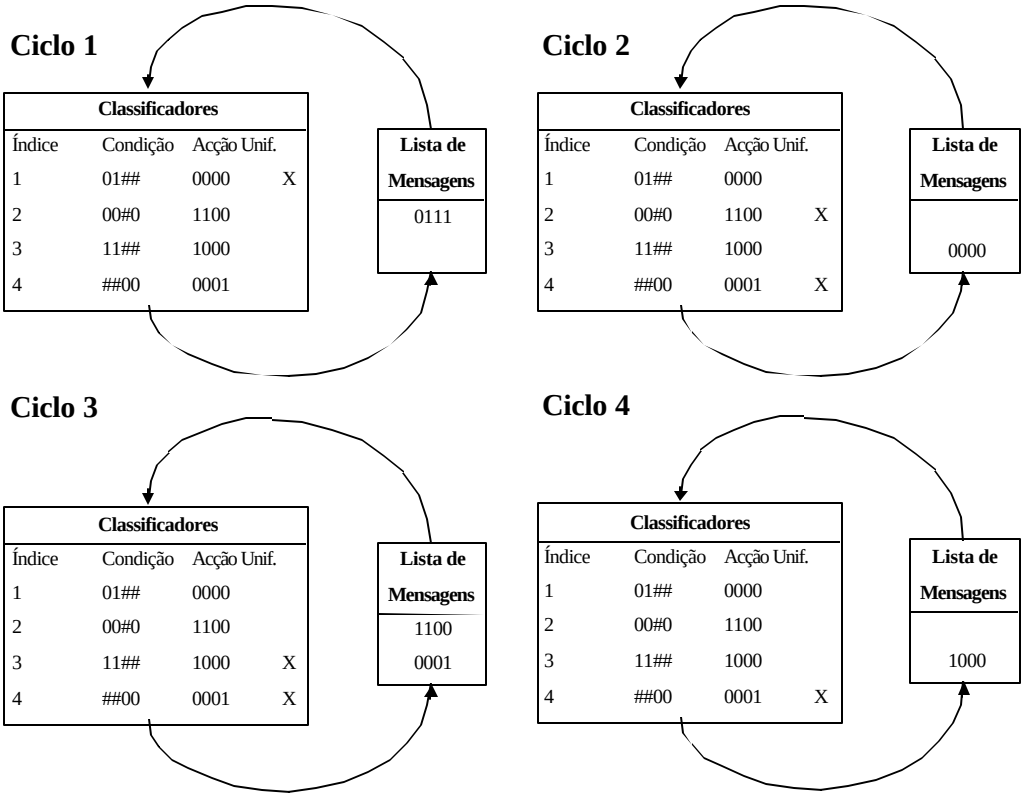


Figura 5 Operação de unificação.

Definição 11 (Conversão de uma Acção numa Mensagem).

Sendo a função *Fo* utilizada para converter uma mensagem, com uma parte *condição* que pode conter caracteres universais (#) e uma parte *acção* possivelmente com alguns caracteres passadores (#) numa nova mensagem, há que recorrer a uma função de conversão de caracteres *fo*. Para o caso do alfabeto em binário, esta função tem a forma $fo: \{0, 1, e\} \times \{0, 1, \#\} \times \{0, 1, \#\} \rightarrow \{0, 1\}$, em que:

$fo(x, 0, 0) = 0$

$fo(x, 0, 1) = 1$

fo(x, 0, #) = 0

fo(x, 1, 0) = 0

fo(x, 1, 1) = 1

fo(x, 1, #) = 1

fo(x, #, 0) = 0

fo(x, #, 1) = 1

fo(x, #, #) = x se x é 0 ou 1.

Se *x=e* a saída tanto poderá ser 0 como 1, ambas com a mesma probabilidade de ocorrência. É possível então representar a função de conversão de uma mensagem numa nova mensagem após o processo de unificação, por forma a que obedeça ao enunciado em epígrafe, e que é dada na forma $Fo:(\{0,1\}^l \text{ È } e) \times \{1,0,\#\}^l \text{ ® } \{0,1\}^l$; ou seja, a nova mensagem é $b' = F_0(b, COND, ACT)$, onde cada posição de *b'* é determinada da seguinte forma:

$$\forall b_i' \in b' : b_i' = f_0(b_i, COND_i, ACT_i) \qquad para\, i = 1,...,l$$

na qual b'_i, b_i, $COND_i$ e ACT_i são, respectivamente, os símbolos de ordem *i* de *b* (uma mensagem), *b'* (a nova mensagem), *COND* (a parte *condição* do classificador) e *ACT* (a parte *acção* do classificador). Como exemplo, considere-se que se pretende converter a mensagem *b=0111* contida na lista de mensagens, numa nova mensagem b', que irá substituir a anterior na lista de mensagens para o próximo ciclo, por unificação com um classificador. Este procedimento passa pela aplicação da função *Fo*, o que é dado na forma (Figura 6):

- $b=0111$, $(b_1=0,\ b_2=1,\ b_3=1,\ b_4=1)$, é a mensagem contida na lista de mensagens; e
- $COND=01\#\#$, $(COND_1=0,\ COND_2=1,\ COND_3=\#,\ COND_4=\#)$, é a parte *condição* do classificador; e
- $ACT=000\#$, $(ACT_1=0,\ ACT_2=1,\ ACT_3=\#,\ ACT_4=\#)$, é a parte *acção* do classificador; e
- $b'=0001$, $(b_1=0, b_2=0, b_3=0, b_4=1)$, é a nova mensagem.

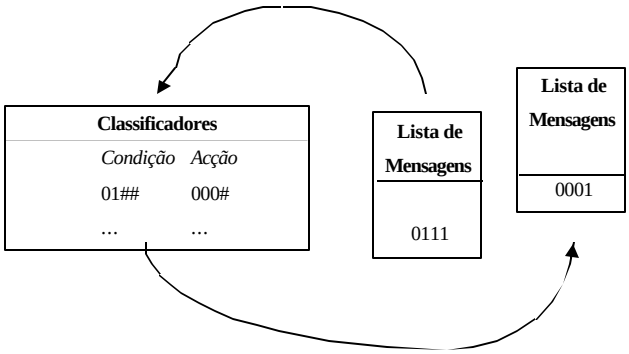


Figura 6 Aplicação da função *Fo*.

Definição 12 (Função de Transição).

Define-se a seguir uma função *F* que converte uma lista de mensagens *B*, numa nova lista *B'*, para um novo ciclo do SC. São ainda definidas duas novas funções:

- $f_g(c,C) \rightarrow \{ verdadeiro, falso\}$; *e*
- $f_z(c,C) \rightarrow \{ verdadeiro, falso\}$.

tal que por intermédio de $f_g(c,C)$ decide-se se um dado classificador deve ou não colocar o operando *acção* na lista de mensagens, por $f_z(c,C)$ fica a saber-se se um classificador gera ou não uma saída para o ambiente do sistema. C é o conjunto dos classificadores do SC e $c \in C$. Por outro lado a cardinalidade do conjunto $\{c \in C \mid f_z(c,C) = verdadeiro\}$ ou é 0 (zero) ou 1 (um), uma vez que a saída do SC ou é uma mensagem ou éter (vazio). Ora isto implica que pelo menos um classificador pode gerar uma saída para o ambiente.

Está-se então em condições de definir um dos ramos da função *F*, a parte que define a nova lista de mensagens, F_B, dada na forma:

$$F_B(C,B) = \begin{cases} \left\{ \left\langle F_o(b, COND_c, ACT_c), c \right\rangle \middle| c \in C \wedge F_m(b,c) = verdadeiro \wedge f_g(c,C) \right\} & se\ F_g(C,B) \neq 0 \\ 0 & se\ F_g(C,B) = 0 \end{cases}$$

em que ACT_c é a parte *acção* do classificador c, e $COND_c$ é a parte *condição* do mesmo classificador. O segundo ramo de F, a parte que dá a saída do sistema, a que se chamará F_g, é dada na forma:

$$Fg\ (\mathrm{C},\mathrm{B}) = \begin{cases} \{ACT_c \mid c \in C \wedge f_z(c,C)\} & se\ \exists_c \in C \\ 0 & \text{em outros casos} \end{cases}$$

onde c denota o classificador activo que apresenta o maior lanço (valor de licitação).

A função F pode agora ser calculada combinando F_B e F_g. A saída da primeira é usada como a saída B' de F e a última é utilizada como a saída g de F. Embora à primeira vista possa parecer que a lista de mensagens não é utilizada pela função F, é com recurso a esta que se determina se um classificador está ou não activado.

Definição 13 (Cálculo do Valor do Lanço).

A partir da entalpia de um classificador pode-se calcular o valor do lanço por parte deste. Este é o elemento que será usado para determinar quais as mensagens que deverão ou não pertencer à nova lista de mensagens. O lanço é calculado considerando a entalpia e a especificidade do classificador, com base na expressão:

$$L_c = b \cdot SP_c \cdot ST_c$$

onde L_c denota o lanço do classificador c, $b \in \mathbb{R}^+$ é uma constante $(0 < b \leq 1)$ que tipifica a fracção da entalpia do classificador que é usada na operação de licitação (normalmente um valor pequeno), SP_c a especificidade do classificador c, e ST_c a entalpia do classificador c.

Definição 14 (As Funções f_g e f_z).

As funções f_g e f_z definidas em epígrafe, a partir das quais se decide se um classificador deve ou não colocar o operando *acção* na lista de mensagens, ou se um classificador gera ou não uma saída para o ambiente do sistema, são passíveis de

diferentes interpretações (embora haja que ter em conta certas condicionantes, como a cardinalidade de f_z como já se referiu). Para que um SC seja eficiente, este não poderá ter um comportamento determinístico (caso contrário tende-se a favorecer o *status quo*), ao mesmo tempo que f_g e f_z, para os classificadores de maior entalpia, são calculadas para a constante lógica *verdadeiro* (estas funções têm que atender a todos os classificadores que foram activados pelo sistema).

Uma possível definição para f_g e f_z é dada por Goldberg [Goldberg 1989], que passa por se somar aos valores dos lanços de todos os classificadores activos no sistema, um valor de ruído dado por uma distribuição normal:

$$N(\overline{m}, s) = BM \cdot s + \overline{m}$$

em que $\overline{m}$ corresponde à média (neste caso zero), s ao desvio padrão (um parâmetro do SC), sendo *BM* gerado pelo método de Box-Muller [Pike 1980], que a seguir se caracteriza.

Algoritmo 1 (Método de Box-Muller).

```
func BM º
    seja t um número real
    seja rndx1 um número real
    seja rndflag um parâmetro global, um valor de verdade que se altera
        sempre que o algoritmo é executado
    seja rndx2 um parâmetro global tomado do conjunto dos números
        reais
    seja rnd um número real gerado aleatoriamente (0£rnd<1)
    se rndflag então
        rndx1 ¬ √(−2 · ln rnd)
        t ¬ 6.28318530 72 · rnd
        rndx2 ¬ rndx1 · sen(t)
        rndflag ¬ falso
        retornar rndx1 · cos(t)
    senão
        rndflag ¬ verdadeiro
        retornar rndx2
    fimse
```

em que *ln* e *sen(t)* denotam, respectivamente as funções logaritmo e seno. O passo a seguir passa por se atribuir o valor de verdade *verdadeiro* aos classificadores activos de maior lanço no caso de f_g, e ao melhor classificador activo, no caso de f_z. Para escolher os classificadores vencedores numa operação de licitação calcula-se um lanço efectivo (LE_c), para cada classificador cuja parte *condição* unifique com a mensagem, sendo este dado pela soma do valor lanço L_c (componente determinística) a que se adiciona um certo ruído (componente não determínistica), ou seja:

$$LE_c = L_c + N(0, s)$$

onde o ruído N é uma função do desvio padrão do ruído considerado, ou seja de s. Após as operações de licitação e de selecção de uma mensagem, os classificadores emissores deverão efectuar um pagamento aos classificadores responsáveis pelo envio das mensagens que activaram os actuais (classificadores) vencedores. Estes classificadores pagam os seus lanços (os valores de L_c, não os valores de LE_c) à compensação, onde o pagamento é dividido por todos os classificadores responsáveis pelo envio das mensagens que seleccionaram os actuais (classificadores) vencedores.

