UMA NOVA ABORDAGEM PARA A EVOLUÇÃO DIFERENCIAL EM OTIMIZAÇÃO DISCRETA

RICARDO SÉRGIO PRADO*, RODRIGO CÉSAR PEDROSA SILVA†, FREDERICO GADELHA GUIMARÃES‡, ORIANE MAGELA NETO‡

*Instituto Federal de Minas Gerais, Campus Ouro Preto - IFMG-OP
Ouro Preto, Minas Gerais, Brasil

†Departamento de Computação
Universidade Federal de Ouro Preto - UFOP
Ouro Preto, Minas Gerais, Brasil

‡Departamento de Engenharia Elétrica
Universidade Federal de Minas Gerais - UFMG
Belo Horizonte, Minas Gerais, Brasil

Emails: rprado@cefetop.edu.br, rcpsilva@gmail.com, frederico.g.guimaraes@gmail.com, oriane@dee.ufmg.br

Abstract— The Differential Evolution (DE) algorithm is an important and powerful evolutionary optimizer in the context of continuous numerical optimization. More recently, some authors have proposed adaptations of its differential mutation mechanism to deal with combinatorial optimization, in particular permutation-based integer combinatorial problems. In this paper, we propose a novel and general DE-based metaheuristic that aims at preserving its interesting search mechanism for discrete domains, by defining the difference between two candidate solutions as a list of movements in the search space. In this way, we can produce a more meaningful and general differential mutation for the context of combinatorial optimization problems. The movements in the list can then be applied to other candidate solutions in the population as required by the differential mutation operator. We present results on instances of the Travelling Salesman Problem (TSP) and the N-Queen Problem (NQP) that suggest the adequacy of the proposed approach for discrete optimization.

Keywords— Differential Evolution, Metaheuristics, Combinatorial Optimization.

Resumo— O algoritmo Evolução Diferencial (DE) é um poderoso algoritmo de otimização evolucionário no contexto da otimização de sistemas a variáveis contínuas. Recentemente, tem sido propostas adaptações ao seu mecanismo de mutação diferencial para otimização de problemas combinatórios, em particular problemas combinatórios com permutação de números inteiros. Uma nova abordagem metaheurística para a Evolução Diferencial é proposta para a otimização discreta, definindo a diferença entre duas soluções candidatas como uma lista de movimentos no espaço de busca, e assim preservando seu interessante mecanismo de busca em domínios de busca discretos. Dessa maneira, obtém-se uma mutação diferencial mais significativa e genérica no contexto de problemas de otimização combinatória. Os movimento da lista são aplicados a outras soluções candidatas da população como é requerido pelo operador de mutação diferencial. Resultados são apresentados para instâncias do problema do caixeiro viajante (PCV) e o problema das N-Rainhas (PNR) que sugerem ser apropriada essa abordagem proposta para otimização discreta.

Keywords— Evolução Diferencial, Metaheurística, Otimização Combinatória.

1 Introdução

O grande desenvolvimento dos Algoritmos Evolucionários (AE) nas últimas décadas têm melhorado sua eficiência e aplicabilidade na solução de problemas de otimização nas diferentes áreas da engenharia e ciência da computação (Eiben, 2008). A família de AEs inclui, hoje em dia, uma vasta gama de técnicas de otimização e métodos heurísticos. Entre essas, a classe de algoritmos genéticos é, talvez, a mais popular e usada com sucesso para resolver várias classes de problemas de otimização discreta e contínua (Gen, 1997), (Goldberg, 1989). Essa família foi aumentada recentemente com novos membros, como por exemplo, os sistemas imunológicos artificiais (de Castro, 2002), Algoritmos de Estimação de Distribuição (Larranaga, 2002), Algoritmo de Evolução Diferencial (Price, 1999) e outros.

