Particle Swarm Optimization (PSO)

Marcone Jamilson Freitas Souza
Departamento de Computação
Universidade Federal de Ouro Preto

*Baseado no material da Profa.
Estéfane G. M. de Lacerda
(www.dca.ufrn.br/~estefane/metaheuristicas/pso.pdf)

Souza, Marcone J. F. Particle Swarm Optimization. Notas de aula de Técnicas Metaheurísticas para Otimização Combinatória. Departamento de Computação, Universidade Federal de Ouro Preto, Ouro Preto, 2021. Disponível em http://www.decom.ufop.br/prof/marcone/Disciplinas/InteligenciaComputacional/PSO.pptx

Particle Swarm Optimization

- Otimização por nuvem de partículas
- Desenvolvido pelo psicólogo social James Kennedy e o engenheiro eletricista Russel Eberhart, em 1995
- Originalmente desenvolvido para resolver problemas de otimização com variáveis contínuas

Princípio

- Simula o comportamento social de um bando de pássaros à procura de um alvo (alimento, local para pouso, proteção contra predadores etc.)
- Estudos apontam que o bando encontra seu alvo por meio de um esforço conjunto
- Isto sugere que eles compartilham informações

Princípio

Princípio

- Algoritmo populacional
  - A população é chamada de “nuvem” ou “enxame”
  - Os indivíduos são chamados de “partículas”
- O enxame evolui por meio de cooperação e competição entre seus membros
- As partículas se beneficiam da sua própria experiência e da experiência de outros membros do enxame durante a busca pelo alvo

Notação

$$\mathbf{x}_i = \begin{bmatrix} x_{i,1} \\ x_{i,2} \\ \vdots \\ x_{i,n} \end{bmatrix}, \quad \text{posição da partícula } i \text{ (coordenadas)}$$

$$\mathbf{v}_i = \begin{bmatrix} v_{i,1} \\ v_{i,2} \\ \vdots \\ v_{i,n} \end{bmatrix}, \quad \text{velocidade da partícula } i$$

$f(\mathbf{x}_i)$, aptidão da partícula i

m, tamanho da população de partículas

Notação

$\mathbf{p}_i$	$pbest_i$ (personal best) a melhor posição encontrada pela partícula i
$\mathbf{g}$	*gbest* (global best) a melhor posição encontrada por todas as partículas
c_1, c_2	parâmetros cognitivo e social (também chamados de taxas de aprendizado)
w	ponderação de inércia
r_{1j}, r_{2j}	números aleatórios entre 0 e 1

Atualização de posição e velocidade

Atualização de velocidade na iteração k

$$v_{ij}^{k+1} = w v_{ij}^{k} + c_1 r_{1j}(p_{ij}^{k} - x_{ij}^{k}) + c_2 r_{2j}(g_{j}^{k} - x_{ij}^{k})$$

para $i = 1, \ldots, m$ e $j = 1, \ldots, n$.