Definição 15 (Procedimento para o Subsistema de Regras e Mensagens).

A cada lista de mensagens e conjunto de classificadores pode-se aplicar a função *F(C, B)*, e obter uma nova lista de mensagens. A esta nova lista de mensagens pode-se aplicar de novo a função *F(C, B)*. Obtém-se assim um procedimento de recorrência na forma $B_{t+1} = F_B(C, B_t)$ (F_B é a parte da mensagem da função F, t é o ciclo do SC). As condições iniciais (i.e., B_0) e um critério de paragem do processo iterativo, são dadas a seguir.

A caracterização de B_0 é simples; i.e., gera-se uma mensagem i, a que se chama a entrada do sistema. Faz-se então $B_0=\{i\}$ (i.e., um conjunto com um único elemento i). O critério de paragem toma a forma:

$$F_g(C, B_t) = 0 \text{ Ú } t>T$$

onde F_g é o contradomínio da função F e T é um parâmetro do sistema que estipula o número máximo de iterações a serem realizadas.

Algoritmo 2 (Subsistema de Regras e Mensagens).

```
proc RM º
seja k a cardinalidade da lista de mensagens

iniciar o sistema
criar aleatoriamente um conjunto inicial de classificadores
enquanto não condição de paragem fazer
        detecção        ler do ambiente as mensagens através dos detectores,
                        descodificá-las e acrescentá-las à lista de mensagens
        unificação      colocar no estado activo todos os classificadores do
                        sistema cujo operando condição unifique com uma
                        das mensagens da lista de mensagens. Remover as
                        mensagens presentes  na lista de mensagens
        licitação       se o número de classificadores activos for £ k
                          então  colocar as suas mensagens na lista de
                                 mensagens
                          senão  invocar a componente de licitação que
                                 selecciona os k classificadores vencedores e
                                 coloca os seus operandos acção na lista de
                                 mensagens
        Seleccionar na lista de mensagens aquelas que se destinam ao
                ambiente do sistema
        resolução de conflitos  seleccionar de entre estas mensagens aquela
                                que deverá ser enviada ao ambiente do sistema
        repercussão     codificar a mensagem para o ambiente do sistema e
                        receber a recompensa desse mesmo ambiente
        colocar o estado de todos os classificadores do sistema na situação
                de não activos
fimenq
```

3 Subsistema de Distribuição de Créditos

Nos SC procura-se catalogar ou avaliar os classificadores, de acordo com o seu impacto no ambiente do sistema. Existem vários caminhos para o fazer, no entanto o processo mais comum recorre ao algoritmo "The Bucket Brigade Algorithm" [Holland 1985] [Goldberg 1989], um algoritmo de distribuição de créditos cuja principal função é a de seriar os classificadores segundo o seu grau de utilidade para os SC. Tudo se passa como se se tivesse (e isto recorrendo ao uso de algumas metáforas) uma economia de informação, onde o direito de negociar é comprado e

vendido pelos classificadores, que actuam em cascata (ou cadeia), cadeia esta que vai desde o fabricante (o ambiente), ao consumidor da informação (os repercussores).

Este modelo económico tem duas vertentes cruciais: os processos de licitação e de compensação. Quando a parte *condição* dos classificadores unifica com uma dada mensagem estes não colocam a parte *acção* directamente na lista de mensagens. O facto de um classificador passar por uma operação de unificação apenas o qualifica a participar numa operação de licitação (diz-se que o classificador foi activado). Para participar na operação de licitação, cada classificador mantém um registo do potencial que representa para o SC, também denominado de peso ou entalpia (*ST*). Ora cada classificador que executa com êxito uma operação de unificação faz um lanço *L* proporcional ao seu peso ou entalpia. Assim quanto maior for o peso de um classificador maior é o seu lanço. Do mesmo modo, quanto mais específica for a componente *condição* maior é o lanço. Deste modo os classificadores com maior entalpia posicionam-se como os primeiros no processo de licitação.

A operação de licitação permite seleccionar os classificadores potencialmente mais adequados a colocarem mensagens na lista de mensagens. O processo de distribuição de créditos não se fica, porém, por aqui. Uma vez seleccionado e activado, um classificador passa pela bolsa de compensação, onde procede ao pagamento, pelo valor do lanço que protagonizou. Este pagamento é, de alguma forma, dividido pelos classificadores que participaram na operação de unificação. A divisão do pagamento (o saldar da dívida) pelos classificadores que participam na operação de unificação ajuda a assegurar a formação de subpopulações de um tamanho apropriado, o que garante a diversidade genética da população de classificadores. Num sistema de aprendizagem com regras deste tipo não se pode procurar uma que seja original. Deve-se, em vez disso, procurar um conjunto de classificadores que por si cubram certas áreas do problema em equação e que, no futuro, permitam um bom desempenho do sistema de aprendizagem.

Definição 16 (Classificadores Credores e Devedores).
Por forma a determinar quais os classificadores que devem receber (i.e., os credores), e quais os que devem pagar (i.e., os de vedores), são definidos:

- o conjunto P_t dos classificadores que colocaram uma mensagem na lista de mensagens, ou geraram uma saída para o sistema no tempo *t*, ou seja:

$$P_t=\{c \mid c \in C \wedge c \text{ está activo no ciclo } t \wedge (f_g(c,C) \vee f_z(c,C))\}$$

- o conjunto $Q_{c,t}$, dos classificadores que colocaram mensagens na lista de mensagens, e que tornaram o classificador *c* activo no ciclo *t*, ou seja:

$$Q_{c,t}=\{x \mid x \in P_{t-1} \wedge \exists b \in B: (b=x: (F_m(b,COND_c) = \text{verdadeiro}))\}$$

onde *x*, o classificador vencedor, é escolhido do conjunto dos classificadores cuja parte *condição* unificou com as mensagens contidas na lista de mensagens.

Definição 17 (A Forma e os Valores dos Pagamentos).
Com estes conjuntos P_t, e $Q_{c,t}$, referidos em epígrafe, é possível calcular as alterações à entalpia de cada classificador; i.e., os pagamentos que devem ser realizados entre classificadores. Os classificadores que não estiveram activos no tempo *t-1*, pagam uma certa taxa a cada ciclo do sistema, chamada de taxa de permanência ou de vida no sistema, ou seja:

$$\forall c \in C(c \in P_{t-1}): (ST'_{c,t}= ST_{c,t-1} - t_{vida}\, ST_{c,t-1})$$

em que $t_{vida} \in \mathbf{R}^+$, $0 \leq t_{vida} < 1$, é um parâmetro do sistema que fixa a taxa a ser paga pelos classificadores pelo simples facto de existirem no sistema (i.e., evita-se a presença de classificadores parasitas no sistema). Aos classificadores activos, e que portanto participaram no processo de licitação, para além da taxa de permanência, é cobrada uma taxa adicional, chamada de taxa de licitação, $t_{lanço}$, ou seja:

$$\forall c \in C(c \in P_{t-1}): (ST'_{c,t}= ST_{c,t-1} - (t_{lanço}+ t_{vida})ST_{c,t-1})$$

ou

$$\forall c \in C(c \in P_{t-1}): (ST'_{c,t}= ST_{c,t-1} - t_{vida}ST_{c,t-1} - t_{lanço}L_{c,t-1})$$

conforme o cálculo de ST' que é seleccionado, onde $t_{lanço} \in \mathbb{R}^+$, e $0 \leq t_{lanço} < 1$. Como consequência deste último enunciado são feitas outras duas modificações aos valores entâlpicos dos classificadores, ou seja:

$$\forall c \in P_t: (ST''_{c,t} = ST'_{c,t} - L_c)$$

em que L_c denota o lanço oferecido pelo classificador c na tentativa de ser activado. Isto faz com que os classificadores do sistema que colocam mensagens na lista de mensagens paguem por isso. Obviamente que há que garantir que $(t_{lanço}+t_{vida})ST_{c,t-1} + L_c < ST_{c,t-1}$, em que $ST_c>0$. A segunda modificação corresponde a:

$$\forall c \in P_t: (\forall x \in Q_{c,t}: (ST_{x,t} = ST'_{x,t} + \frac{1}{|Q_{c,t}|} \cdot L_c))$$

i.e., os classificadores que ajudaram terceiros a ficar activos são compensados, ou seja, compensaram-se todos os classificadores que ajudaram a activar $c \in Q_{c,t}$ com uma distribuição equitativa do lanço de c, L_c. Para os restantes classificadores, há uma variação negativa da entalpia para o tempo t $(ST_{c,t})$, variação esta que a iguala a $ST'_{c,t}$ se o classificador não esteve activo, e a $ST''_{c,t}$ se o classificador esteve activo.

Definição 18 (Entradas e Saídas).

Esta definição do algoritmo de distribuição de créditos adequa-se aos passos do SC que não têm nada a ver com entradas nem saídas. Mas o que acontece quanto às entradas e saídas?

Na fase de entrada, coloca-se apenas uma mensagem na lista de mensagens, $B_0=i$. Os conjuntos P_{-1} (i.e., o conjunto dos classificadores activos no sistema antes do primeiro ciclo) e $Q_{c,0}$ (para todos os c) são conjuntos vazios, consequentemente não existem classificadores a compensar. O valor da entalpia dos classificadores que se tornaram activos foi decrementada, o que faz com que a entalpia do sistema decresça. É possível ler-se isto como um pagamento feito pelo sistema ao ambiente (i.e., ao mundo exterior).