O algoritmo Evolução Diferencial (DE) foi primeiramente proposto em meados da década de 1990 para a otimização de sistemas a variáveis contínuas ((Price, 1999), (Price, 1997), e (Price, 2005)) e, hoje em dia, é um algoritmo de otimização importante e poderoso em otimização de sistemas mono-objetivo (Mezura-Montes, 2006) , (Price, 2005) e, mais recentemente, em problemas multiobjetivos (Batista, 2009), (Xue, 2003). Esse sucesso deve-se principalmente ao seu poderoso e ainda simples mecanismo de mutação diferencial, o qual usa a diferença entre dois vetores, escolhidos aleatoriamente das soluções candidatas, para produzir novas soluções. À medida que a população evolui, a direção de busca e o tamanho do passo na mutação mudam ao longo do tempo, auto-ajustando-se de acordo com a distribuição da população no espaço de busca. DE usa uma abordagem gulosa e , ao mesmo tempo, estocástica para solucionar o problema de otimização, combinando operadores aritméticos simples com as operações clássicas de cruzamento, mutação e seleção, de modo que a população inicialmente aleatória possa evoluir para uma população com soluções de alta qualidade.

O algoritmo DE enquadra-se à classe dos AE e, por essa razão, herda muito das terminologias dos AEs. Entretanto, a mutação diferencial não é di-

retamente baseada ou inspirada em processos naturais e sim, em heurísticas e argumentos puramente matemáticos adequados à otimização, ao invés de argumentos derivados da natureza. Todavia, a metaheurística da DE pode ser classificada como uma instância dos AEs, desde que ela usa operadores com heurísticas estocásticas inspirados em muitas idéias da computação natural e da adaptação natural para evoluir uma população inicial de soluções candidatas.

Embora o algoritmo tenha sido inicialmente desenvolvido para otimização de sistemas contínuos e não-lineares, algumas versões discretas do DE têm sido propostas na literatura, particularmente nos últimos anos. Por exemplo, (Qian, 2008) apresenta uma versão discreta do DE para permutação em problemas de sequenciamento da produção (Flow Shop scheduling). (Alatas, 2008) desenvolveu uma versão discreta do DE para minerar regras em problemas de mineração de dados, propondo arredondamentos e operadores de reparo específicos. (Onwubolu, 2006) tem mostrado também resultados interessantes para problemas de sequenciamento.

A principal característica da mutação diferencial é usar informações sobre o que é diferente nos indivíduos da população, de modo a produzir direções e o tamanho do passo da mutação. Entretanto, em problemas de natureza combinatória, a diferença vetorial não tem uma correspondência clara nas direções do espaço de busca. As subtrações de vetores, codificados com números inteiros, não geram soluções viáveis e nem representam direções significativas, uma vez que, os números nas soluções são rótulos arbitrários e não representam nenhuma quantidade numérica. Existem abordagens em que os elementos do vetor diferença são arredondados para o próximo número inteiro válido, de maneira a produzir soluções viáveis (Pan, 2009) (Qian, 2008). Essas abordagens não preservam, no domínio discreto, a principal característica que faz do DE um algoritmo de tamanho sucesso em otimização de sistemas no domínio contínuo: a adaptação da direção e dos tamanhos do passos na geração de novas soluções. Outras abordagens empregam esquemas mais sofisticados e alguns deles são revistos aqui.

Uma nova e genérica versão para a metaheurística da DE é proposta, e visa preservar seu interessante mecanismo de busca no domínio discreto, definindo-se a diferença entre as soluções candidatas como uma lista de movimentos no espaço de busca. Desta forma, é obtido um operador de mutação diferencial mais significativo e genérico no contexto dos problemas de natureza combinatória. Os movimentos na lista podem então ser aplicados a outras soluções candidatas da população como requerido pelo operador de mutação diferencial. À medida em que a população evolui, o número e a diversidade de movimentos da lista diminuem, uma vez que, a mutação diferencial auto-adapta-se de acordo com a distribuição da população no espaço de busca, da mesma maneira que o DE opera no domínio contínuo.

Resultados apresentados, para instâncias do Problema do Caixeiro Viajante (PCV) e o Problema das N Rainhas (PNR), sugerem adequada a abordagem aqui proposta, quando comparada com outras apresentadas na literatura. Este artigo é organizado como segue: A seção II apresenta o algoritmo DE básico no domínio contínuo; a Seção III revê outras versões discretas do DE encontradas na literatura e apresenta a versão aqui proposta; A Seção IV apresenta os resultados obtidos com instâncias PCV e PNR e a Seção V conclui o artigo com uma breve discussão e considerações finais.