Atualização de posição na iteração k

$$\mathbf{x}_i^{k+1} = \mathbf{x}_i^{k} + \mathbf{v}_i^{k+1}$$

para $i = 1, \ldots, m$

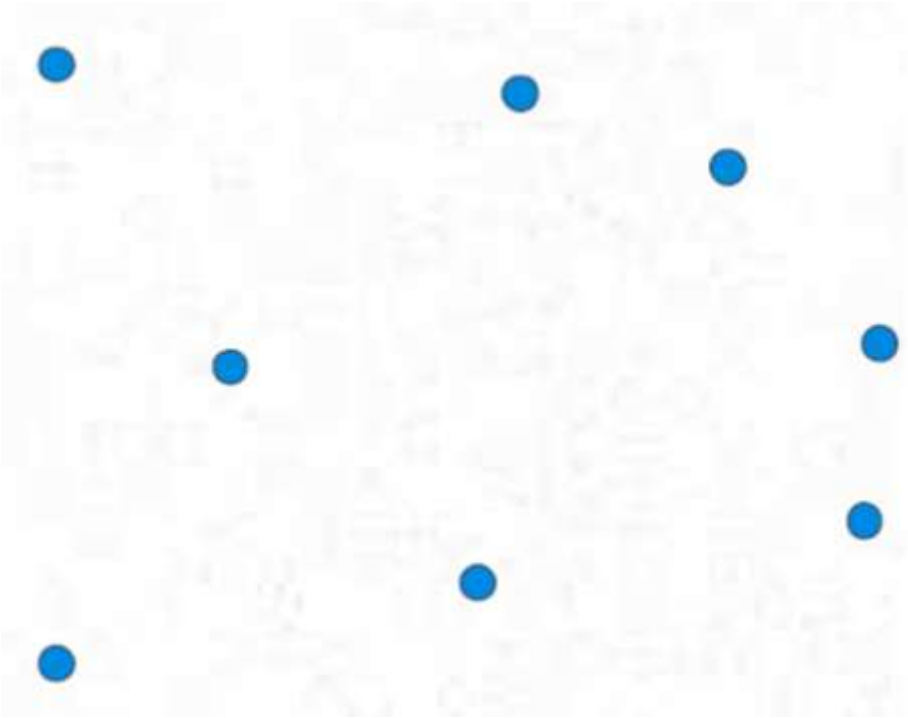
Componentes Cognitivo e Social

- (p_i^k - x_i^k) é o componente cognitivo: representa a experiência individual da partícula *i* até a *k*-ésima geração
- (g^k – x_i^k) é o componente social: representa a experiência da nuvem de partículas até a geração atual

Atualização de posição e velocidade

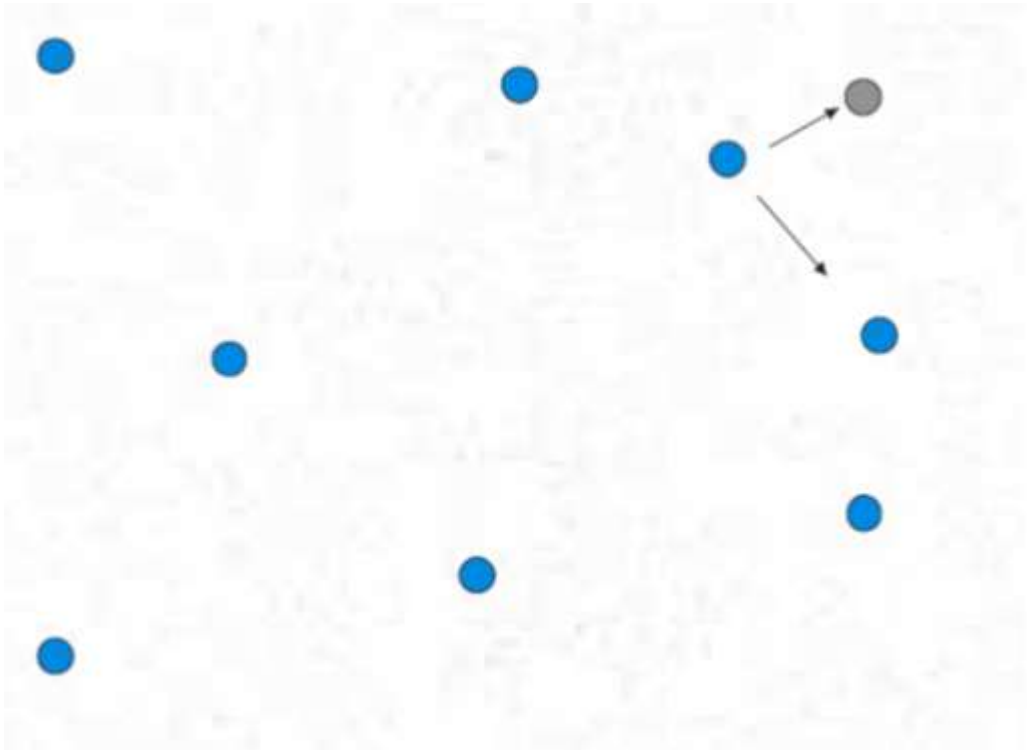
- A nova velocidade da partícula é influenciada por três fatores:
  - velocidade anterior
  - distância entre sua posição atual e melhor posição alcançada até então
  - distância entre sua posição atual e a melhor posição do grupo
- A partir do cálculo dessa nova velocidade, a partícula "voa" para sua nova posição