Para a saída do sistema o procedimento é análogo. O classificador seleccionado para gerar uma saída é recompensado pelo ambiente com um valor entâlpico r, que é

função da utilidade da resposta fornecida pelo sistema para uma dada entrada. Este valor é adicionado à sua carga entâlpica, $ST'_c = ST_c + r$, onde ST_c é a entalpia do classificador que gerou a saída. Recompensas negativas não são permitidas no sistema, uma vez que, por definição da utilidade de um classificador para o sistema, a esta não pode ser associado um valor negativo (o que acabaria por deturpar o processo de licitação). É agora possível, se $ST_{c,0}$ for a entalpia do classificador c no instante $t=0$, calcular o valor desse mesmo parâmetro no instante t, através do procedimento recorrente:

$$ST_{c,t} = (1-(t_{lanço} + t_{vida})) \cdot ST_{c,0} + \sum_{i=0}^{t-1} \left(R_{c,i} \cdot (1-(t_{lanço} + t_{vida}))^{t-i-1} \right)$$

em que $R_{c,i}$ denota o valor total das receitas com que o classificador c foi contemplado no instante i. Se o processo continuar indefinidamente, com uma recompensa constante $R_{c,i}=R_{c,est}$, caminha-se para uma situação de estado estacionário, em que $ST_{c,t+1} = ST_{c,t} = ST_{c,est}$, e no qual a carga entâlpica estabilizada de um classificador é dada na forma:

$$ST_{c,est} = \frac{R_{c,est}}{(t_{lanço} + t_{vida})}$$

De igual modo, o valor estabilizado do lanço de um classificador c, $L_{c,est}$, pode ser calculado na forma:

$$L_{c,est} = \frac{t_{lanço}}{(t_{lanço} + t_{vida})} \cdot R_{c,est}$$

em que $t_{vida} << t_{lanço}$, e o valor da licitação, $L_{c,est}$, aproxima-se do valor da recompensa estabilizada $R_{c,est}$, $L_{c,est} \approx R_{c,est}$. Escolhendo os coeficientes t_{vida} e $t_{lanço}$ de uma forma apropriada, pode-se encorajar a formação do que é denominado de *hierarquias por defeito*. Numa *hierarquia por defeito* os classificadores de natureza mais genérica (i.e., aqueles cuja parte *condição* tem um menor número de símbolos diferentes de "#") tendem a sobrepor-se aos classificadores mais específicos.

As *hierarquias por defeito* têm a vantagem de permitirem não só a utilização de um menor número de classificadores para a resolução de um mesmo problema, mas também de favorecerem o aparecimento de um maior número de soluções. Ora este é um aspecto interessante, visto que um menor número de classificadores traz ganhos para o desempenho do sistema. Por outro lado, a presença de diferentes conjuntos de classificadores com um bom desempenho leva ao aparecimento de melhores soluções para o problema (alargamento do espaço de soluções).

O meio mais simples de encorajar a formação de *hierarquias por defeito* passa pela escolha de valores adequados para as operações licitação dos classificadores, de tal modo que valor do lanço de um classificador *c* seja proporcional ao produto do seu valor entâlpico pela medida da sua especificidade, nos termos:

$$L_{c,t} = b \cdot (lic_1 + lic_2 \cdot SP) \cdot ST_{c,t}$$

onde lic_1 e lic_2 são parâmetros do SC que, dependendo dos valores que possam tomar, possibilitam a exploração de diferentes estratégias de licitação. Do mesmo modo o valor do lanço efectivo pode ser calculado por:

$$LE_{c,t} = b \cdot (elic_1 + elic_2 \cdot SP_c) \cdot ST_{c,t} + N(0, s)$$

onde $elic_1$ e $elic_2$ são parâmetros do SC, definidos em termos similares a lic_1 e lic_2.

Nestas condições, o valor entâlpico, $ST_{c,est}$, e o valor do lanço, $L_{c,est}$, do classificador *c* quando é atingido o estado estacionário, podem ser calculados na seguinte forma:

$$ST_{c,est} = \frac{R_{c,est}}{tl_{lanço} \cdot (lic_1 + lic_2 \cdot SP_c) + t_{vida}} \text{; e}$$

$$L_{c,est} = \frac{t_{lanço} \cdot (lic_1 + lic_2 \cdot SP_c)}{t_{lanço} \cdot (lic_1 + lic_2 \cdot SP_c) + t_{vida}} \cdot R_{c,est}$$

Se se considerar que existe uma *hierarquia por defeito*, então a presença de uma excepção tende a esconder os erros das regras de pouca especificidade.

Dado que o subsistema de distribuição de créditos está intimamente ligado ao subsistema de regras e mensagens, apresenta-se a seguir um algoritmo que faz a consolidação dos dois subsistemas (Figura 2).

Algoritmo 3 (Subsistema de Regras e Mensagens e Subsistema de Distribuição de Créditos)

proc ***SC***º
seja *C o conjunto dos classificadores do SC*
seja *c um classificador de C*
seja *n a cardinalidade do conjunto de classificadores C*
seja *B a lista de mensagens do SC*
seja $ST_{c,t}$ *o valor entâlpico de classificador c no ciclo t*
seja $L_{c,t}$ *o lanço de um classificador c no ciclo t*
seja $LE_{c,t}$ *o lanço efectivo de um classificador c no ciclo t*
seja *k o tamanho da lista de mensagens B*
seja *r a recompensa por parte do ambiente*
seja *t o ciclo do SC*

iniciar o sistema
criar aleatoriamente um conjunto C de n classificadores
atribuir o mesmo valor (pode ser seleccionado aleatoriamente) à entalpia,
ST_c, de todos os classificadores c Î C
t ¬ 1
enquanto *não* ***condição de paragem*** ***fazer***
detecção *ler do ambiente do sistema as mensagens através dos detectores, descodificá-las e acrescentá-las à lista de mensagens B*
unificação *passar a activo o estado de todos os classificadores cuja parte condição unifique com uma das mensagens da lista de mensagens; remover as mensagens da lista de mensagens; para cada classificador c activo calcular $L_{c,t}$ e $LE_{c,t}$*
licitação ***se*** *o número de classificadores activos for £ k*
então *colocar as suas mensagens na lista de mensagens*
senão *invocar o procedimento de licitação para seleccionar os k classificadores vencedores (os de maior $LE_{c,t}$) e colocar as suas mensagens na lista de mensagens*
distribuição de créditos *cada um dos k classificadores vencedores no tempo t paga o valor $L_{c,t}$ aos classificadores vencedores no tempo t-1*
colecta de taxas *cada classificador activo paga uma taxa de licitação; todos os classificadores pagam uma*

taxa de sobrevivência
resolução de conflitos *seleccionar de entre os classificadores vencedores aquele que deverá enviar a mensagem ao ambiente do sistema*
repercussão *codificar a mensagem para o ambiente do sistema e receber a recompensa r do ambiente e entregá-la ao classificador seleccionado no ponto anterior; colocar o estado de todos os classificadores em não activo; incrementar t (t ¬ t + 1)*

fimenq

4 Algoritmos Genéticos

O subsistema de distribuição de créditos corresponde a um procedimento para avaliar regras e decidir entre as alternativas em oposição. Falta, porém, neste ponto, uma forma de injectar novas regras, possivelmente melhores, no sistema. Este é precisamente o motivo pelo qual o AG é considerado. Utilizando um AG, novas regras podem ser criadas através de um processo tripartido de reprodução, cruzamento e mutação. Estas regras são então colocadas na população de regras e processadas pelas operações de licitação e de compensação. Neste trabalho dá-se particular ênfase não à problemática da substituição de toda uma população por uma nova população, mas atende-se a quem é que substitui quem. Vai-se assim olhar às diferenças entre o AG clássico e aquele de que se necessita para o SC.

4.1 Introdução

Os AG configuram processos adaptativos de procura num espaço de soluções, por aplicação de operadores modelados de acordo com o conceito de herança inerente à teoria de Darwin da evolução das espécies. Pertencem à classe de algoritmos probabilísticos, mas distinguem-se pelo método de procura que utilizam e pelo tratamento muito específico dos óptimos locais. Têm vindo a ser aplicados, essencialmente, em problemas que envolvem a melhoria de soluções (Anexo H).

Os princípios básicos dos AG foram definidos por Holland na década de setenta [Holland 1975], embora hoje o termo se aplique ao conjunto de métodos que, usando operadores de selecção e recombinação no espaço de soluções, possam contribuir para a resolução de um problema, por optimizações sucessivas de possíveis soluções [Whitley 1994]. O algoritmo originalmente proposto por Holland é actualmente designado por AG canónico ou AG simples [Goldberg 1989].

O principal objectivo de Holland, ao contrário daquele que está subjacente às estratégias evolutivas e à programação evolutiva, não foi o de resolver problemas específicos, mas sim de estudar formalmente o fenómeno da adaptação tal e qual ele ocorre na natureza e incorporar os mecanismos da adaptação natural nos sistemas de computação. No seu livro *Adaptation in Natural and Artificial Systems* [Holland 1975, 1995], apresentou o AG quer como uma abstracção na evolução biológica quer como um enquadramento teórico para a sua adaptação ao meio ambiente.

Na natureza, qualquer indivíduo é obrigado a lutar para sobreviver, competindo com os outros por recursos essenciais, como alimento ou abrigo. Além disso, em todas as espécies sexuadas, os *especimen* de cada sexo competem entre si, também com o objectivo de atrair um elemento do sexo oposto para efeitos de reprodução. Os indivíduos mais capazes irão, com uma maior probabilidade, deixar uma descendência mais numerosa, levando a que os seus genes se tornem predominantes na população.

Os AG fazem uma analogia directa com este processo, trabalhando com uma população de indivíduos, em que cada um representa uma possível solução para um dado problema. A cada indivíduo é atribuído um *valor de utilidade*[5], a entalpia, o que na natureza corresponde à sua aptidão para a sobrevivência e reprodução. No caso dos AG, este valor representa a qualidade da solução codificada no genótipo, ou seja, sendo obtido por aplicação da função objectivo à solução.
Os indivíduos com melhor entalpia, terão mais hipóteses de serem seleccionados para o processo de reprodução e, desta forma, transmitir os seus genes para as gerações vindouras. Favorecer a reprodução dos indivíduos mais capazes, ao longo de um número elevado de gerações conduz à exploração das zonas mais promissoras do

[5] Designado por *fitness* ou *strength* na literatura anglo-saxonica.

espaço de procura, ou seja, as constituídas por indivíduos com entalpias (médias) mais elevadas. Nos AG este processo de evolução conduzirá, ou assim se espera, a soluções de qualidade.

4.2 Metáfora Biológica

Todos os organismos vivos são formados por células, e cada célula contém o mesmo conjunto de um ou mais cromossomas (cadeias de ADN[6]) que são o retrato do organismo. Conceptualmente um cromossoma pode ser divido em genes (blocos funcionais do ADN), codificando cada um uma proteína[7] particular. É licito pensar que um gene codifica uma característica do organismo como por exemplo a cor dos olhos ou a cor do cabelo etc. A cada valor possível de um gene (e.g., castanho, azul, verde) chama-se alelo.

Muitos organismos possuem múltiplos cromossomas numa célula. O genótipo corresponde ao conjunto de genes contido no material genético de um organismo (conjunto de todos os cromossomas - o genoma). O genótipo de um organismo vivo está na origem do seu fenótipo, ou seja, as suas características físicas e mentais, como a cor dos olhos, altura, tamanho do cérebro e inteligência.

Os organismos cujos cromossomas são agrupados em pares são chamados diplóides em oposição aos haplóides. Na natureza, a maior parte das espécies sexuadas, incluindo os seres humanos, são diplóides e no caso destes últimos cada célula contém 23 pares de cromossomas. No processo da reprodução sexual dá-se lugar à recombinação (cruzamento). No caso da reprodução diplóide, em cada progenitor os genes são trocados dentro dos pares de cromossomas para formar os gámetas (um único cromossoma), e depois entre os gámetas dos progenitores para criar um conjunto completo de pares de cromossomas. No caso da reprodução haplóide, os genes são trocados entre os cromossomas dos progenitores. No processo de troca ocorrem por vezes erros de cópia, a mutação, na qual alguns alelos são alterados. O

[6] Ácido Desoxirribonucleico.
[7] Composto orgânico de elevadíssimo peso molecular, constituído por carbono, oxigénio, hidrogénio, azoto, e, por vezes também enxofre e fósforo.

valor de cada organismo é definido, normalmente, como a probabilidade do organismo sobreviver e reproduzir-se (viabilidade), ou como uma função do número de descendentes que o organismo tem (fertilidade).

