Otimização Automática de Parâmetros de Sistemas Evolutivos de Acordo com a Aplicação

Marcio Kassouf Crocomo, Eduardo do Valle Simões, Alexandre C. B. Delbem

fcc1@merconet.com.br, simoes@icmc.usp.br, acbd@icmc.usp.br

Instituto de Ciências Matemáticas e de Computação (ICMC) – Universidade de São Paulo (USP - São Carlos)

Av. do Trabalhador São-Carlense, 400 - Centro - Cx. Postal 668

São Carlos - São Paulo - Brasil CEP 13560-970

Abstract. *This paper describes a strategy applying two genetic algorithms to solve a navigation problem of real robots. This approach allows the use of genetic algorithms by non-specialized users. To achieve this, a second genetic algorithm is applied to adjust the parameters of the first one, which is directly applied to the problem. The automatic optimization of the parameters of the first algorithm was achieved this way, obtaining maximum performance with the selected functions. The obtained results show a speed improvement of 12.9% in relation to the configuration chosen by a specialist.*

Resumo. *Este artigo descreve uma estratégia que emprega dois algoritmos genéticos para resolver um problema de navegação de robôs reais. Para possibilitar a um usuário sem conhecimento de técnicas de otimização de desempenho de algoritmos genéticos a aplicação destes para solucionar seus problemas, é proposto um sistema onde um algoritmo genético é empregado para ajustar os parâmetros de um outro, que por sua vez, é aplicado no problema em questão. Assim, é conseguida automaticamente a otimização dos parâmetros do primeiro algoritmo para se obter ótima performance com as funções selecionadas. Os resultados obtidos mostram que o ganho de desempenho foi de cerca de 12.9% em comparação com os resultados obtidos por um especialista.*

1. Introdução

Várias aplicações de robôs reais têm como pré-requisito uma rápida adaptação dos mesmos ao meio ambiente onde trabalham [1]. Ao se utilizar Algoritmos Genéticos [2] surge o problema de que a velocidade de adaptação dos robôs depende do tamanho da população destes, e várias vezes a população de robôs disponível é pequena (o que

prejudica a velocidade de aprendizado) [8], como pode ser visto na figura 1. Neste artigo, é apresentado o desenvolvimento de um Algoritmo Genético (AG2), que trate de ajustar os parâmetros de um outro Algoritmo Genético (AG1) que é responsável por encontrar uma solução para o problema original, de forma que AG2 seja independente deste problema, mas dependa apenas da forma como AG1 é estruturado. A figura 2 apresenta um diagrama representando esta estratégia.

Tradicionalmente, é o usuário quem deve ajustar os parâmetros do AG [2] para encontrar mais rapidamente a solução para um determinado problema. Neste artigo, é proposta uma solução que emprega um AG para fazer esse ajuste (AG2 na figura 2). Mas quem irá ajustar os parâmetros do AG2? Essa solução ainda exige que o desenvolvedor do software ajuste os parâmetros do AG2. Porém, uma vez ajustados, os parâmetros devem funcionar corretamente sempre, desde que a estrutura do AG1 não mude (Os parâmetros do AG2 dependem da estrutura do AG1).