Atualização da nova posição



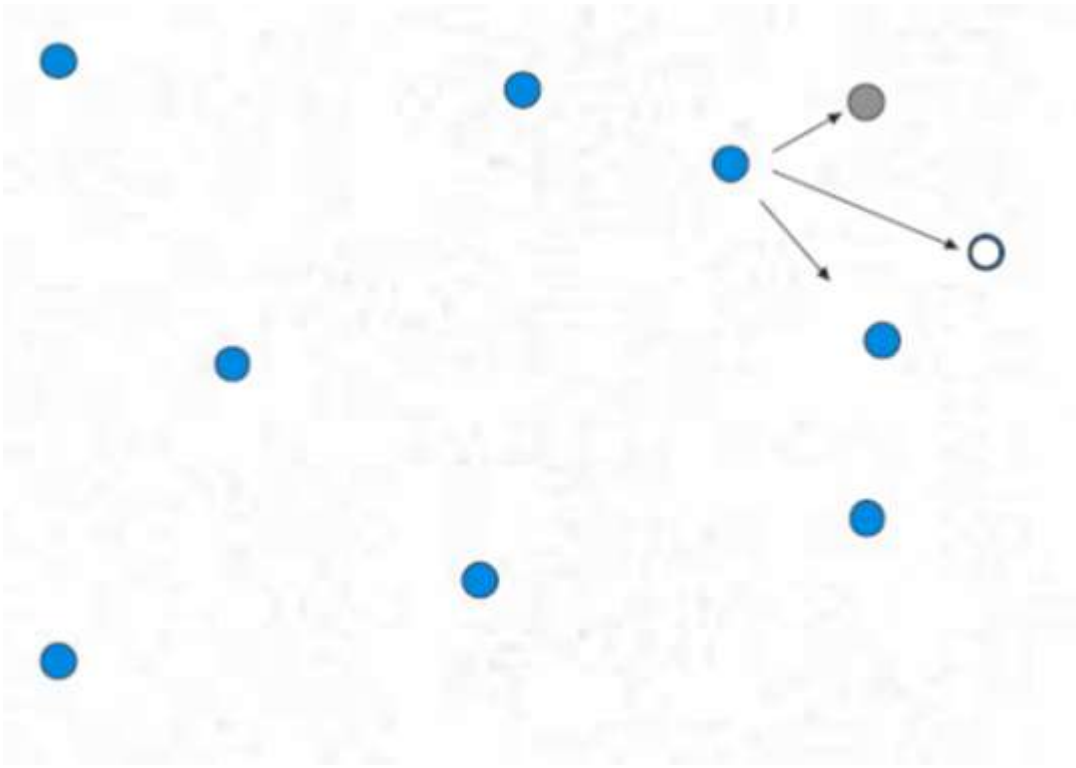
Inicialmente, diversas partículas são espalhadas aleatoriamente no espaço de busca