Os termos acima apresentados oriundos da ciência biológica podem, após as devidas adaptações, ser transportados para a ciência computacional. Nesse sentido, o termo cromossoma refere uma solução candidata para um problema, codificada através de uma cadeia de *bits*. Os genes correspondem a *bits* ou a grupos de *bits* adjacentes que codificam um elemento particular da solução. O alelo numa cadeia de *bits* pode ser um 0 ou um 1, num alfabeto de maior cardinalidade (e.g., sistema decimal) poderá assumir cada um dos valores possíveis nesse alfabeto. O cruzamento, por seu turno, consiste na troca de elementos entre os cromossomas de progenitores haplóides. A mutação é implementada através da inversão de *bits* em determinadas posições escolhidas aleatoriamente nos cromossomas (para alfabetos de maior cardinalidade o símbolo é trocado por um outro seleccionado aleatoriamente dentro do alfabeto). O genótipo de um indivíduo no AG corresponde ao arranjo dos *bits* no interior do cromossoma. Nos AG não existe paralelo para fenótipo, embora em algumas aplicações dos AG mais recentes, nomeadamente nas redes neuronais, seja possível reconhecer o genótipo na codificação das soluções de um dado problema, e o fenótipo na topologia da rede neuronal em si.

4.3 Estrutura dos Algoritmos Genéticos

Geralmente, um AG executa uma procura multi-direccional, e promove a formação de informação e troca entre essas direcções. Isso é feito mantendo uma população de soluções propostas (cromossomas) para um dado problema. Cada solução é representada num alfabeto fixado (normalmente binário) com um significado estabelecido. A população segue uma evolução simulada: as soluções relativamente "boas" produzem descendentes, que subsequentemente substituem as "piores". A estimativa da qualidade de uma solução é baseada numa função de avaliação que é característica do universo de discurso (ambiente).

Algoritmo 4 (Algoritmo Genético)

```
proc AGº
    seja POP(t) a população de cromossomas no tempo t
    seja t o tempo actual
    t ¬ 0
    iniciar POP(t)
    avaliar POP(t)
    enquanto não condição paragem fazer
        gerar POP'(t+1) a partir de POP(t) por aplicação do
            operador de selecção
        gerar POP''(t+1) a partir de POP'(t+1) por aplicação do
            operador de cruzamento
        gerar POP'''(t+1) a partir de POP''(t+1) por aplicação do
            operador de mutação
        POP(t+1) ¬ P'''(t+1)
        avaliar POP(t+1)
        t ¬ t+1
    fimenq
```

O AG inicia-se com a geração de um conjunto de indivíduos (cromossomas), ou seja, possíveis soluções para o problema em equação (população inicial) e a sua avaliação por aplicação de uma função de avaliação cujo contradomínio corresponde ao valor intrínseco de cada indivíduo para obtenção de uma solução (utilidade da solução). Em cada novo ciclo do AG, chamado ciclo reprodutivo, parte-se da população actual *POP(t)*, sendo gerada uma nova instanciação da população *POP(t+1)* através de um processo tripartido que corresponde à aplicação dos operadores genéticos de selecção, cruzamento e mutação. A condição de saída do AG define-se, em princípio, com base num número máximo de iterações podendo, no entanto, incluir medidas de convergência da solução.

A caracterização de um AG na procura de uma solução para um problema em particular envolve a descrição de um conjunto de componentes, em que há a salientar:

- uma representação com base genética das soluções potenciais para o problema, que também definem o espaço de soluções do problema; e
- o desenvolvimento de procedimentos para codificação das soluções em sequências binárias (cromossomas) e sua descodificação; e

- um método de gerar a população inicial de soluções potenciais; e
- uma função de avaliação que tipifique o universo de discurso (ambiente), e classifique as soluções em termos da sua utilidade (valor) ou adaptabilidade para a resolução do problema; e
- operadores genéticos que alterem a composição das soluções (cromossomas) durante a operação de recombinação; e
- valores para os vários parâmetros que o algoritmo usa (e.g., cardinalidade da população, probabilidades associadas aos operadores genéticos, condição(ões) de paragem).

4.4 Operadores Genéticos

A forma mais simples de um AG conta com três tipos de operadores: o de selecção, o de cruzamento, e o de mutação.

4.4.1 Operador de Selecção

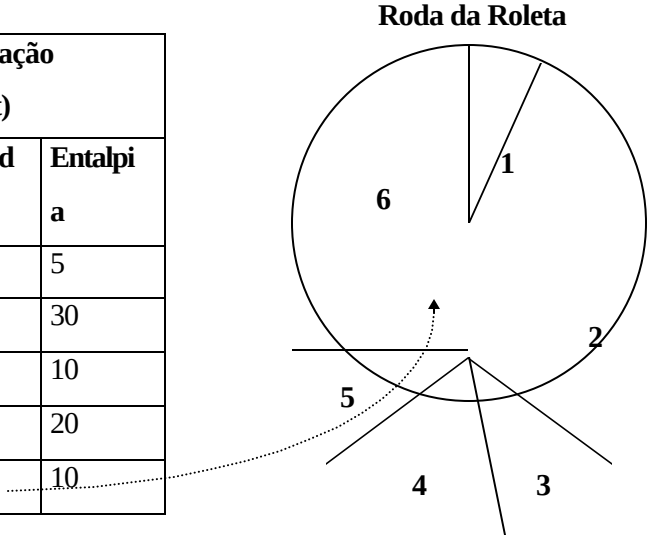
Um dos métodos, aquele que foi adoptado no contexto dos SC, para a selecção dos indivíduos que se devem reproduzir, baseia-se no método da roleta [Goldberg 1989]. A ideia é imitar uma roleta cuja roda é compartimentada de acordo com o valor entâlpico de cada indivíduo, ou seja, cada compartimento da roleta representa um indivíduo e possui uma área proporcional à sua utilidade para a resolução do problema (Figura 7). O objectivo é simular o atirar da bola sobre a roda, e como se pode facilmente perceber, a probabilidade de um indivíduo com um maior valor entâlpico (e.g., o indivíduo 6) ser seleccionado é maior que a de um indivíduo com um menor valor entâlpico (e.g., o indivíduo 1), uma vez que o primeiro teve o privilégio de ser associado a um compartimento com maior área, e claro, existem mais possibilidades de a bola lá ir parar. Isto não invalida que indivíduos de menor valor entâlpico não venham a ser seleccionados.
Este método pode ser descrito pelo seguinte algoritmo:

Algoritmo 5 (Selecção)

```
func SELº
    seja n o tamanho da população de cromossomas
    seja Sum a totalidade dos valores entâlpicos dos n elementos da
         população
    seja R Î [0,Sum] um número gerado aleatoriamente
    seja m a ordem do elemento seleccionado (primeiro elemento da
         população de cromossomas cujo valor entâlpico adicionado
         ao dos restantes elementos da população precedente,
         seja maior ou igual a R)
    seja random uma função geradora de números aleatórios
    seja ST_i a entalpia de um cromossoma i
    seja x um número real
    seja m um número inteiro
    Sum ¬ 0
    para i ¬ 1 até n fazer
         Sum ¬ Sum+ST_i
    fimpara
    R ¬ random(0, Sum)
    m ¬ 1
    x ¬ 0
    enquanto x < R e m <n fazer
         x ¬ x+ST_m
         m ¬ m+1
    fimenq
    retornar m
```

Como resultado deste procedimento tem-se a devolução, de forma aleatória, de um cromossoma.

População POP(t)	
Indivíduo	**Entalpia**
1	5
2	30
3	10
4	20
5	10



6	25.0

Figura 7 Selecção de indivíduos através do método da roleta.

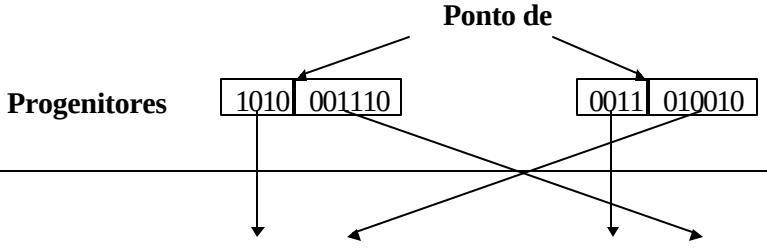
4.4.2 Operador de Cruzamento

O operador de cruzamento permuta aleatoriamente peças genéticas entre dois cromossomas na tentativa de propagar soluções parciais, e pode ser definido através de uma função, a função *Cruz*, que tem como entradas dois elementos da população, um conjunto de parâmetros de controlo, e dá como retorno dois novos indivíduos, ou seja:

Cruz: Indivíduo x Indivíduo x Parâmetros Controlo → Indivíduo x Indivíduo

A operação de cruzamento não tem necessariamente que ser aplicada à totalidade dos indivíduos seleccionados para a recombinação, sendo normalmente definida uma medida da possibilidade de aplicação do operador a um dado par de cromossomas, designada por probabilidade de cruzamento p_c. Este factor é o mais importante do conjunto de parâmetros de controlo, e toma valores normalmente no intervalo [0.6, 1.0].

Quando se opta por uma representação binária o operador mais comummente utilizado é o cruzamento de ponto único. Este operador permite seleccionar, aleatoriamente, uma posição de corte nos cromossomas dos progenitores, criando, assim, quatro subsequências, duas iniciais e duas finais. Estas são ligadas de modo a que as duas subsequências, em cada um dos dois descendentes, provenham de um progenitor diferente, tal como é descrito na Figura 8.



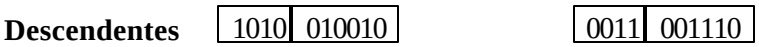


Figura 8 Cruzamento de ponto único.

O funcionamento deste operador é dado pelo algoritmo que a seguir se explicita, tomando como exemplo cromossomas definidos no alfabeto binário.

Algoritmo 6 (Cruzamento de Ponto Único).

proc *CRUZº*

seja *pro1 e pro2 os cromossomas progenitores a cruzar de tamanho l*
seja $pro1_i$ *o bit de ordem i do cromossoma pro1*
seja $pro2_i$ *o bit de ordem i do cromossoma pro2*
seja *des1 e des2 os cromossomas descendentes*
seja *l o tamanho de um cromossoma (número de bits)*
seja *cruz uma medida da ocorrência ou não da operação de cruzamento*
seja p_c *a probabilidade de cruzamento*
seja *pcruz o ponto de cruzamento*
seja *random uma função geradora de número aleatórios* $\hat{I}$ N^+
cruz ¬ *falso*
se $p_c = 1.0$ ***então***
cruz ¬ *verdadeiro*
senão
cruz ¬ *(random £* p_c*)*
fimse
se *cruz* ***então***
pcruz ¬ *random(1,l)*
des1 ¬ $pro1_1$,..., $pro1_{pcruz}$, $pro2_{pcruz+1}$,..., $pro2_l$
des2 ¬ $pro2_1$,..., $pro2_{pcruz}$, $pro1_{pcruz+1}$,..., $pro1_l$
fimse

Outros algoritmos têm sido desenvolvidos envolvendo mais do que um ponto de corte. Por exemplo, no cruzamento duplo [De Jong 1975] [Booker 1982] são seleccionados dois pontos de cruzamento o que possibilita a diminuição dos fenómenos de ruptura do cruzamento simples. A ruptura é um fenómeno que consiste em, durante o processo de recombinação genética, certas informações dos progenitores não serem passadas aos descendentes. Na prática pode dizer-se que a redução deste efeito de

disrupção produz um aumento da eficiência do processo de cruzamento. Este fenómeno pode ser reduzido se se considerarem dois pontos de cruzamento seleccionados aleatoriamente, conforme é ilustrado na Figura 9.