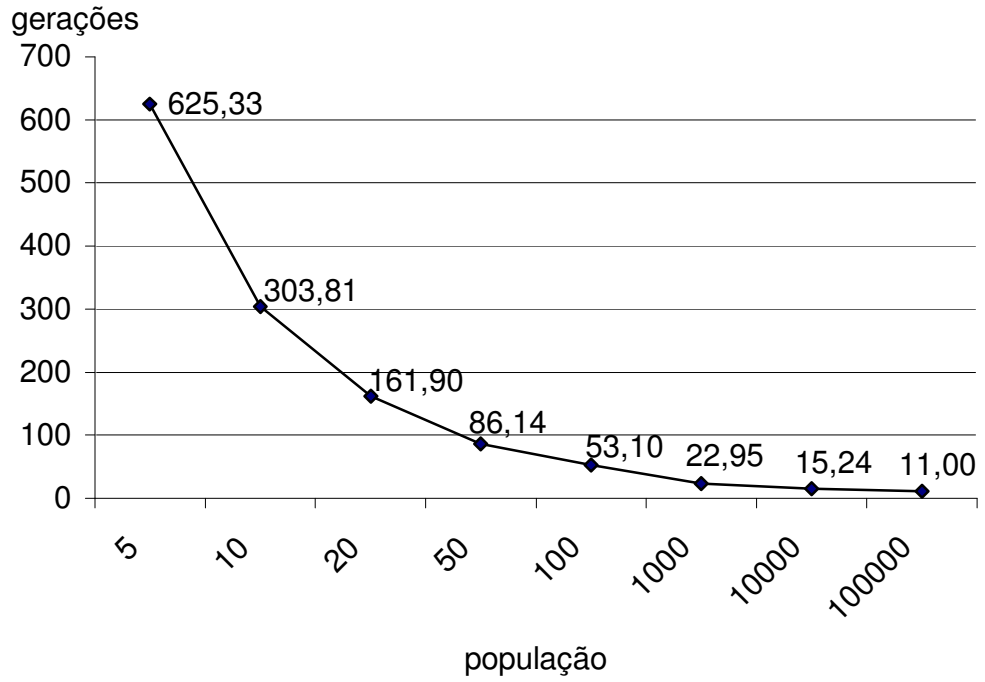


Figura 1 – Gráfico mostrando a influência do tamanho da população na velocidade de aprendizado do algoritmo genético.

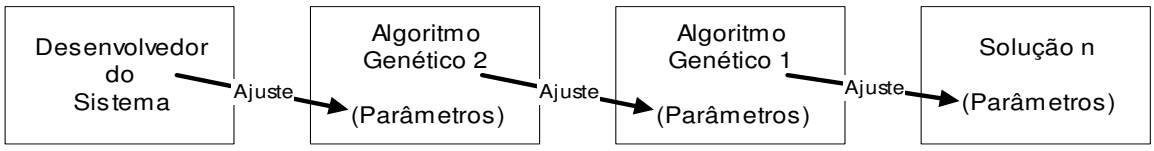


Figura 2 – Dois sistemas evolutivos combinados para resolver um determinado problema

2. O Ambiente de Trabalho

2.1 O Robô

A figura 3 apresenta o robô com seus seis sensores dispostos ao redor do chasis. A tabela da figura 4 mostra como funciona o navegador de um robô: quando um sensor do robô detecta algum obstáculo ele é marcado como 1, caso contrário é marcado como 0, os sensores são a entrada do navegador, para cada situação de sensores, tem-se uma saída que indica a ação que o robô deve tomar.

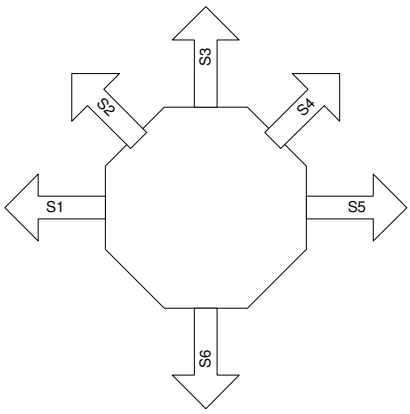


Figura 3 – O robô e seus sensores

S1	S2	S3	S4	S5	S6	Saída
0	0	0	0	0	0	0
0	0	0	0	0	1	1
0	0	0	0	1	0	2
1	1	1	1	1	1	3

Figura 4 – O Navegador do robô na forma de uma tabela verdade contendo todas as possibilidades de saída para cada entrada possível.

2.2 O Simulador Utilizado (AG1)

O teste das funções foi feito utilizando o simulador apresentado a seguir [3]. A figura 5 apresenta o simulador utilizado.

1. Gera-se aleatoriamente os cromossomos de uma população de robôs.
2. Comparam-se as saídas dos cromossomos dos robôs com um cromossomo "meta" para gerar o *fitness*[4].