Atualização da nova posição



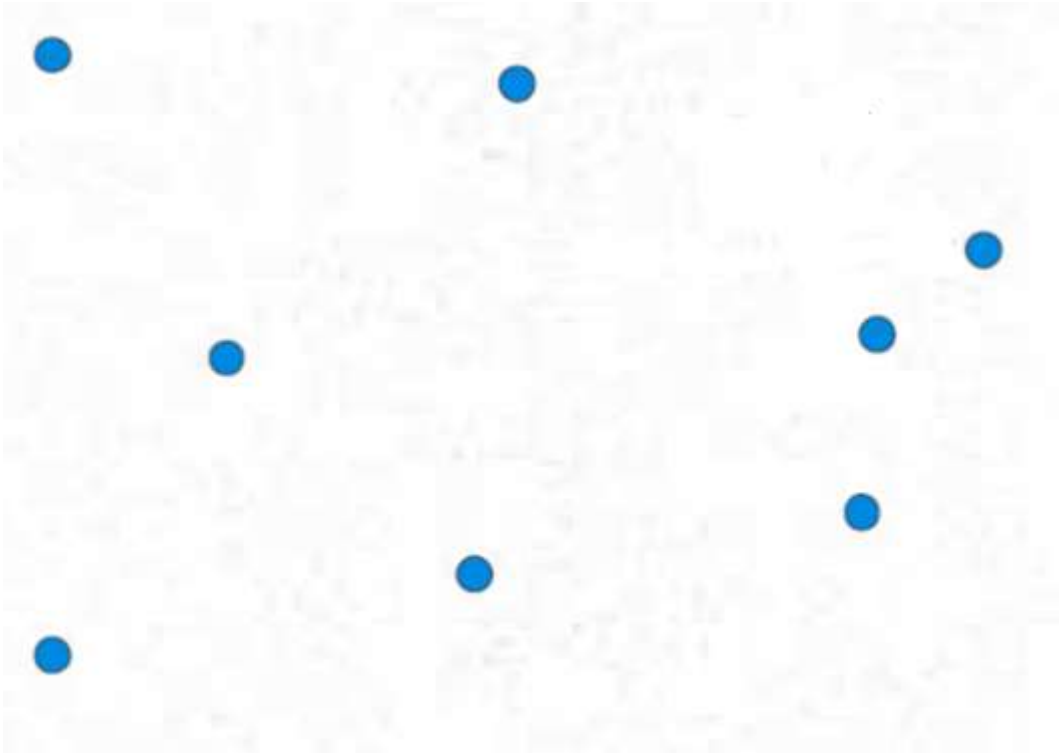
Cada partícula utiliza sua melhor posição no passado (em cinza) e sua melhor posição na vizinhança