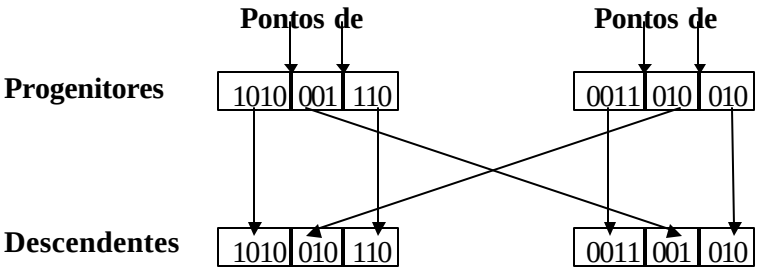


Figura 9 Cruzamento de dois pontos.

Um outro método utilizado tem sido também o designado por cruzamento uniforme. Neste caso, o cruzamento depende de uma máscara binária, gerada aleatoriamente, com um comprimento igual ao dos cromossomas dos progenitores. O comportamento deste operador é dado na Figura 10 e pode ser descrito da seguinte forma: nas posições onde a máscara contém o valor binário 1 (um) o descendente toma o valor do gene do primeiro progenitor nessa posição, senão toma o valor do gene do segundo cromossoma. O segundo descendente é construído de forma inversa.

Máscara	1	0	0	1	0	1	1	1	0	0
Progenitor 1	1	0	1	0	0	0	1	1	1	0
Progenitor 2	0	1	0	1	0	1	0	0	1	1
Descendente 1	1	1	0	0	0	0	1	1	1	1
Descendente 2	0	0	1	1	0	1	0	0	1	0

Figura 10 Cruzamento uniforme.

Uma vez que estes operadores são definidos em relação a peças sintácticas, na representação dada (onde cada cromossoma é visto como uma sequência de símbolos de um alfabeto), a procura tem propriedades independentes do domínio do problema. Todavia, a aplicabilidade de um AG a um problema em particular depende da ênfase que a representação atribui à semântica das peças de informação (chamada construção

de blocos) a ser usada pelo operador de cruzamento, em conjunção com a função de avaliação na condução da procura; a aplicabilidade do AG decresce se os operadores forem chamados a manipular estruturas sintácticas de baixo nível.

A argumentação quanto aos méritos de cada um dos operadores de cruzamento, incluindo aqueles aqui não referidos, não produziu ainda resultados conclusivos [Beasley et al. 1993], embora seja geralmente aceite que a utilização de dois pontos de corte se prefigura como uma opção mais vantajosa em relação ao cruzamento de ponto único.

4.4.3 Operador de Mutação

A mutação é um operador unário que pode ser definido, em termos de uma função, a função de mutação, que toma como argumentos um indivíduo e um conjunto de parâmetros de controlo, tendo como retorno um novo indivíduo:

$$\text{Mut: Indivíduo x Parâmetros Controlo} \rightarrow \text{Indivíduo}$$

O principal parâmetro de controlo é uma medida da taxa de ocorrência da operação de mutação (normalmente um valor na ordem das centésimas) sobre o genótipo, numa dada posição, num dado indivíduo, e para uma dada iteração do AG, a qual é designada por probabilidade de mutação (p_m).

Uma mutação dá origem, regra geral, a uma pequena alteração no código genético do indivíduo, permitindo, nos casos em que se utilizam AG, dotar a procura de uma componente aleatória, a qual se tem revelado central neste tipo de sistemas.

A motivação para a introdução de um operador de mutação advém do facto de, deste modo, se impedir que para um dado gene haja alelos que se percam irremediavelmente. De facto, mesmo quando o valor de um dado gene é o mesmo para toda a população, ou seja, impossível de alterar por uma operação de cruzamento, existe sempre a possibilidade de um novo alelo ser recuperado através de uma operação de mutação.

O operador de mutação utilizado sobre representações em binário inverte cada um dos *bits* dos cromossomas com uma probabilidade p_m, como se ilustra no seguinte algoritmo.

Algoritmo 7 (Mutação).

```
func MUT º
seja bit o bit a mutar
seja mut uma medida da ocorrência ou não da operação de mutação
seja pm a probabilidade de mutação
seja random uma função geradora de números aleatórios Î [0.0, 1.0[
mut ¬ falso
se pm = 1.0 então
                mut ¬ verdadeiro
        senão
                mut ¬ (random £ pm)
fimse
se mut então
                bit ¬ não bit
fimse
retornar bit
```

Quando se utilizam alfabetos de maior cardinalidade pode ser utilizada uma função de ordem entre os vários símbolos do alfabeto para efeitos de mutação de um símbolo, ou alternativamente, escolher aleatoriamente (com igual probabilidade) um dos outros símbolos do alfabeto.

4.5 Porque Funcionam Os Algoritmos Genéticos

Os pressupostos teóricos dos AG baseiam-se no conceito de esquema [Goldberg 1989], um formato que permite a exploração de semelhanças entre cromossomas.

Definição 19 (Esquema)

Um esquema H é construído pela introdução de um novo símbolo, o caracter universal # no alfabeto dos genes - tal esquema representa todos os cromossomas que unificam em todas as posições diferentes de #. Assumindo que o alfabeto é binário e uma população de n cromossomas de comprimento l, existem entre 2^l e $n \cdot 2^l$ esquemas diferentes.

Veja-se então como é que os AG propagam a construção de blocos. Seja, por conseguinte:

- *POP(t)* uma população de *n* cromossomas na iteração *t*, e cada cromossoma uma sequência de caracteres de tamanho *l* no alfabeto binário *{0, 1}*; e
- *H* um esquema, ou seja, uma palavra no alfabeto $\{0, 1, \#\}^l$. Por exemplo, H_1=(0##1100##) é um esquema no conjunto de palavras de tamanho *l*=9. O esquema H_1 representa o conjunto de todos os cromossomas que é possível obter substituindo # ora por 0 ora por 1. Neste caso H_1 é o conjunto de $2^4=16$ cromossomas; e
- a ordem do esquema dada por *o(H)*; i.e., o número de símbolos subtraído do número de caracteres # presentes num cromossoma. Essencialmente, aqui define-se a especificidade de um esquema. Neste caso, $o(H_1)$=5; e
- o comprimento de um esquema dado por *l(H)*; i.e., a distância entre os dois símbolos mais distantes no esquema que sejam diferentes de #. Denota o grau de compactação da informação contida num esquema. Neste caso $l(H_1)$=6.

Pode-se agora caracterizar o efeito que cada operador genético tem na construção de blocos. O resultado é dado pelo teorema do esquema [Goldberg 1989].

Definição 20 (Teorema do Esquema).

Este teorema, também conhecido como Teorema Fundamental dos AG, postula que esquemas de comprimento reduzido com desempenhos acima da média (construção de blocos) crescem em número de uma forma exponencial. Pode ser traduzido pela expressão:

$$m(H,t+1) \geq m(H,t) \cdot \frac{f(H,t)}{\overline{f(t)}} \cdot \left[1 - p_c \frac{l(H)}{l-1} - o(H)\, p_m\right]$$

onde *m(H,t)* é o número de esquemas *H* na população *POP(t)*, *f(H,t)* é o desempenho médio do esquema *H* na iteração *t*, e $\overline{f(t)}$ o desempenho médio da população *POP(t)*, ou seja:

$$\overline{f(t)} = \frac{\sum_{i=1}^{n} ST_i}{n}$$

onde ST_i corresponde ao valor entâlpico de um cromossoma *i* na população *POP(t)*, p_c denota a probabilidade de ocorrência de cruzamentos (uniformemente distribuída), e p_m denota a probabilidade de ocorrência de mutações.

Goldberg [Goldberg 1989] defende que o poder dos AG reside na sua capacidade de descobrir blocos construtores, definidos como esquemas de comprimento reduzido, ou seja, constituídos por um pequeno número de alelos para genes adjacentes. Um bloco construtor de boa qualidade contém genes que funcionam bem quando juntos e melhoram o desempenho dos indivíduos onde estão inseridos.

Para que a formação deste tipo de blocos construtores de boa qualidade seja possível, tem de haver algum cuidado no processo de codificação de soluções, nomeadamente:

- deve existir pouca interacção entre genes, levando a que a sua ordem seja irrelevante; e
- nos casos em que existam dependências entre parâmetros, os genes que codifiquem parâmetros fortemente relacionados devem ser codificados em posições adjacentes.

A interacção entre genes, também designada por epistase, mede a contribuição de um gene para o valor entâlpico do indivíduo a que pertence e depende da sua localização no respectivo cromossoma. Este fenómeno seria evitável se fosse possível cumprir as condições anteriores, o que é um caso raro em problemas reais. Aliás, a sua plena concretização levaria a que fosse possível optimizar cada parâmetro do problema o que, obviamente, não é o caso geral. Resta, então, tentar codificar os parâmetros do problema de modo a minimizar este fenómeno, para que o AG possa funcionar tão bem quanto possível. A questão de encontrar codificações ideais para as soluções de um dado problema, bem como as metodologias para as obter, são temas ainda em aberto.

4.6 Como Funcionam os Algoritmos Genéticos

A Figura 11 dá uma medida de um AG em funcionamento. Neste exemplo é considerada a função multimodal de uma variável *f(X)*, em que $X \hat{I} [a,b]$. Usando o alfabeto binário, a variável *X* pode ser codificada como um cromossoma. O número de *bits* nesta representação depende do domínio da variável *X* e da precisão desejada. Assumindo um comprimento de *l=10 bits*, um cromossoma (e consequentemente, uma solução potencial) pode ser dada por *0011101011*.

Claro que, para avaliar cada número binário, deve-se em primeiro lugar convertê-lo para o equivalente decimal (235 neste caso) e então escalá-lo no domínio *(b-a)*. Por exemplo, para o cromossoma *0011101011* tem-se que:

$$235 \cdot \frac{b-a}{2^{10}-1}$$

De seguida, há que definir o tamanho da população (*n*=20 neste caso), e preenchê-la com cromossomas gerados aleatoriamente (*POP(0), t=0*). Esta primeira operação corresponde a uma amostragem do espaço de soluções. Avaliam-se as amostras e, de seguida, selecciona-se uma nova população composta pelos cromossomas de maior valor entâlpico a quem se aplicam os operadores genéticos para produzir novos exemplares.

O AG é iterado durante um certo número de ciclos. Após algumas destas iterações, os cromossomas concentram-se à volta do sub-espaço de soluções com maior valor entâlpico (Figura 11 após 100 e 1000 iterações). Com um número suficiente de iterações a solução pode ser encontrada (ou aproximada, regra geral).