3. Funções de mutação[5], *crossover*[6], predação randômica e de síntese, realizam a convergência dos cromossomos em aprendizado para o cromossomo meta.

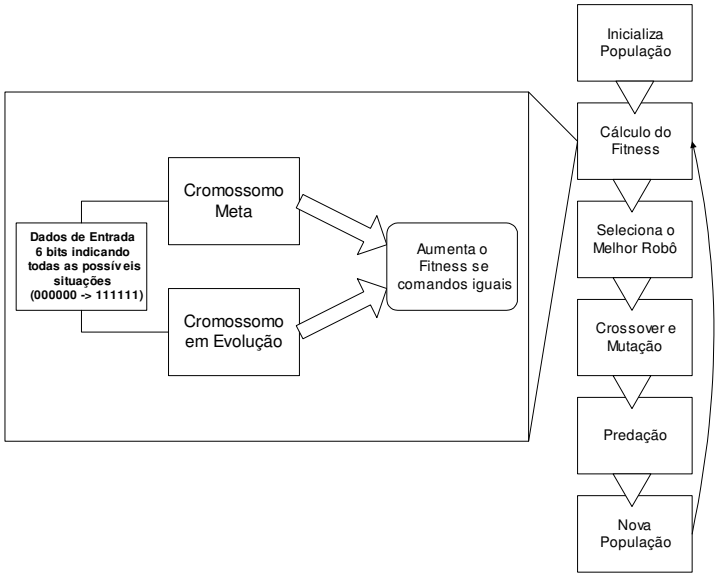


Figura 5 – Um diagrama do Simulador Utilizado

Funções utilizadas:

•Predação Randômica:

O cromossomo de menor pontuação tem seus parâmetros iniciados novamente (de forma randômica), esta função tem como objetivo inserir variedade genética na população. Isto acontece com um período determinado, após um número específico de gerações.

•Predação por Síntese:

O cromossomo de menor pontuação tem seus locus substituídos pelos resultados mais comuns encontrados nos outros cromossomos (como mostra a figura 6). A função tem como objetivo aumentar a velocidade do aprendizado, sintetizando as características mais freqüentes da população.

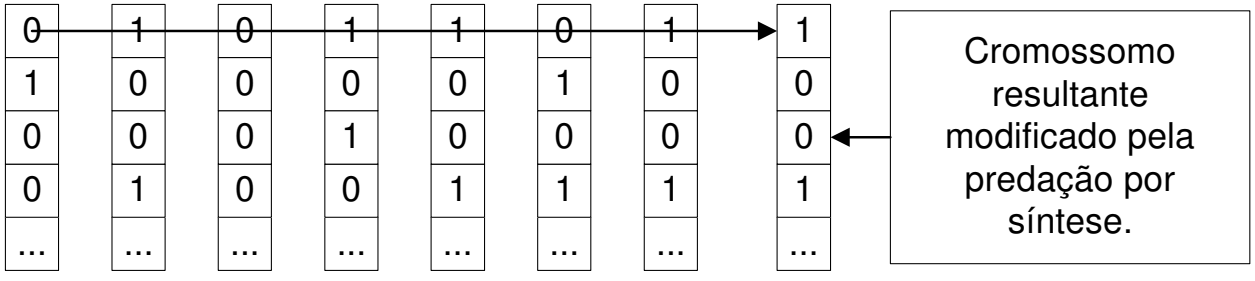


Figura 6 – Um exemplo da predação por síntese

Parâmetros utilizados para AG1:

- Taxa de mutação (variando entre 0.1% e 100%)

- Uso da Função Predação Randômica (1 sim, 0 não)

- Uso da Função Predação por Síntese (1 sim, 0 não)

- Período da Função Predação Randômica (intervalos de gerações em que a função opera).

- Período da Função Predação por Síntese (intervalos de gerações em que a função opera).

Os parâmetros apresentados acima são os parâmetros de configuração do AG1, que serão ajustados pelo AG2 (os cromossomos gerados por AG2), ressaltando que os parâmetros apresentados são os trabalhados nos experimentos apresentados neste artigo; para outras aplicações de AGs, esses parâmetros mudariam conforme a estrutura dos respectivos AG1s.