Atualização da nova posição



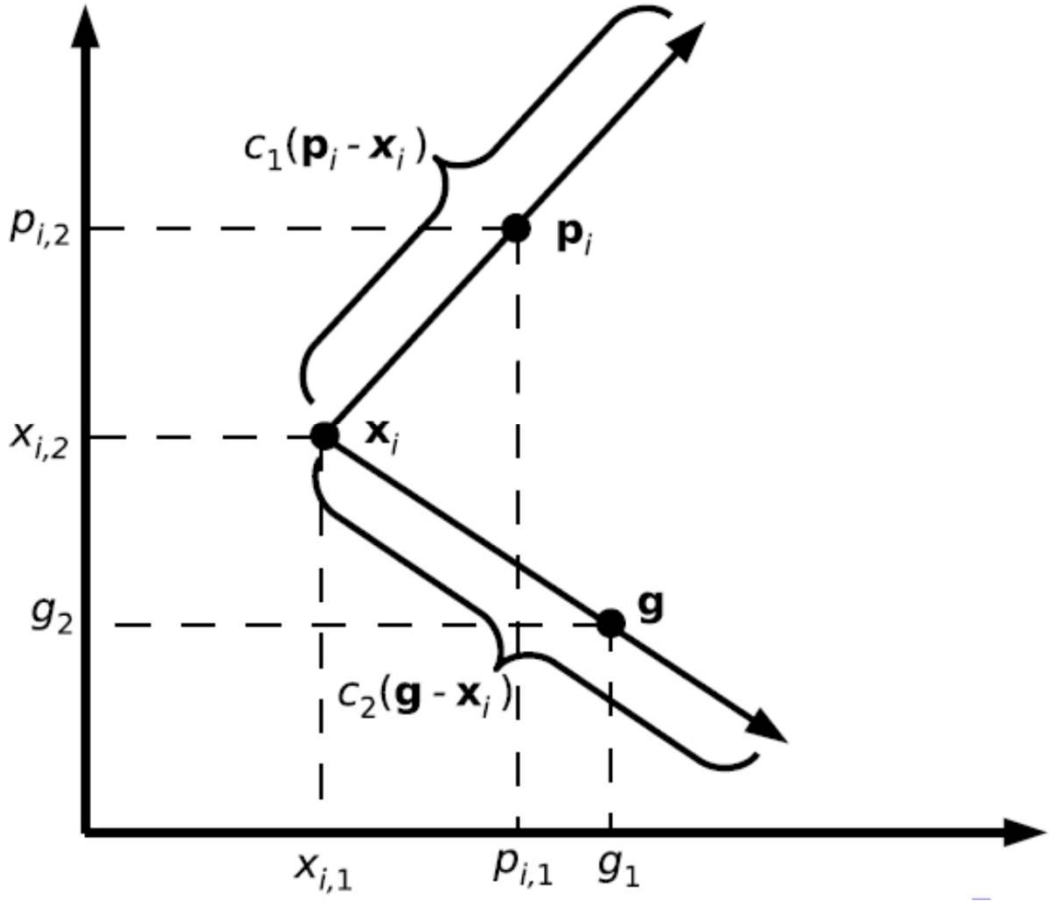
A partícula se move para a nova posição, pela combinação linear desses dois vetores, com pesos diferentes.