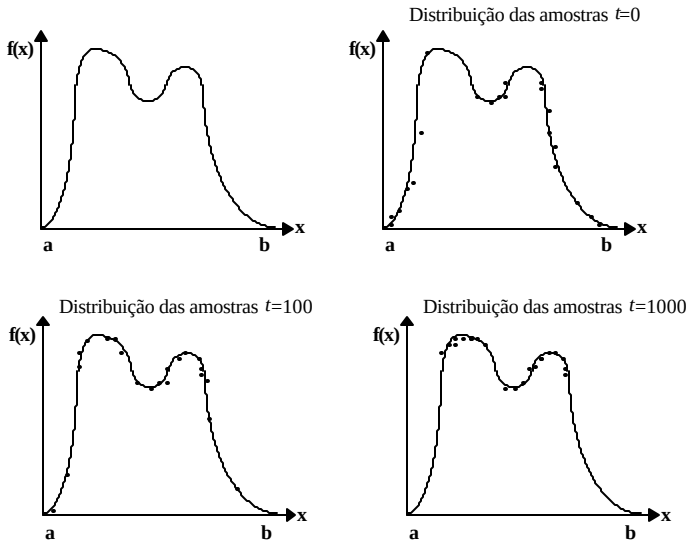


Figura 11 Exemplo de execução de um AG.

4.7 Algoritmos Genéticos nos Sistemas de Classificação

Num processo de aprendizagem o interesse está em maximizar o desempenho do sistema para que a aprendizagem possa ser frutífera; i.e., para ser utilizado nos SC, o AG (convencional) deverá ser alvo de algumas adaptações, que a seguir se anunciam.

Em primeiro lugar, o AG não é invocado em todos os ciclos de vida do SC. Para isso, é definida uma métrica denominada de período do AG, T_{ag}, que denota o número de ciclos (ciclos de regras e mensagens) entre as invocações do AG. Este período pode ser tratado de uma forma determinística (o AG é executado todos os T_{ag} ciclos) ou estocasticamente (o AG é chamado probabilisticamente com um período médio T_{ag}). Adicionalmente, a invocação do AG na aprendizagem pode ser condicionada por eventos particulares, tais como a necessidade de unificação (o sistema não possui regras unificáveis com as mensagens de entrada e consequentemente não gera saídas) ou o decréscimo nos níveis de desempenho no sistema (e.g., a percentagem de soluções correctas).

Em segundo lugar, o processo de selecção é executado da mesma forma, utilizando-se a técnica da roleta onde o valor entâlpico de cada classificador é usado como uma medida da sua aptidão. No entanto, o número de classificadores a seleccionar é $n=2k$, em que k é a cardinalidade da lista de mensagens. Não se geram populações completas, o que obriga a ter um cuidado especial na escolha dos elementos da população que irão ser substituídos por aqueles que são gerados pelas operações de selecção, cruzamento e mutação. O procedimento de purificação definido por De Jong pode ser utilizado para promover a substituição de elementos similares na população [Goldberg 1989].

Definição 21 (Operador de Purificação).

A operação de purificação é utilizada para fazer a escolha dos classificadores a serem eliminados por forma a se obter espaço para acolher os novos classificadores gerados pelos operadores da selecção, cruzamento e mutação. Este método consiste na selecção dos indivíduos, com pior desempenho e mais próximos dos descendentes a serem inseridos na população. Em cada ciclo é escolhido um indivíduo para uma possível substituição. Para esse fim é constituída uma subpopulação de purificação $P_{purif}(t)$ com indivíduos escolhidos de forma aleatória (cujo tamanho n_{purif} £ n é um parâmetro do sistema). Desta população retém-se o indivíduo com menor valor entâlpico - o "pior". Este processo é executado tantas vezes quantas o factor de purificação f_{purif} (um parâmetro do sistema) indicar. No final selecciona-se de entre os f_{purif} elementos aquele que se assemelha mais ao classificador a inserir no sistema.

Algoritmo 8 (Purificação).

Este algoritmo selecciona um classificador para ser substituído.

func *PURIF º*
  seja *C o classificador a inserir na população*
  seja *C' o classificador a remover da população que se pretende seleccionar*
  seja f_{purif} *o factor de purificação*
  seja n_{purif} *o tamanho da subpopulação a analisar*
  $P \neg \{\}$
  para $i \neg 1$ ***até*** f_{purif} ***fazer***
    colocar em P', aleatoriamente, n_{purif} *classificadores;*
    escolher em P' o pior classificador C' (com menor valor entâlpico);

P ¬ P È C'
fim para
em P eleger o classificador C' mais próximo de C
retornar *C'*

Por seu lado, a operação de mutação deve ser alterada porque os SC usam um alfabeto ternário {0,1,#} (ou até de maior ordem). A probabilidade de mutação p_m é definida da mesma forma; todavia, quando ocorrer uma mutação, o caracter mutado deve ser substituído por um dos outros caracteres do alfabeto (para o alfabeto ternário: 0 por 1 ou #, 1 por 0 ou # , # por 0 ou 1), com igual probabilidade. O AG, assim adaptado pode ser utilizado nos SC.

Algoritmo 9 (Algoritmo Genético Adaptado).
Dado um conjunto *C(t)* de classificadores para o ciclo *t*, o AG adaptado aos SC pode ser descrito da seguinte forma:

proc *AG_{sc} º*
seja *C(t) o conjunto de classificadores no ciclo t*
seja *SUB o conjunto dos classificadores a serem substituídos em C(t), SUB Ì C(t)*
seja *NOV o conjunto dos novos classificadores*
seja *k a cardinalidade da lista mensagens*
seja *p_c a probabilidade de cruzamento*
seja *p_m a probabilidade de mutação*
seja *n_{purif} o tamanho da subpopulação de purificação*
seja *f_{purif} o factor de purificação*
seja *A, SUB, NOV conjuntos de classificadores*
SUB ¬ {}
NOV ¬ {}
para *i ¬ 1* ***até*** *k* ***fazer***
A ¬ selecção(C(t),2)
A ¬ cruzamento(A,p_c)
A ¬ mutação(A,p_m)
A ¬ avaliação(A)
SUB ¬ SUB È purificação(C(t),A,n_{purif},f_{purif})
NOV ¬ NOV È A
fimpara
C(t+1) ¬ C(t) - SUB È NOV

onde as funções têm o seguinte desenvolvimento: *selecção(C(t),2)* retorna dois classificadores seleccionados pelo método da roleta a partir da população *C(t);*

cruzamento(A,p_c) retorna um conjunto de classificadores obtidos por cruzamento dos classificadores contidos em *A* usando uma probabilidade p_c; *mutação(A,p_m)* retorna o conjunto dos classificadores presentes em *A* devidamente mutados com probabilidade de mutação p_m; *avaliação(A)* determina a especificidade de cada um dos classificadores contidos em *A; purificação(C(t),A,n_{purif},f_{purif})* retorna o conjunto dos classificadores seleccionados para serem substituídos por aqueles que constam do conjunto A, com um factor de purificação f_{purif} e uma subpopulação de purificação n_{purif}.

5 Tipos de Sistemas de Classificação

A investigação na área dos SC tem dado como resultado muitos tipos de sistemas, não existindo, hoje em dia, um padrão para o SC. Todavia podem distinguir-se duas grandes correntes de investigação, as abordagens de *Michigan* e a de *Pittsburg*. Na primeira, cada um dos classificadores, contido no sistema, codifica uma solução distinta, baseando-se a aprendizagem na cooperação entre os classificadores. Na segunda, o conjunto dos classificadores, no seu todo, codifica apenas uma solução, centrando-se a aprendizagem na optimização do conjunto dos classificadores.

Com a intenção de melhorar a eficiência dos SC, em alguns domínios de aplicação, têm vindo a ser desenvolvidas diferentes versões por incorporação de novas técnicas, sendo possível, na actualidade, fazer-se menção a:

Sistemas de Classificação Baseados na Precisão (SCBP): são um novo tipo de SC, criado por Stewart Wilson [Wilson 1995, 1998, 1999], cuja diferença principal reside no facto de a valorização dos classificadores se basear na precisão; i.e., na precisão com que o sistema prevê o valor que irá receber como recompensa do ambiente.

Sistemas de Classificação Antecipatórios (SCA): são um tipo de SC introduzido por Stolzmann [Stolzmann 1997, 1998, 1999] cuja função de aprendizagem se baseia no mecanismo cognitivo de controlo do comportamento antecipatório. Este modelo é suportado pelas teorias dos mecanismos de aprendizagem oriundas da Psicologia

Cognitiva, iniciadas por Hoffmann [Hoffmann 1992]. Os SCA permitem, neste caso, para além da aprendizagem reforçada, realizar a aprendizagem latente[8].

Sistemas de Classificação de Nível Zero (SCNZ): são uma espécie de avô dos SCBP, foram criados em 1994 por Stewart Wilson [Wilson 1994], essencialmente diferem do SC clássico pela presença de um operador de cobertura que é aplicado nas situações: (i) não existirem classificadores activados durante um ciclo; (ii) a soma dos valores entâlpicos dos classificadores activos ser inferior à média dos valores entâlpicos de toda a população. Este operador tem a missão de criar um novo classificador cuja parte *condição* iguala a mensagem recebida do ambiente do sistema e a parte *acção* é preenchida aleatoriamente ou heuristicamente.

Sistemas de Classificação Heterogéneos (SCH): ao contrário dos SC clássicos, onde todos os classificadores possuem a mesma estrutura e propriedades, no SCH permite-se a existência de grupos heterogéneos de classificadores com estruturas e inclusivamente propriedades diferentes [Lattaud 1997, 1999]. Aplicam-se essencialmente a problemas de aprendizagem multi-objectivo.

Sistemas de Classificção Corporativos (SCC): trata-se de um SC onde os classificadores são ligados por forma a criarem corporações em que a coordenada tempo é central a todos os processos computacionais, e que atendem a aspectos relacionados com as tarefas desenvolvidas numa sequência de passos. Os alicerces deste tipo de SC foram lançados por Tomlinson e Bull [Tomlinson & Bull 1998, 1999].

Sistemas de Classificção Organizacionais (SCO): este tipo de SC situa-se a meio do debate entre os adeptos da abordagem *Michigan* e os adeptos da abordagem *Pittsburg*. Um SCO, combina um conjunto de operadores de dimensionamento organizacional com índices de reputação associados aos classificadores, permitindo diminuir de um

[8] A aprendizagem consiste basicamente na aquisição de conhecimento. Na aprendizagem reforçada, trata-se de conhecimento acerca da forma como conseguir maiores recompensas da parte de um ambiente. A aprendizagem latente consiste na aquisição de conhecimento referente às expectativas criadas em volta das consequências que determinados comportamentos terão no futuro.

modo eficiente o número de classificadores parasitas que se vão instalando no SC [Wilcox 1995].

Referências

[Arthur 1990] Arthur, W. B., A Learning Algorithm that Replicates Human Learning, Technical Report 90-026, Santa Fé Institute, USA, 1990.

[Banzhaf et al. 1998] Banzhaf, W., Nordin, P., Keller, R. E., Francone, F. D., Genetic Programming - An Introduction, Morgan Kaufmann Publishers, Inc., USA, 1998.

[Bart 1994] Bart, Boer. Classifier Systems - A Useful Approach to Machine Learning?, Master's Thesis, Leiden University, The Netherlands, 1994.

[Beasley et al. 1993] Beasley, D., Bull, D., Martin, R., An Overview of Genetic Algorithms: Part 2, Research Topics. University Computing, 15(4):170-181, 1993.

[Bonelli & Parodi 1991] Bonelli, P., Parodi, A., An Efficient Classifier System and its Experimental Comparison with two Representative Learning Methods on Three Medical Domains, in Proceedings of the Fourth International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, San Mateo, California, USA, 1991.