2 Evolução Diferencial para Otimização Contínua

Nesta seção, as bases do algoritmo DE para solução de problemas de otimização de sistemas contínuos é brevemente revista (Price, 1997), (Price, 1999), (Price, 2005). O problema de otimização de sistemas contínuos não-lineares pode ser generalizado como:

$$x^* = \arg \min_x f(x)$$

Onde $f(.) : \mathcal{S} \subset \mathbb{R}^n \rightarrow \mathbb{R}$ é a função de custo (ou função-objetivo) a ser minimizada. A região de domínio do problema é usualmente descrito como:

$$\mathcal{S} = \left\{x \in \mathbb{R}^n : x_k^{min} \leq x_k \leq x_k^{max}, k = 1, \ldots, n\right\}$$

O algoritmo DE, assim como outros algoritmos evolucionários, utiliza uma população de soluções candidatas randomicamente geradas, dentro dessa região de domínio, para minimizar a função de custo. Cada solução candidata, ou indivíduo, é representada por um vetor do espaço Real:

$$x_{t,i} = \begin{bmatrix} x_{t,i,1} \\ x_{t,i,2} \\ \vdots \\ x_{t,i,n} \end{bmatrix}$$

Tal que, $t = 1, \ldots, T$ indica a geração que o indivíduo pertence, $i = 1, \ldots, \mu$ indica o indivíduo na população e o terceiro índice $j = 1, \ldots, n$ representa o índice da variável ou parâmetro.

Para gerar novas populações, DE combina o típico operador de cruzamento com seu peculiar operador de mutação: o operador de mutação diferencial. A mutação diferencial usa a diferença entre pares de vetores escolhidos randomicamente dentre as soluções candidatas da população e o vetor resultante dessa diferença é adicionado a uma terceira solução também escolhida aleatoriamente. A mutação diferencial representa uma perturbação que é aplicada ao terceiro indivíduo de modo a produzir soluções mutantes de acordo com a expressão:

$$v_{t,i} = x_{t,r_1} + F(x_{t,r_2} - x_{t,r_3}) \qquad (1)$$

Onde os índices r_1, r_2, r_3 representam indivíduos randômicos e mutuamente distintos, escolhidos da

população. F, um peso escalar aplicado ao vetor diferença, é um parâmetro do algoritmo. A solução x_{t,r_1}, na qual a mutação diferencial é aplicada, é denominada solução base ou vetor base. Para cada indivíduo $x_{t,i}$ da população é gerado um vetor mutante correspondente.

A solução $v_{t,i}$ é recombinada com a solução corrente $x_{t,i}$ para produzir uma solução experimental $u_{t,i}$ que, por sua vez, competirá com a solução corrente. Na versão básica do algoritmo DE, uma recombinação discreta com probabilidade $P \in [0,1]$ é adotada, mas outros operadores de recombinação podem também ser utilizados.

Finalmente, os valores da função objetivo da solução experimental, $u_{t,i}$, e da solução corrente, $x_{t,i}$, são comparados para decidir qual solução irá sobreviver na próxima geração. Se a solução experimental for melhor que, ou igual, a solução corrente, ela substitui a solução corrente na próxima geração, caso contrário, a solução corrente é preservada e a solução experimental é descartada; Esse procedimento pode ser descrito como:

$$x_{t+1,i} = \begin{cases} u_{t,i}, & \text{se} f(u_{t,i}) \leq f(x_{t,i}) \\ x_{t,i}, & \text{senão} \end{cases}$$

É possível demonstrar que a mutação diferencial é rotacionalmente invariante ao sistema de coordenadas (Price, 2005). Além disso, as direções e tamanho do passo da mutação se adaptam automaticamente de acordo com a distribuição espacial da população, veja exemplos em (Batista, 2009)e (Price, 2005). Outras variações desse algoritmo básico são possíveis e muitos estão disponíveis na literatura (Chakraborty, 2008), (Mezura-Montes, 2006). Basicamente, essas variações estão na escolha do número de vetores diferença utilizados na mutação, na escolha do vetor base, na seleção do fator escalar F e no modo da recombinação dos operadores.