Atualização da nova posição

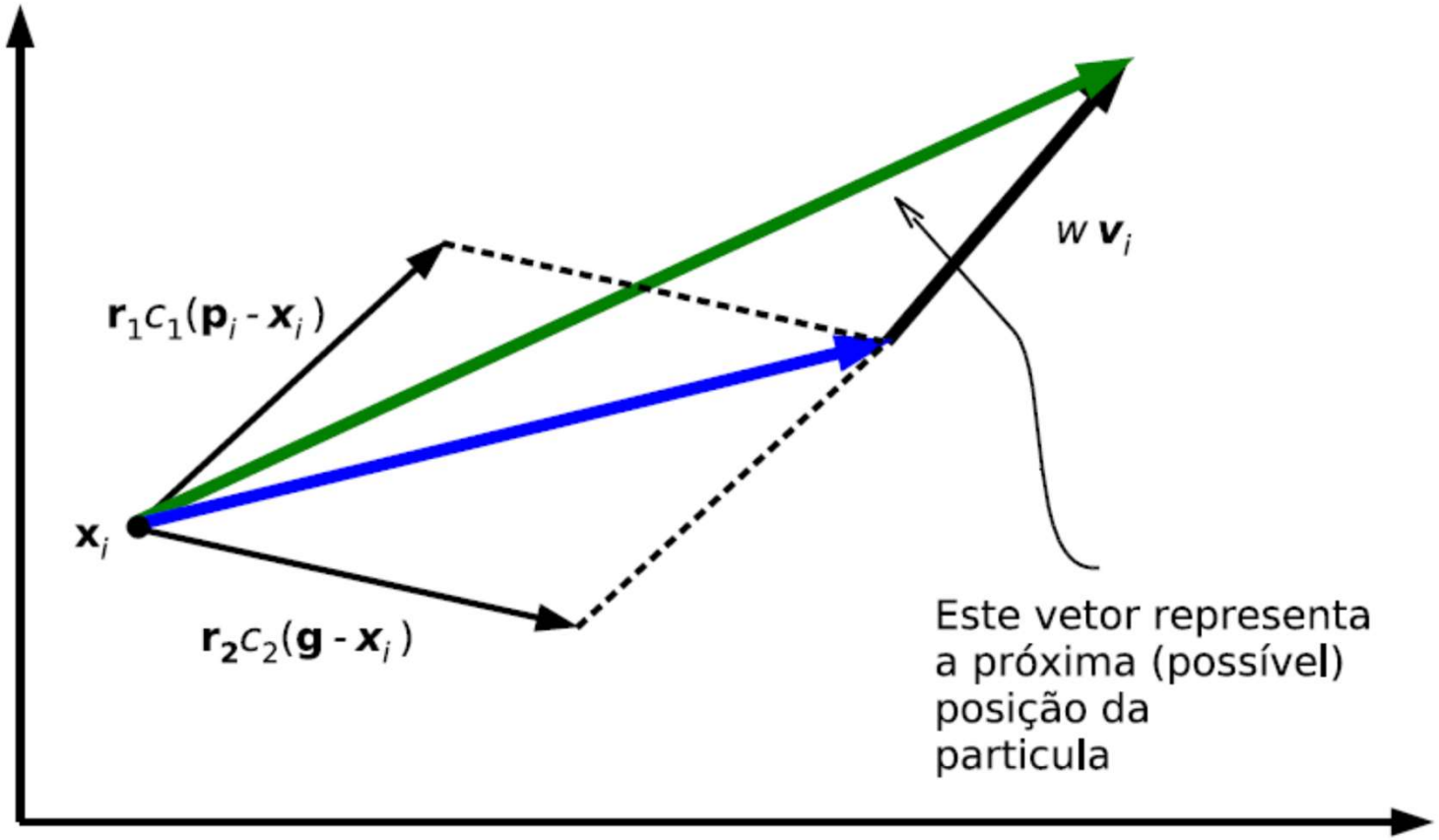


Nova posição para a partícula

Interpretação Geométrica



Interpretação Geométrica



Diversificação versus intensificação

- O algoritmo PSO fornece um mecanismo bem balanceado entre diversificação e intensificação:

$$v_{ij}^{k+1} = \underbrace{wv_{ij}^{k}}_{\text{diversificação}} + \underbrace{c_1 r_{1j}(p_{ij}^{k} - x_{ij}^{k}) + c_2 r_{2j}(g_{j}^{k} - x_{ij}^{k})}_{\text{intensificação}}$$

Algoritmo PSO

inicialize a nuvem de partículas
repita
  para $i = 1$ **até** m
    se $f(\mathbf{x}_i) < f(\mathbf{p}_i)$ **então**
      $\mathbf{p}_i = \mathbf{x}_i$
      se $f(\mathbf{x}_i) < f(\mathbf{g})$ **então**
        $\mathbf{g} = \mathbf{x}_i$
      fim se
    fim se
    para $j = 1$ **até** n
      $r_1 = \text{rand}()$, $r_2 = \text{rand}()$
      $v_{ij} = wv_{ij} + c_1 r_1 (p_i - x_{ij}) + c_2 r_2 (g_j - x_{ij})$
    fim para
    $\mathbf{x}_i = \mathbf{x}_i + \mathbf{v}_i$
  fim para
até satisfazer o critério de parada

Algoritmo PSO

Passo 0 (Definição das variáveis):

P o tamanho da população do PSO.

$PSO[i]$ a posição da i-ésima partícula da população do PSO, que representa uma solução candidata para o problema.

$fitness[i]$ o custo da função da i-ésima partícula.

$V[i]$ a velocidade da partícula.

G_{best} um índice para a melhor posição global.

$P_{best}[i]$ a posição de melhor localização da i-ésima partícula.

$P_{best\_fitness}[i]$ o melhor fitness local visitado pela i-ésima partícula.

Passo 1 (Inicialização): Para cada partícula i na população:

Passo 1.1: Inicialize o $PSO[i]$ randomicamente.

Passo 1.2: Inicialize $V[i]$ randomicamente.

Passo 1.3: Avalie o $fitness[i]$.