3 O AG2

3.1 A População

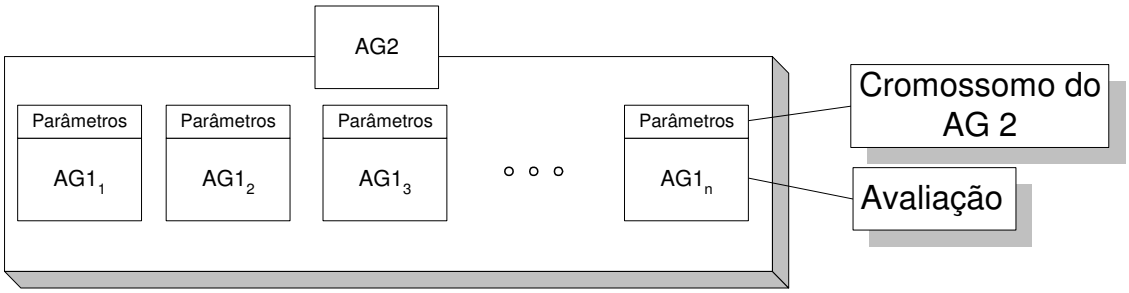


Figura7 – Um diagrama do AG2, mostrando como este tenta evoluir a população de parâmetros para AG1 que são responsáveis pela solução do problema.

A População do AG2 constitui-se das listas de parâmetros do AG1, o AG1 é responsável pelo cálculo do fitness dos indivíduos do AG2 (como mostra a figura 7).Iniciou-se o AG2 utilizando reprodução sexuada, porém foi verificado mais tarde que a reprodução assexuada apresentava uma convergência mais rápida, o AG2 atual divide parte de sua população em reprodução sexuada e parte em assexuada (figura 8).

3.2 Funções Utilizadas

As funções realizadas pelo sistema evolutivo são explicadas abaixo e estão representadas no diagrama da figura 8.

- **Inicializar randomicamente os indivíduos:**

  Cada um dos 20 indivíduos tem seus parâmetros inicializados de forma randômica.

- **Aplicar mutação em todos os indivíduos.**

  A mutação é aplicada de acordo com a taxa de mutação em todos os cromossomos menos no que possui o melhor fitness. Cada item do cromossomo é mutado diferentemente. Para a taxa de mutação e os períodos das funções de predação, é gerado um número randômico entre -5 e +5 que é somado ao atual. Cada uma das funções de predação tem uma chance de 10% de mutar, ou seja, de inverter o bit que seleciona o uso da técnica.

- **Calculo do fitness:**

  É chamado o AG1 para cada cromossomo 20 vezes e é calculada a média (fitness) do número de gerações que foram necessárias para se atingir o cromossomo meta.

- **Análise do Progresso:**

  Observa-se o gradiente da curva, caso este indique que a convergência está ocorrendo muito lentamente, a população de indivíduos que efetuará reprodução assexuada será aumentada, e a população que efetuará reprodução sexuada é diminuída gradativamente, de 50% assexuada a 80%.

- **Seleção do cromossomo de melhor fitness.**

  O cromossomo que contiver o conjunto de parâmetros que obteve o aprendizado mais rápido quando aplicado a AG1 é selecionado.

- **Copiar o cromossomo selecionado para os indivíduos da População**:

  Na reprodução assexuada, copia-se o indivíduo com melhor fitness [2]. para os outros membros que se reproduzem de forma assexuada.

- **Aplica-se mutação aos cromossomos:**

  Após copiar o indivíduo selecionado para os membros assexuados, aplica-se mutação nos mesmos.

- **Crossover e mutação:**

  Utilizando a técnica de elitismo [7], é aplicado o crossover entre o indivíduo selecionado e os outros membros sexuados, em seguida é aplicada a mutação.