3 Evolução Diferencial para Otimização Discreta

O algoritmo DE é, originalmente, aplicável à otimização contínua devido ao seu mecanismo de busca ser baseado nas perturbações criadas a partir do vetor diferença. Essa definição da mutação diferencial está estritamente ligada ao espaço Euclidiano, uma vez que os vetores são variáveis reais. Uma diferença análoga não é diretamente obtida no domínio das variáveis discretas, o que faz com que, uma extensão das técnicas do DE para esse contexto, nada simples à primeira vista. Nesta seção são revistos alguns trabalhos sobre o algoritmo DE no domínio discreto presentes na literatura e, então, uma nova versão é proposta.

3.1 Indexação por Posição Relativa

Nos últimos anos, pesquisas têm sido feitas com o intuito de adaptar o algoritmo DE à otimização de sistemas discretos, através de propostas análogas ao operador de mutação diferencial no contexto das variáveis contínuas (Alatas, 2008), (Onwubolu, 2008), (Onwubolu, 2006), (Pan, 2009), (Price, 2005), (Qian, 2008). Extensivas revisões de métodos de evolução diferencial aplicado à otimização combinatória têm sido também publicadas, veja em (Onwubolu, 2009).

Uma abordagem proposta na literatura é a Indexação por Posição Relativa (IPR) utilizada para a evolução diferencial no domínio discreto (Pan, 2009) (Qian, 2008). Nesta abordagem, aplicável somente em problemas baseados em permutações, os elementos dos vetores são transformados em números reais, dentro do intervalo $[0,1]$. Assim, a mutação diferencial no domínio contínuo pode ser empregada e, então, os elementos dos vetores são convertidos de volta para o domínio dos números inteiros usando uma indexação relativa. O primeiro passo é normalizar os vetores dividindo-se cada um de seus elementos pelo maior dentre eles. Após essa normalização, a equação de mutação diferencial, equação 1, é aplicada como no caso do domínio contínuo. O vetor mutante resultante não foi gerado por uma permutação, portanto, para convertê-lo para o domínio dos inteiros, os elementos no vetor são ordenados de acordo com seus valores. Assim, o menor número real obtido é relacionado ao menor valor inteiro, o próximo menor valor real ao próximo valor inteiro e assim sucessivamente, até todos os elementos no vetor serem convertidos para o domínio dos números inteiros.

3.2 Transformação Forward Backward

Outra abordagem interessante é proposta em (Onwubolu, 2008). Ela é conhecida como Transformação "Para Frente e Para trás" (*Forward Backward Transformation*). Nesta abordagem, os vetores inteiros são transformados em valores reais usando a transformação para frente, dada por:

$$x_{i_f} = -1 + (1+\epsilon)x_i$$

Onde ϵ é um número muito pequeno. Assim, como na abordagem anterior, a equação 1 pode ser aplicada. A conversão de volta é realizada usando-se a transformação para trás, dada por:

$$x_i = round[(1+x_{i_f})(2-\epsilon)]$$

Embora a transformação para trás produza, efetivamente, valores inteiros, eles são geralmente soluções inválidas, isto é, são valores inteiros que não representam nenhuma permutação válida. Essas soluções, portanto, devem ser reparadas por algum método apropriado.

3.3 Matriz de Permutação

A abordagem por Matriz de Permutação (MP) foi proposta por Price e Storn como um operador de mutação diferencial (Price, 2005). Uma matriz de permutação P é a matriz que mapeia a permutação de um dado vetor em outro. Sejam $x_{t,r2}$ e $x_{t,r3}$ dois vetores randomicamente escolhidos da população. A matriz de

permutação que mapeia $x_{t,r2}$ em $x_{t,r3}$ satisfaz a relação:

$$x_{t,r2} = Px_{t,r3}$$

A matriz de permutação é dada pela permutação das linhas da correspondente matriz Identidade, realizando assim a permutação necessária dos elementos de $x_{t,r3}$. Pode-se dizer, portanto, que a matriz de permutação "leva" $x_{t,r3}$ à $x_{t,r2}$. No contexto da evolução diferencial, essa matriz é considerada a "diferença" obtida de duas soluções candidatas. A equação análoga à equação da mutação diferencial, 1, é, então, escrita como:

$$v_{t,i} = P\eta.x_{t,r1}$$

onde $P\eta$ é a matriz de permutação modificada pelo parâmetro escalar η, significando aqui, uma probabilidade de se utilizar uma parcela da permutação representada pela matriz de permutação original P. Portanto, o método MP aplica algumas permutações ao vetor base, randomicamente selecionadas do conjunto de permutações de P. Esta abordagem é tão somente aplicável aos problemas combinatórios baseados em permutações.