Passo 1.4: Inicialize G_{best}.

Passo 1.5: Inicialize $P_{best}[i]$ com uma cópia de $PSO[i]$ $\forall i \leq P$.

Passo 2: Repita até que um critério de parada seja satisfeito

Passo 2.1: Encontre o G_{best} tal que $fitness[G_{best}]$ $fitness[i]$ $i \leq P$.

Passo 2.2: Para cada partícula i:

$P_{best}[i] = PSO[i]$ *se* $fitness[i] > P_{best\_fitness}[i]$ $\forall i \leq P$.

Passo 2.3: Para cada particular i:

Atualize $V[i]$ e $PSO[i]$ de acordo com as equações 1 e 2.

Passo 2.4: Avalie o $fitness[i]$ $\forall i \leq P$

Detalhes de implementação

- Limites para as posições de uma partícula:
  - $x_{ij} \in [x_{min}, x_{max}]$
  - Caso x_{ij} saia deste intervalo, fazer:
    - $x_{ij} = x_{min}$ ou $x_{ij} = x_{max}$, conforme o caso
    - $v = 0$.
- Velocidade máxima:
  - $v_{min} \leq v \leq v_{max}$
- Não é necessário armazenar **g** no computador. Basta armazenar o índice *i* tal que $\mathbf{p}_i = \mathbf{g}$

Melhoramentos

- Redução linear da ponderação de inércia
  - Para reduzir gradativamente a influência da diversificação
- Fator de constrição
  - Para evitar a mudança brusca de velocidade

Redução linear da ponderação de inércia

A cada iteração k a ponderação é reduzida:

$$w^{k+1} = w_{\max} - k\left(\frac{w_{\max} - w_{\min}}{k_{\max}}\right)$$

onde $k_{\max}$ é o número máximo de iterações.

Shi e Eberhart (1998) relataram que

$$\begin{aligned} w_{\max} &= 0,9 \\ w_{\min} &= 0,4 \\ c_1 = c_2 &= 2 \end{aligned}$$

deu bons resultados em uma variedade de problemas.

Fator de constrição

Atualização de velocidade:

$$v_{ij}^{k+1} = \chi \left[v_{ij}^{k} + c_1 r_{1j} (p_{ij}^{k} - x_{ij}^{k}) + c_2 r_{2j} (g_{j}^{k} - x_{ij}^{k}) \right]$$

$$\chi = \frac{2\kappa}{|2 - \varphi - \sqrt{\varphi^2 - 4}|}$$

onde χ é o fator de constrição, $\varphi = c_1 + c_2$, $\varphi > 4$.

Valores usuais, $\kappa = 1, \varphi = 4,1 \Rightarrow \chi = 0,73$. $c_1 = c_2 = 2,05$.

PSO Discreto

- Os vetores de posição (atual, melhor posição da partícula e melhor posição global) usam valores discretos
- O vetor velocidade é substituído por um vetor de trocas de posições (que representam as trocas necessárias para sair de uma posição e ir à outra)

PSO Discreto aplicado ao PCV

- População inicial:
  - Cada partícula é um vetor *n*-dimensional, sendo *n* o número de cidades
  - Partículas geradas aleatoriamente
- Operadores discretos:
  - Velocidade
  - Movimentação
  - Obtenção da velocidade
  - Soma de velocidades
  - Multiplicação de uma constante por uma velocidade

PSO Discreto aplicado ao PCV

- Operador velocidade:
  - Lista de trocas que serão feitas. Se não há trocas, a velocidade é nula. Quanto maior o número de trocas, maior a velocidade da partícula.
  - Exemplo de uma lista de trocas (transposição):
    - $\{(i_1, j_1), (i_2, j_2), \ldots, (i_n, j_n)\}$
  - Nesta transposição, troca-se inicialmente a cidade da posição i_1 com a da posição j_1, depois faz-se a troca da cidade da posição i_2 com a da posição j_2, até a troca de i_n com j_n
  - Duas velocidades v_1 e v_2 são ditas equivalentes se ao aplicá-las a uma partícula, obtivermos uma mesma partícula resultante
  - Uma velocidade nula é uma lista vazia
  - $|v|$ indica o número de transposições