- **Predação por Síntese e Randômica**

  Depois de aplicada a mutação, entram em vigor as funções de predação, agindo apenas sobre os indivíduos sexuados.

- **Critério de parada.**

  É difícil dizer se a solução encontrada para uma função com várias variáveis é a solução ótima do problema, então deve-se parar o processo quando se obter resultados satisfatórios, que geralmente serão apresentados quando não for produzida mais nenhuma melhora por um determinado número de gerações.

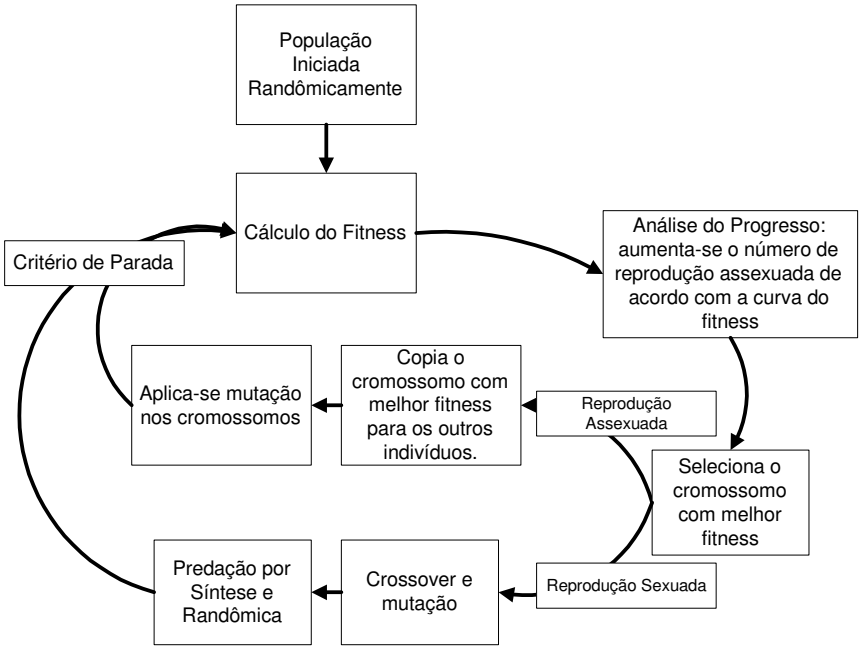


Figura 8 – O Diagrama de operação do AG2

4. Resultados Obtidos

O gráfico abaixo (Figura 9) ilustra os testes realizados manualmente para se atingir os melhores parâmetros para combinar as funções do AG1; este gráfico foi feito utilizando-se a média de 20 testes realizados. É mostrado a influência do período utilizado na função Predação Randômica em uma população de 5 indivíduos, já fixado o melhor período para a função Síntese (1), encontrado analisando um gráfico da mesma forma. Pode-se verificar que a relação entre esses parâmetros não ocorre de forma trivial, o que mostra a importância do AG2 para otimização dos mesmos.