3.4 Algoritmo Proposto para Evolução Diferencial Discreta: Abordagem por Lista de Movimentos

Em problemas combinatórios, envolvendo permutações de vetores de inteiros, a aplicação direta do mecanismo de mutação diferencial não é a melhor estratégia, pois a subtração de tais vetores, não levaria a nenhuma direção significativa no espaço de busca. Além disso, após a multiplicação pelo escalar F, e adicionando esse resultado ao vetor base, as soluções obtidas seriam geralmente inviáveis, como visto nas seções anteriores. Este problema surge porque esses vetores de inteiros não representam quantidades numéricas, mas sim, rótulos identificadores arbitrários.

De modo a conceber uma metaheurística para o método de otimização da evolução diferencial discreta, a diferença entre dois vetores distintos deve ser definida no espaço das variáveis discretas. A idéia, por trás desta abordagem, é definir a diferença entre duas soluções candidatas como sendo uma lista de movimentos no espaço de busca. Essa lista é definida como:

Definição 1: A lista de movimentos M_{ij} é a lista contendo uma sequência de movimentos válidos m_k tais que, a aplicação destes movimentos à solução $s_i \in \mathcal{S}$ leva à solução $s_j \in \mathcal{S}$.

Desta forma, a "diferença" entre duas soluções é definida como sendo esta lista de movimentos:

$$M_{ij} = s_i \ominus s_j \tag{2}$$

onde $\ominus$ é um operador de subtração binário especial que retorna a lista de movimentos M_{ij} que representa o "caminho" da solução s_i até a solução s_j. Esta lista, de algum modo, captura a diferença entre essas duas soluções.

A multiplicação da lista de movimentos por um escalar deve ser também definida. Duas alternativas são dadas:

Definição 2: A multiplicação da lista de movimentos, M_{ij}, por um escalar $F \in [0, 1]$, retorna a lista M'_{ij} com os **primeiros** $\lceil F \times |M_{ij}| \rceil$ movimentos de M_{ij}, onde $|M_{ij}|$ é o tamanho da lista

Definição 3: A multiplicação da lista de movimentos M_{ij} por um escalar $F \in [0, 1]$, retorna a lista M'_{ij} com os $\lceil F \times |M_{ij}| \rceil$ movimentos escolhidos **randomicamente** de M_{ij}, onde $|M_{ij}|$ é o tamanho da lista.

Portanto, a multiplicação da lista de movimentos por um escalar pode ser denotado, usando o operador de multiplicação binário especial $\otimes$, por:

$$M'_{ij} = F \otimes M_{ij} \tag{3}$$

Finalmente, a aplicação da lista de movimentos a uma dada solução é definida como se segue:

Definição 4: A aplicação da sequência de movimentos da lista M_{ij} na solução s_k, retorna uma nova solução s'_k

A definição 4 pode ser escrita como:

$$s'_k = s_k \oplus M_{ij} \tag{4}$$

Com as definições acima, pode-se escrever a equação do vetor mutante como:

$$\begin{aligned} v_i &= x_1 \oplus F \otimes (x_2 \ominus x_3) \\ v_i &= x_1 \oplus F \otimes M_{23} \\ v_i &= x_1 \oplus M'_{23} \end{aligned}$$

que é a versão discreta proposta para a equação 1.

3.5 Exemplo Ilustrativo

O PCV é um exemplo típico de problema de natureza combinatório com permutação. Se as cidades são rotuladas com números inteiros, rotas válidas são apenas permutações destes números. Para uma instância com 9 cidades, as 3 rotas seguintes podem ser representas por vetores como mostrado na figura 1.