[Booker 1982] Booker, L. B., Intelligent Behavior as an Adaptation to the Task Environment, PhD Dissertation, University of Michigan, USA, 1982.

[Booker 1991] Booker, L. B., Representing Attribute-Based Concepts in a Classifier System, in Foundations of Genetic Algorithms, Morgan Kaufmann Publishers, Inc, USA, 1991.

[Booker et al. 1990] Booker, L.B., Goldberg, D.E., Holland, J.H., Classifier Systems and Genetic Algorithms, in Machine Learning Paradigms and Methods, J.G. Carbonell, ed, The MIT Press, England, 1990.

[Bull & Fogarty 1993] Bull, L., Fogarty, T. C., Coevolving Communicating Classifier Systems for Tracking, in R F Albrecht, C R Reeves & N C Steele, eds, Artificial Neural Networks and Genetic Algorithms, Morgan Kaufmann, USA, 1993.

[Bull 1999] Bull, L., On Using ZCS in a Simulated Continuous Double-Auction Market, in Proceedings of the Genetic and Evolutionary Computation Conference, Orlando, Florida, Morgan Kaufmann Publishers, San Francisco California, USA, 1999.

[Bull et al. 1995] Bull, L., Fogarty, T. C., Snaith M., Evolution in Multi-Agent Systems: Evolving Communicating Classifier Systems for Gait in a Quadrupedal Robot, in L.J. Eshelman, ed, Proceedings of the Sixth International Conference on Genetic Algorithms, Morgan Kaufmann, 1995.

[Cambier 1994] Cambier, C., SIMDELTA: Un Système Multi-Agent pour Simuler la Pêche sur le Delta Central du Niger, PhD Thesis, Université de Paris, France, 1994.

[Carse et al. 1995] Carse B., Fogarty, T. C. Munro, A., Adaptive Distributed Routing using Evolutionary Fuzzy Control, in L. J. Eshelman, ed, Proceedings of the Sixth International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, USA, 1995.

[Cerrada & Aguilar 1998] Cerrada, M. L., Aguilar, J. C., Fuzzy Classifier System: An Application for Fault Tolerance in Industrial Processes, in Hamza, M.H., ed, Proceedings of the Internacional Conference in Artificial Intelligence and Soft Computing, IASTED Press, México, 1998.

[Cribs & Smith 1996] Cribs, B., Smith, Robert E., Classifier System Renaissance:

New Analogies, New Directions, in Proceedings of the First Annual Conference of Genetic Programming, MIT Press, USA, 1996.

[De Jong & Spears 1991] De Jong, K., Spears, W., Learning Concept Classification Rules Using Genetic Algorithms, in Proceedings of the Twelfth International Joint Conference on Artificial Intelligence, Sidney, Australia, 1991.

[De Jong 1975] De Jong, K. A., An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Dissertation Abstracts International 36(10), 5140B. (University Microfilms nº 76-9381), USA, 1975.

[Dorigo & Maniezzo 1993] Dorigo, M., Maniezzo, V., Parallel Genetic Algorithms: Introduction and Overview of Current Research, in Joachim Stender, ed, Parallel Genetic Algorithms: Theory and Applications, IOS Press, USA, 1993.

[Dorigo & Schnept 1992] Dorigo, M., Schnept, U., Genetics-Based Machine Learning and Behaviour-based Robotics: A New Synthesis, IEEE Transactions On Systems Man and Cybernetics, 1992.

[Druhan & Mathews 1989] Druhan, B. B., Mathews, R., C., THIYOS: A Classifier System Model of Implicit Knowledge in Artificial Grammars, in Proceedings of the Annual Conference of the Cognitive Science Society, USA, 1989.

[Dwormann 1994] Dwormann, G., Games Computers Play: Simulating Characteristic Function Game Playing Agents with Classifier Systems, in Proceedings of the 1994 IEEE Conference on Evolutionary Computation, IEEE, 1994.

[Farmer 1990] Farmer, J. D., A Rosetta Stone for Connectionism, in Stephanie Forrest, eds, Proceedings of the Ninth Annual International Conference of the Center for Nonlinear Studies on Self-organizing, Collective, and

Cooperative Phenomena in Natural and Artificial Computing Networks, in Physica D, vol. 42, North Holland, 1990.

[Federman et al. 1997] Federman, F., Dorchak, S., Representation of Music in a Learning Classifier System, in RAS & Skowron, eds, Proceedings of Foundations of Intelligent Systems: 10th International Symposium ISMIS'97, Charlotte, North Carolina, Springer-Verlag, Germany, 1997.

[Federman et al. 1999] Federman, F., Sparkman, G., Watt, S., Representation of Music in a Learning Classifier System Utilizing Bach Chorales, in Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann Publishers, San Francisco California, USA, 1999.

[Flockhart 1995] Flockhart, I., GA-Miner: Parallel Data Mining and Hierarchical Genetic Algorithms, Technical Report: EPCC-AIKMS-GA-MINER-REPORT 1.0, Univ. of Edinburgh, United Kingdom, 1995.

[Forrest 1985] Forrest, S., A Study of Parallelism in the Classifier System and its Application to Classification in KL-ONE Semantic Networks, PhD Thesis, The University of Michigan, Ann Arbor, Michigan, USA, 1985.

[Forrest 1991] Forrest, S., Parallelism and Programming in Classifier Systems, Morgan Kaufmann Publishers Inc., San Mateo, California, USA, 1991.

[Frey & Slate 1991] Frey, P. W., Slate, D. J., Letter Recognition Using Holland-style Adaptive Classifiers, Machine Learning 6, 161-182, 1991.

[Giordana et al. 1994] Giordana, A., Neri, F. Saitta, L., Search-Intensive Concept Induction, Technical Report, Universita di Torino, Dipartimento di Informatica, Torino, Italy, 1994.

[Goldberg 1983] Goldberg, D.E., Computer-aided Gas Pipeline Operation Using Genetic Algorithms and Rule Learning, PhD Thesis, University of

Michigan, USA, 1983.

[Goldberg 1989] Goldberg, D. E., Genetic Algorithms in Search, Optimization, and Machine Learning. Addison Wesley Publishers, USA,1989.

[Hilty 1992] Hilty, John A., Prediction of Social Events Using a Classifier System, PhD Thesis, University of Illinois at Urbana-Champaign, USA, 1992.

[Holland & Miller 1991] Holland, J. H., Miller, J., Artificially Adaptive Agents in Economic Theory, American Economic Review, 81(2):365-370, 1991.

[Holland & Reitman 1978] Holland, J. H., Reitman, J. S., Cognitive Systems Based on Adaptive Algorithms, in Waterman, D. A., Heyes-Roth, F., eds, Pattern-Directed Inference Systems, Academic Press, USA, 1978.

[Holland 1975] Holland, J. H., Adaptation in Natural and Artificial Systems, Ann Arbor: University of Michigan Press, USA, 1975.

[Holland 1985] Holland, J., Properties of the Bucket Brigade, in Proceedings of the First International Conference on Industrial an Engineering Applications of Artificial Intelligence and Expert Systems, ACM Press, USA, 1985.

[Holland 1995] Holland, J. H., Adaptation in Natural and Artificial Systems, Ann Arbor: University of Michigan Press, USA, 1995.

[Holland 1997] Holland, J. H., A Ordem Oculta - Como a Adaptação Gera a Complexidade, Gradiva, Portugal, 1997.

[Holland et al. 1986] Holland, J.H., Holyoak, K.J., Nisbett, R.E., Thagard, P. R. Induction: Processes of Inference, Learning and Discovery, MIT Press, Cambridge, MA, USA, 1986.

[Holmes 1997] Holmes, J. M., Discovering Risk of Disease with a Learning Classifier System, in Bäck, T., ed, Proceedings of the Seventh ICGA,

Morgan Kaufmann Publishers, USA, 1997.

[Holmes 1999] Holmes, J. H., Quantitative Methods for Evaluating Learning Classifier System Performance in Forced Two-Choice Decision Tasks, in Proceedings of the Second International Workshop on Learning Classifier Systems, USA, 1999.

[Holyoak et al. 1990] Holyoak, K. J., Koh, K., Nisbett, R. E., A Theory of Conditioning: Inductive Learning within Rule-Based Default Hierarchies, Psycha Review 96, 315-340, 1990.

[Janikow 1991] Janikow, C. Z., Inductive Learning of Decision Rules From Attribute-based Examples: A Knowledge-intensive Genetic Algorithm Approach, PhD Thesis, The University of North Carolina at Chapel Hill, USA, 1991.

[Lanzi 1999] Lanzi, P. L., Reinforcement Learning with Classifier Systems, PhD Thesis, Politecnico de Milano, Italy, 1999.

[Lattaud 1997] Lattaud, C., A Classification for the Control of the Evolution of Adaptative Agents, in Proceedings of the 10th International F. Artificial Intelligence Research Symposium, 449-454, Daytona, USA, 1997.

[Lattaud 1999] Lattaud, C., Non-Homogeneous Classifier Systems in a Macro-Evolution Process, in Proceedings of the Second International Workshop on Learning Classifier Systems, USA, 1999.

[Liepins & Wang 1991] Liepins, G. E., Wang, L. A., Classifier System Learning of Boolean Concepts, in Proceedings of the Fourth International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, 1991.

[Marengo & Tordjman 1996] Marengo, L., Tordjman, H., Speculation, Heterogeneity and Learning: A Model of Exchange Rate Dynamics,

KYKLOS 49(3):407-438, 1996.

[Palmer et al. 1994] Palmer, R., Arthur, W. B., Holland, J. H., LeBaron, B., Tayler, P., Artificial Economic Life: A Simple Model of a Stock market, Physica D 75:264-274, 1994.

[Pike 1980] Pike, M. C. Algorithm 267 Random Normal Deviate G5, in Collected Algorithms from ACM, II, 267, 1980.

[Potter et al. 1995] Potter, M., De Jong, K., Grefenstette, J., A Coevolutionary Approach to Learning Sequential Decision Rules, in L. J. Eshelman, ed, Proceedings of the Sixth International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, 1995.

[Riolo 1988] Riolo, R. L., Empirical Studies of Default Hierarchies and Sequences of Rules in Learning Classifier Systems, PhD Dissertation, University of Michigan, USA, 1988.

[Riolo 1991a] Riolo, R. L., Lookahead Planning and Latent Learning in Classifier System. in Meyer, J. A., Wilson, S. W., eds, From Animals to Animates: Proceedings of the First International Conference on Simulation of Adaptative Behaviour, The MIT Press, Cambridge, MA, USA, 1991.

[Riolo 1991b] Riolo, R. L., Modeling Simple Human Category Learning with a Classifier System, in Proceedings of the Fourth International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, 1991.

[Robert & Sheri 1996] Robert, A. R., Sheri, D. S., Three-dimensional Shape Optimization Utilizing a Learning Classifier System, in Genetic Programming - Proceedings of the First Annual Conference, The MIT Press, USA, 1996.

[Robertson 1987] Robertson, G. G., Parallel Implementation of Genetic Algorithms in a Classifier System, in Proceedings of the Second International

Conference on Genetic Algorithms, Lawrence Erlbaum Associate Publishers, 1987.

[Rocha & Neves 1998] Rocha, M., Neves, J., Uma Aproximação à Resolução do Problema do Caixeiro Viajante via Programação Genética, Unidade de Ensino, Departamento de Informática, Universidade do Minho, Braga, Portugal, 1998.