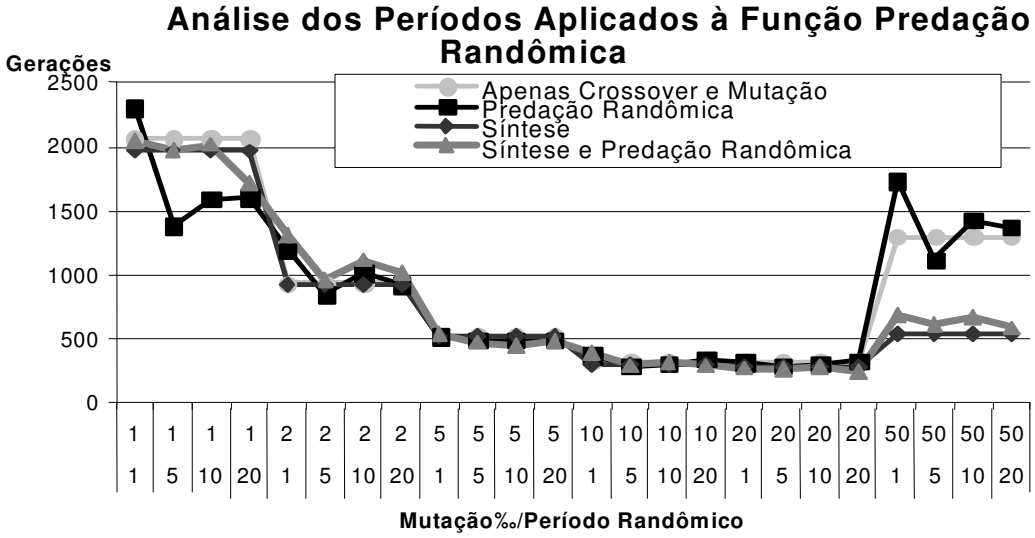


Figura 9 – Testes realizados variando-se período randômico e mutação

Procurou-se inicialmente encontrar os melhores parâmetros para o AG1 acima. Utilizando-se de tentativa e erro, foram realizados vários experimentos e incluídas as funções Predação Randômica e Predação por Síntese. Foi conseguido então uma diminuição significativa no número de gerações necessárias para se atingir o resultado esperado no AG1 (fitness do AG2), no entanto, ao se aplicar o AG2 (figura 10) para otimizar automaticamente esses parâmetros, foi conseguida uma melhoria ainda maior na eficiência do AG1.