Pela definição 1, a equação 2 pode ser escrita como:

$$M_{23} = x_2 \ominus x_3$$

A lista M_{23} é consruída iterativamente encontrando-se movimentos que levem a solução x_3 à solução x_2. A figura 2 mostra o primeiro passo para a construção da lista. Supondo que os vetores iniciam-se com o índice zero, trocando-se as cidades de índices 1 e 2 em x_3, leva a solução x_3 aproximar-se da solução x_2

O próximo movimento, mostrado na figura 3, é trocar as cidades nas posições 2 e 3

Procedendo-se assim, uma lista, composta por pares de índices, é obtida como mostrado abaixo:

$$M_{23} = (1, 2); (2, 3); (3, 4); (4, 6); (5, 7); (6, 8)$$

A aplicação desses movimentos da lista ao vetor x_3, levaria à solução x_2.

Figura 1: Três rotas possíveis para uma Instância PCV de 9 cidades e respectiva representação vetorial

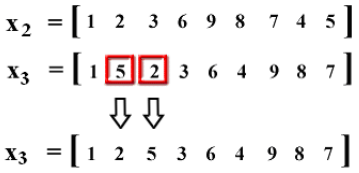

Figura 2: Primeira troca no vetor x_3

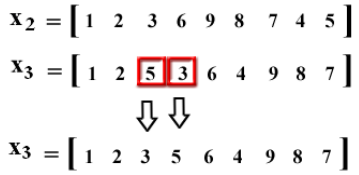

Figura 3: Segunda atroca no vetor x_3

Entretanto, de acordo com a definição 3, é necessário multiplicar essa lista pelo escalar F e, então, aplicar o resultado a uma outra solução. Usando a equação 3 e supondo $F = 0,6$, a lista M'_{23}, contendo 4 movimentos aleatoriamente escolhidos da lista original, é obtida como mostrado abaixo:

$$M'_{23} = (2,3); (5,7); (1,2); (6,8)$$

A aplicação desta sequência de movimentos ao vetor x_1, o vetor base, gera o vetor mutante v_i, de acordo com a definição 4 e a equação 4. As figuras de 4 a 7 ilustram essas etapas.

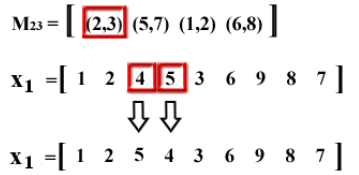

Figura 4: Aplicação do primeiro movimento da lista

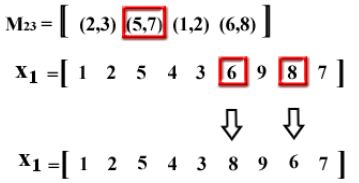

Figura 5: Aplicação do Segundo movimento da lista

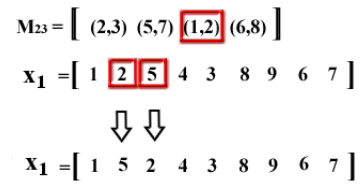

Figura 6: Aplicação do terceiro movimento da lista

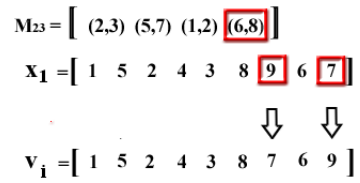

Figura 7: Aplicação do quarto movimento da lista e o vetor mutante resultante

Esse simples exemplo mostra como essa nova abordagem para a mutação diferencial age para produzir uma versão discreta da metaheurística do DE. Embora o exemplo tenha sido aplicado a um problema de permutação, as definições são gerais e podem ser usadas em qualquer problema de otimização de natureza combinatória pois, movimentos específicos geram listas de movimentos específicas de acordo com o tipo de problema.

4 Resultados

Nesta seção são comparados os desempenhos encontrados para abordagem IPR, apresentada na seção 3.1, e a abordagem proposta, utilizando as definições 2 e 3. Primeiramente, os problemas utilizados para teste são descritos e, então, os resultados são apresentados e discutidos.

4.1 Problema do Caixeiro Viajante (PCV)

O PCV consiste em encontrar o caminho de menor custo para um caixeiro viajante que tem que percorrer um conjunto de cidades, passando uma única vez em cada uma delas e retornar a cidade de origem. A formulação do problema é feita considerando N o número de cidades e uma matriz de distâncias D com d_{ij} elementos representando as distâncias entre as cidades *i* e *j*. Os percursos são representados por vetores de permutação, de modo que, cada permutação representa

um caminho percorrido.