[Sedbrook et al. 1991] Sedbrook, T. A., Wright, H., Wright, R., Application of a Classifier for Patient Triage, in Proceedings of the Fourth International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, 1991.

[Seredynski et al. 1995] Seredynski, F, Cichosz, P., Klebus, G., Learning Classifier Systems in Multi-Agent Environments, in Proceedings of the First IEE/IEEE Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications, 1995.

[Sichman et al. 1997] Sichman, J. M., Conte, R., Demazeau, Y., Castelfranchi, C., A Social Reasoning Mechanism Based on Dependence Networks, in Huhns, M. N., Singh, M. P., eds, Readings in Agents, Morgan Kaufmann Publishers, San Francisco, pp 416-420, USA, 1997.

[Singh & Nuhuns 1994] Singh, M. P., Nuhuns, M. N., Automating Workflows for Service Provisioning: Integrating AI and Database Technologies, IEEE Expert, 9(5):19-23, October, 1994.

[Smith & Cribbs 1993] Smith, R. E., Cribbs, H. B., Is a Learning Classifier System a Type of Neural Network? 1993.

[Smith 1980] Smith, S. F., A Learning System Based on Genetic Adaptive Algorithms, PhD Dissertation, University of Pittsburg, 1980.

[Smith 1991] Smith, R. E., Default Hierarchy Formation and Memory Exploitation in Learning Classifier Systems, PhD Dissertation, The University of

Alabama, Tuscaloosa, Alabama, USA, 1991.

[Smith et al. 1999] Smith, R. E., Dike, B. A., Mehra, R. K., Ravichandran, B., El-Fallah, A., Classifier Systems In Combat: Two-Sided Learning of Manoeuvres For Advanced Fighter Aircraft. Computer Methods in Applied Mechanics and Engineering, Elsevier, (in Press), 1999.

[Stolzman 1997] Soltzman, W., Antizipative Classifier Systeme, PhD Dissertation, Fachbereich Mathematik/Informatik, University of Osnabrueck, Aachen: Shaker Verlag, Germany, 1997.

[Stolzman 1998] Soltzman, W., Antecipatory Classifier Systems, in Koza, J. R. et al., eds, Genetic Programming – Proceedings of the Third Annual Conference, University of Wisconsin, Madison, Wisconsin, San Francisco, Morgan Kaufmann Publishers, USA, 1998.

[Stolzman 1999] Soltzman, W, Latent Learning in Khepera Robots with Anticipatory Classifier Systems, in Proceedings of the Genetic and Evolutionary Computation Conference, Orlando, Florida, Morgan Kaufmann Publishers, San Francisco California, 1999.

[Tomlinson & Bull 1998] Tomlinson, A., Bull, L., A Corporate Classifier System, in Eiben, A. E., Back, T., Schoenauer, M., Schwefel, H. P., eds, Parallel Problem Solving from Nature, PPSN V, pp. 550-559, Springer, 1998.

[Tomlinson & Bull 1999a] Tomlinson, A., Bull, L., On Corporate Classifier Systems: Increasing the Benefits From Rule Linkage, in Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., Smith, R. E., eds, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99), Morgan Kaufmann Publishers, USA, 1999.

[Wilcox 1995] Wilcox, J. R., Organizational Learning Within A Learning Classifier System, Msc Thesis, University of Illinois, USA, 1995.

[Wilson & Goldberg 1989] Wilson, S., Goldberg, D., A Critical Review of Classifier Systems, in Proceedings of The Third International Conference On Genetic Algorithms, Morgan Kaufmann Publishers, San Mateo California, USA, 1989.

[Wilson 1985] Wilson, S. W., Knowledge Growth in An Artificial Animal, in Proceedings of the First International Conference on Genetic Algorithms and their Applications, 1985.

[Wilson 1994] Wilson, S. W., ZCS: A Zeroth Level Classifier System, in Evolutionary Computation, 2(1): 1-18, 1994.

[Wilson 1995] Wilson, S. W., Classifier Fitness based on Accuracy, in Evolutionary Computation 3(2), 149-175, 1995.

[Wilson 1998] Wilson, S. W., Generalization in the XCS Classifier System, in Proceedings of Third Annual Genetic Programming Conference (GP-98), 1998.

[Wilson 1999] Wilson, S. W., State of XCS Classifier System Research, in Proceedings of the Second International Workshop on Learning Classifier Systems, USA, 1999.

Apêndice A
Referencial de Sistemas de Classificação

Neste ponto é compilado um conjunto significativo de referências relativas à área dos SC e estruturadas em cinco grupos distintos: conferências mais importantes; doutoramentos realizados; livros editados; domínios de aplicação; e um guia de endereços de páginas Web que funcionam como portais de acesso à Internet para consulta de informação sobre os SC.

Conferências

- ICGA ("International Conference on Genetic Algorithms");
- ACEP ("Annual Conference on Evolutionary Programming");
- GECCO ("Genetic and Evolutionary Computation COnference") ;
- GP ("Annual Conference on Genetic Programming").
- "International Workshop on Learning Classifier Systems"; realizado pela primeira vez em Outubro de 1992, no Jhonson Space Center, Houston, TX, USA.

Doutoramentos

[Smith 1980] [Booker 1982] [Goldberg 1983] [Forrest 1985] [Riolo 1988] [Janikow 1991] [Simth 1991] [Hilty 1992] [Stolzman 1997] [Lanzi 1999]

Livros editados/em edição

[Holland 1975, 1995, 1997] [Goldberg 1989] [Forrest 1991] [Smith et al. 1999]

Domínios de aplicação

- controlo [Goldberg 1983, 1989];
- reconhecimento de escrita [Frey & Slate 1991];
- medicina [Bonelli & Parodi 1991] [Sedbrook et al. 1991] [Holmes 1997];
- ciências sociais [Hilty 1992];
- robótica [Smith 1980] [Dorigo & Schnept 1992] [Stolzmann 1999];
- aprendizagem de conceitos [Liepins & Wang 1991] [De Jong & Spears 1991] [Janikow 1991];
- optimização [Robert & Sheri 1996];
- aviação militar [Smith et al. 1999];
- aprendizagem latente [Riolo 1991a] [Stolzmann 1997, 1998, 1999];
- modelação/simulação do comportamento de organismos adaptativos e vida artificial [Riolo 1982] [Wilson 1985] [Holyoak et al. 1990] [Druhan & Mathews 1989] [Holland et al. 1986] [Riolo 1991b];
- ambientes multi-agente [Bull & Fogarty 1993][Bull et al. 1995] [Bull 1999] [Smith 1980] [Dorigo & Schept 1992] [Carse et al. 1995] [Potter et al. 1995] [Seredynski et al. 1995];
- economia computacional [Arthur 1990] [Holland & Miller 1991] [Marimon et al. 1990] [Dwormann 1994] [Marengo & Tordjman 1996] [Palmer et al. 1994];
- sistemas de tratamento de falhas em processos industriais [Cerrada & Aguilar 1998];
- música [Federman et al. 1997, 1999]; e
- "data mining" [Giordana et al. 1994] [Flockhart 1995].

Guia de Recursos Electrónicos

Portal	URL
GASLAB Genetic Adaptive Systems Lab. Sob a direcção de Sushil J. Louis, está sediado no departamento de Ciências da Computação da Universidade de Nevada-Reno.	http://gaslab.cs.unr.edu/
GAG George Mason University GA Group. Sob a direcção de Kenneth A. De Jong, está sediado na Universidade George Mason em Fairfax na Virginia.	http://www.cs.gmu.edu/research/gag
IlliGAL Illinois Genetic Algorithms Lab. Sob a alçada de David E. Goldberg, está sediado no Departamento de Engenharia na Universidade de Illinois em Urbana-Champaign.	http://gal4.ge.uiuc.edu/
GARAGe Genetic Algorithms Research and Applications Group. Sediado na Universidade Estatal de Michigan e sob a direcção de Bill Punch.	http://garage.cse.msu.edu/
NCARAI The Navy Center for Applied Research in Artificial Intelligence. Sediado na Divisão de Tecnologias da Informação do Laboratório de Investigação Naval, sob a direcção de Alan Meyrowitz.	http://www.aic.nrl.navy.mil/
EVONET Rede de Excelência em Computação Evolutiva, criada em 1996 sob os auspícios do ESPRIT (programa Europeu de Tecnologias da Informação).	http://www.dcs.napier.ac.uk/evonet/
Adaptive Computation Group Sediado no Departamento de Ciências da Computação da Universidade do Novo México sob a direcção de Stephanie Forrest.	http://www.cs.unm.edu/~forrest/adaptive-web/ac_main.htm
GEMINA Conexionist, Evolutionary and Genetic Computational Group. Sediado no Departamento de Informática da Universidade do Minho, sob a direcção de José Neves.	http://www.venus.di.uminho.pt/SI/GEMINA

Portal	URL
Learning Classifier Systems Página sediada na Universidade de West of England, mantida por Alwyn Barry.	http://www.csm.uwe.ac.uk/ãmbarry/LCSWEB/classsys.htm

Apêndice B
Áreas de Aplicação dos Algoritmos Genéticos

Segue-se uma lista onde se pretende apenas realçar algumas das áreas em que os AG foram aplicados com maior sucesso [Rocha & Neves 1998].

Optimização Os AG foram aplicados a um número muito vasto de tarefas de optimização, incluindo a optimização, numérica e problemas de optimização combinatorial tal como o desenho de circuitos, planeamento, e ao problema do caixeiro viajante.

Programação automática Os AG foram utilizados na evolução de programas de computador para tarefas específicas, e no desenho de outras estruturas computacionais como autómatos celulares e redes de ordenação.

Aprendizagem automática Os AG foram utilizados em muitas aplicações de aprendizagem automática, onde se inclui a classificação e previsão (e.g., a previsão do tempo ou a estrutura de proteínas). Foram também utilizados no melhoramento de sistemas de aprendizagem, tal como na caracterização dos pesos associados às ligações em redes neuronais, regras para os SC (a utilização estudada no contexto deste trabalho) ou sistemas de

	produção simbólicos, e sensores para robots.
Economia	Os AG foram utilizados na modelação de processos de inovação, desenvolvimento de estratégias de licitação, e nascimento de mercados económicos.
Sistemas imunitários	Os AG foram utilizados na modelação de características de sistemas imunitários naturais, incluindo a mutação somática que ocorre durante a vida de um indivíduo, e a descoberta de famílias de multi-gene durante o tempo de evolução.
Ecologia	Os AG foram utilizados para a modelação de fenómenos ecológicos tais como a corrida às armas biológicas, coevolução de parasita-portador, simbiose, e fluxo de recursos.
Populações genéticas	Os AG foram utilizados no estudo de questões ligadas às populações genéticas: "Sob que condições um gene para recombinação será evolutivamente viável?"
Evolução e aprendizagem	Os AG têm vindo a ser utilizados no estudo da correlação entre a aprendizagem individual e a evolução da sua espécie.
Processamento de imagem	Os AG têm vindo a ser aplicados com sucesso no alinhamento e análise de imagens.
Sistemas sociais	Os AG foram utilizados no estudo dos aspectos evolutivos dos sistemas sociais, tais como a evolução do comportamento de colónias de insectos e, mais genericamente, na evolução da cooperação e comunicação em sistemas multi-agente.