PSO Discreto aplicado ao PCV

- Operador movimentação:
  - Seja P a posição de uma partícula e v sua velocidade
  - Uma nova posição P’ para esta partícula é obtida aplicando-se a P o operador velocidade v, isto é:
    - $P' = P \oplus v$
  - Dessa forma, a partícula da posição P “voa” para a posição P’

PSO Discreto aplicado ao PCV

- Exemplo: Seja a partícula P dada por P = (2, 3, 4, 5, 1) e sua velocidade v dada por v = {(1, 2),(2, 4)}.
  - Aplicando essa velocidade v à partícula P, obtém-se uma nova partícula P' com as seguintes operações:
  - Aplicação da transposição (1, 2), isto é, troca da cidade da primeira posição de P com a cidade da segunda posição de P:
    - P' = (3, 2, 4, 5, 1)
  - Aplicação da transposição (2, 4), ou seja, troca da cidade da segunda posição de P com a cidade da quarta posição de P:
    - P' = (3, 5, 4, 2, 1)
- Desta forma, o resultado da operação $P' = P \oplus v = (3, 5, 4, 2, 1)$

PSO Discreto aplicado ao PCV

- Operador obtenção da velocidade:
  - Sejam P_1 e P_2 duas posições de uma partícula
  - A velocidade v é calculada com base na diferença entre as posições P_1 e P_2, isto é:
    - $v = P_1 - P_2$
- Exemplo: Sejam P_1=(1, 2, 3, 4, 5) e P_2=(2, 3, 1, 5, 4)
  - $P_1[1] = P_2[3] = 1$. Assim, a primeira transposição é (1, 3) e escreve-se $P_2' = P_2 + S(1, 3) = (1, 3, 2, 5, 4)$
  - $P_1[2] = P_2'[3] = 2$. Assim, a segunda transposição é (2, 3) e escreve-se $P_2'' = P_2' + S(2, 3) = (1, 2, 3, 5, 4)$
  - $P_1[4] = P_2''[5] = 4$. Assim, a terceira transposição é (4, 5) e escreve-se $P_2''' = P_2'' + S(4, 5) = (1, 2, 3, 4, 5)$
  - Desta forma, a velocidade v de uma partícula pode ser obtida a partir das posições P_1 e P_2, com base na seguinte lista de transposições:
    - $v = P_1 - P_2 = \{S(1,3), S(2, 3), S(4, 5)\}$

PSO Discreto aplicado ao PCV

- Operador multiplicação de uma velocidade por um coeficiente
  - Seja c um coeficiente e v uma velocidade
  - As constantes são usadas para determinar o quanto da velocidade atual será mantida
  - Ex.: Seja a constante $c_1=2$ e a velocidade v={(1,2),(2,4),(3,5),(3,6)}
  - Neste caso, significa que devemos escolher duas trocas aleatoriamente dentro da lista de trocas que definem a velocidade
  - Supondo que sejam escolhidas as trocas (2,4) e (3,6), o resultado do produto da constante c1 pela velocidade v é a velocidade $v' = c_1 v$ dada por:
    - v' = {(2,4), (3,6)}

PSO Discreto aplicado ao PCV

- Operador soma de velocidades:
  - Sejam v_1 e v_2 duas velocidades
  - $v = v_1 + v_2$ é calculado pela concatenação das listas de transposições associadas a v_1 e v_2
- Exemplo:
  - Se v_1 = {(1,2),(2,4),(3,5),(3,6)} e
  - v_2 = {(1,5),(3,4)}, então
  - $v = v_1 + v_2$ =
  - v = {(1,2),(2,4),(3,5),(3,6), (1,5),(3,4)}