4.2 Problema das N Rainhas (PNR)

O PNR foi proposto originalmente pelo enxadrista Max Friedrich William Bezzel em 1848 para um tabuleiro de xadrez com oito rainhas. O problema proposto era colocar as oito rainhas no tabuleiro de tal modo que, nenhuma rainha seria atacada por outra, respeitando-se as regras de movimento de uma rainha em um jogo de xadrez. Mais tarde, o problema foi generalizado para N rainhas em um tabuleiro de NxN casas. O problema tem dificuldade crescente com o número de rainhas e, conseqüentemente, com o tamanho do tabuleiro e é um problema de natureza combinatorial.

4.3 Resultados Obtidos

Para o PCV foram executados 30 testes para cada instância, utilizando como critério de parada o número máximo de 2500 gerações. Os parâmetros iniciais utilizados foram: número de indivíduos da população $\mu = 50$, $F \in [0,4; 1,0]$ e $P\eta = 0,6$. Uma busca local foi aplicada a solução experimental , $u_{t,i}$, antes da fase de seleção. Todas as abordagens empregaram o critério de Primeira Melhora para a busca local, de modo a acelerar a convergência do algoritmo.

A tabela 1 apresenta os resultados obtidos para algumas instâncias PCV conhecidas, disponíveis em http://www.tsp.gatech.edu/index.html, utilizadas como teste. Na comparação, vê-se que o DE proposto, utilizando a definição 2, converge com um número menor de gerações e com variações pequenas mostrando ser mais robusto que a abordagem IPR.

Para o PNR foram adotados os mesmos parâmetros e condições iniciais anteriores. Em nenhuma das abordagens foi utilizada busca local, com o intuito de comparar tão somente o comportamento e diferenças nos métodos de mutação diferencial empregados.

A tabela 2 apresenta alguns resultados para este problema. Vê-se que, em todos os testes realizados, a mutação diferencial proposta utilizando a definição 3 foi a que apresentou melhores resultados, mesmo quando seguido de perto pela abordagem MP. Por outro lado, o DE utilizando a definição 2 tem seu desempenho degradado quando usado em instâncias maiores. De todas as abordagens, a IPR é a que apresenta os piores resultados. Isto comprova que, o uso de operadores de arredondamento no reparo de soluções inválidas não é uma boa estratégia para utilizar o DE em problemas combinatórios com permutação.

5 Conclusão

Neste artigo, uma nova metaheurística para otimização combinatória foi apresentada para o algoritmo de evolução diferencial no domínio discreto. A abordagem proposta visa preservar o tão bem sucedido mecanismo de busca da mutação diferencial no domínio contínuo, ao definir a diferença entre duas soluções candidatas como sendo uma lista de movimentos no espaço de busca.

Com as definições apresentadas, obteve-se um operador de mutação diferencial discreto, genérico e significativo para otimização de problemas não somente de natureza combinatórios, mas também de problemas de otimização discreta em geral.

Testes comparativos, entre essa nova abordagem e outras apresentadas na literatura, foram realizados para instâncias do PCV e PNR. Os resultados obtidos sugerem adequadas as definições aqui propostas para o operador de mutação diferencial.

Agradecimentos

Este trabalho foi financiado pelo CNPq e pela FAPEMIG (APQ-02170-09).

Referências

Alatas, B., A. E. . K. A. (2008). *MODENAR: Multi-objective differential evolution algorithm for mining numeric association rules*, Applied Soft Computing, 8(1), 646-656, January.

Batista, L. S. Guimarães, F. G. . R. J. A. (2009). *A differential mutation operator for the archive population of multi-objective evolutionary algorithms*, In Proceedings of the IEEE Congress on Evolutionary Computation (CEC) (pp. 1108-1115). Piscataway: IEEE Press.

Chakraborty, U. K. E. (2008). *Advances in Differential Evolution*, ser. Studies in Computational Intelligence, vol. 143. Berlin: Springer-Verlag.

de Castro, L. N., . T. J. (2002). *Artificial Immune Systems: A New Computational Intelligence Approach*, Springer.

Eiben, A.E. & Smith, J. (2008). *Introduction to Evolutionary Computing*, ser. Natural Computing Series.Berlin:Springer-Verlag.