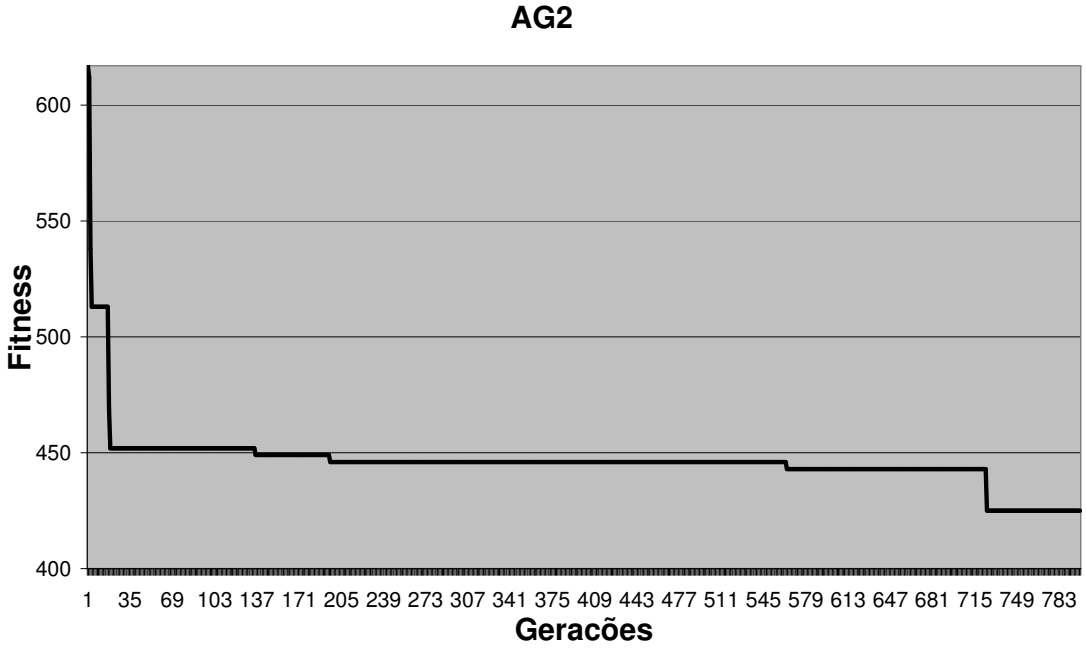


Figura 10 – AG2 Encontrando os melhores parâmetros para AG1

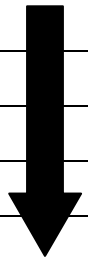
5. Conclusão

Foram testadas exaustivamente todas as possíveis combinações de parâmetros para se encontrar o melhor caso (dentro de um espaço de busca reduzido onde se sabia que se encontrava o melhor caso, analisando o gráfico da figura 9), que foi o mesmo caso encontrado utilizando o AG2: mutação = 1,9%, com predação por síntese com período igual a 1 e predação randômica com período igual a 73. A Tabela I apresenta um sumário dos resultados obtidos. Observa-se uma melhoria de aproximadamente 30.28% pela adição das funções Predação por Síntese e Predação Randômica (resultados obtidos manualmente) Ao se aplicar o AG2 para encontrar o melhor conjunto de parâmetros para as funções, a melhora obtida automaticamente foi de 12.9%, ou seja, aproximadamente 40% desde a etapa original.

Desta maneira, este artigo traz uma solução para aqueles usuários que desejam aplicar AGs para resolver seus problemas mas não possuem grande experiência na otimização de seus parâmetros. Assim, o usuário precisa apenas especificar o problema e a função de avaliação para o AG1. A partir daí, o sistema duplo proposto pode automaticamente configurar o AG1 para resolver de maneira eficiente o problema.

Tabela I – Número de gerações necessárias para atingir o objetivo no AG1 em relação à inclusão de funções específicas.

Etapas	**Melhor caso (geração)**
Apenas *Crossover* e Mutação	700
Inserção da função Predação Randômica	644
Inserção da função Predação por Síntese	543
Utilizadas Predação Randômica e Síntese	488
Utilizando AG2 para se obter melhore parâmetros	425



6. Referencias Bibliográficas

[1] Floreano, D. and Mondada, F., *Hardware Solutions for Evolutionary Robotics*. Proceedings of the First European Workshop on Evolutionary Robotics, Husbands, P. and Meyer, J.-A. (Eds.), Publisher: Springer Verlag, Berlin, Germany, pp. 137-151, 1998.

[2] Shipman, R., Shackleton, M., Ebner, M., and Watson, R. A., *Neutral Search Spaces for Artificial Evolution: A Lesson From Life*. Proceedings of the Seventh International Workshop on the Synthesis and Simulation of Living Systems - Artificial Life VII, Aug. 1-2, 2000, Reed College, Portland, Oregon, USA, Bedau, M., McCaskill, J., Packard, N. et. al. (Eds.), pp. 52-59, 2000.

[3] Todd, P. M. and Miller, G. F., *Biodiversity Through Sexual Selection*. In Artificial Life V, Langton, C. G. and Shimohara, K. (Eds.), Publisher: MIT Press, Cambridge, MA. USA, pp. 289-299, 1997.

[4] Mataric, M. J., *Kin Recognition, Similarity, and Group Behavior*. Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society, Boulder, CO, USA, pp. 705-710, 1993.

[5] Harvey, I. and Thompson, A., *Through the Labyrinth Evolution Finds a Way: a Silicon Ridge*. Proceedings of the First International Conference on Evolvable Systems: From Biology to Hardware - ICES96, Oct. 7-8, 1996, Tsukuba, Japan, Higuchi, T. and Iwata, M. (Eds.), Publisher: Springer-Verlag LNCS, ISBN: 3540631739, pp. 406-422, 1996.

[6] Watson, R. A. and Pollack, J. B., *Recombination Without Respect: Schema Combination and Disruption in Genetic Algorithm Crossover*. Proceedings of the Genetic and Evolutionary Computation Conference, Las Vegas, Nevada, USA, Whitley, D. (Ed.), Publisher: Morgan Kaufmann, ISBN: 1-55860-708-0, pp. 112-119, 2000.

[7] Tomassini, M., *A Survey of Genetic Algorithms*. In Annual Reviews of Computational Physics, World Scientific, pp. 87-118, 1995.

[8] Simões, E. D. V. and Dimond, K. R., "Embedding a Distributed Evolutionary System into a Population of Autonomous Mobile Robots", In IEEE International Conference on Systems, Man and Cybernetics, Tucson-AZ, USA, October, 2001, pp. 1069-1074, ISBN 0-7803-7089-9., 2001.