Gen, M. & Cheng, R. (1997). *Genetic Algorithms and Engineering Design.*, 1st ed. Wiley-Interscience.

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley.

Larranaga, P. & Lozano, J. A. (2002). *Estimation of Distribution Algorithms: A New Tool of Evolutionary Computation.*, ser. Genetic Algorithms and Evolutionary Computatio Netherlands: Kluwer.

Mezura-Montes, E., V.-R. J. . C. C. A. C. (2006). *A comparative study of differential evolution variants for global optimization*, In Proceedings of the 8th Annual Conference on Genetic and

Tabela 1: Média e Desvio Padrão do Número de Gerações Necessárias para a Convergência de Algumas Instâncias PCV

Instancia PCV	Melhor Conhecido	DE usando IPR		DE usando Definição 2	
		Encontrado	Geração	Encontrado	Geração
br17	39	39±0,0	1,03±0,18	39±0,0	0,37±0,96
ftv33	1286	1286±0.0	134,0±113,8	1286±0.0	116,4±61,8
ftv35	1473	1473±0	292,0±256,6	1473,3±0,7	142,3±70,5
ftv38	1530	1530,2±0,6	538,1±363,8	1530,6±0,9	221,5±120,6
ftv44	1613	1616,7±5,8	1414,4±636,5	1618,6±9,4	594,8±235,4

Tabela 2: Média e Desvio Padrão do Número de Gerações Necessárias para Encontrar o Mínimo Global de Algumas Instâncias PNR

N	DE usando IPR	DE usando MP	DE usando Def. 2	DE usando Def. 3
10	6,63±6,98	7,40±4,95	6,73±5,58	5,93±4,18
12	31,50±32,55	30,40±17,15	28,77±20,15	25,3±18,57
14	131,90±102,82	82,43±66,33	87,90±61,71	75,17±61,71
16	481,10±343,68	237,46±171,00	338,43±240,38	223,80±135,86
18	1270,43±826,37	700,07±356,47	815,83±460,47	527,03±284,04
20	3330,37±2706,27	1842,07±670,88	2594,53±1476,67	1465,30±764,51
22	11490,37±7834,16	2693,33±1101,54	6508,87±3640,53	2494,60±1251,72

Evolutionary Computation (ACM GECCO) (pp. 485-492). ACM Press.

Onwubolu, G. C. (2008). *Design of hybrid differential evolution and group method of data handling networks for modeling and prediction*, Information Sciences, 178, 3616-3634.

Onwubolu, G. C., . D. D. (2006). *Scheduling flow shops using differential evolution algorithm*, European Journal of Operational Research, 171(2), 674-692.

Onwubolu, G. C., . D. D. E. (2009). *Differential Evolution: A Handbook for Global Permutation-Based Combinatorial Optimization,*, ser. Studies in Computational Intelligence, vol. 175. Berlin: Springer.

Pan, Q. K., W. L. . Q. B. (2009). *A novel differential evolution algorithm for bi-criteria no-wait flow shop scheduling problems*, Computers & Operations Research, 36(8), 2498-2511.

Price, K. V. (1999). *An Introduction to differential evolution. e*, In D.Corne, M. Dorigo, F. Glover (Ed), New Ideas in Optimization, (pp.79-108) ser. Advanced Topics in Computer Scienc McGraw-Hill.

Price, K. V., . S. R. M. (1997). *Differential Evolution - a simple and efficient heuristic for global optimization over continuous spaces*, Journal of Global Optimization, 11(4), 341-359, December.

Price, K. V., S. R. M. . L. J. A. (2005). *Differential Evolution: A Pratical Approach to Global Optimization*, 1st ed., ser. Natural Computing Series, Berlin: Springer-Verlag.

Qian, B., W. L. H. R. W. W.-L. H. D.-X. . W. X. (2008). *A hybrid differential evolution method for permutation flow-shop scheduling*, The International Journal of Advanced Manufacturing Technology, 38, 757-777.

Xue, F., S. A. C. . G. R. J. (2003). *Pareto-based multiobjective differential evolution.*, In Proceedings of the IEEE Congress on Evolutionary Computation (CEC), vol. 2 (pp. 862-869). Piscataway: IEEE Press.