Evolución Diferencial

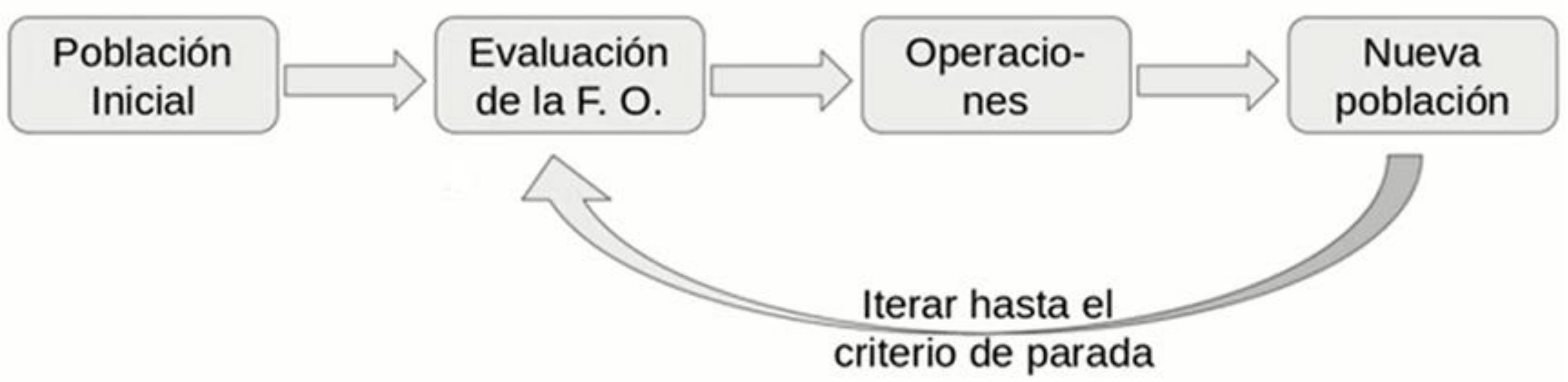
Dra. Andrea Villagra – Mg. Daniel Pandolfi
UNPA – UACO

Maestría en Informática y Sistemas

Agenda

Estructura general de las MHs

- Las Metaheurísticas en general siguen el siguiente esquema :



Evolución Diferencial

- Es un modelo evolutivo que enfatiza la mutación, utiliza un operador de cruce/recombinación a posteriori de la mutación.
- Fue propuesto por Rainer Storn and Kenneth Price (reporte técnico 1995)
- Posteriormente se consolida en un artículo científico en el Journal of Global Optimization.

Kenneth V. Price, Rainer M. Storn, and Jouni A. Lampinen Differential Evolution: A Practical Approach to Global Optimization (Natural Computing Series) Springer-Verlag, 2005.

Evolución Diferencial

- Se trata de una técnica no determinista basada en la evolución de una población de vectores (individuos) de valores reales que representan soluciones en el espacio de búsqueda.
- La generación de nuevos individuos se lleva a cabo a través de operadores de mutación y cruce.

Mutación "Diferencial"

$$\mathbf{v}_{i,g} = \mathbf{x}_{r0,g} + F \times (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g})$$

crossover:

$$\begin{array}{l} \mathbf{x}_{i,g} \in P_{\mathbf{x},g} \\ \mathbf{v}_{i,g} \in P_{\mathbf{v},g} \end{array} \quad (+) \quad \mathbf{u}_{i,g} \in P_{\mathbf{u},g}$$

Evolución Diferencial – Ejemplo numérico

- Consideramos un problema de optimización simple (function sphere) con dos variables de decisión.

- Estrategia de DE utilizada es DE/aleatorio/1/bin

- Definiremos el problema, los parámetros (del problema y de DE), la inicialización y la actualización de DE.

Evolución Diferencial – Ejemplo numérico

- Definición del Problema:

Minimizar $f(x) = {x_1}^2 + {x_2}^2 \qquad -5 \leq x_1, x_2 \leq 5$

- Parámetros de DE y del Problema

- Tamaño de la población (NP) = 5
- Dimensión del Problema (D) = 2
- Criterio de parada = Máxino nro. Iteraciones = 2
- Factor de mutación F= 0,5
- Prob. Crossover Cr = 0,7

Evolución Diferencial – Ejemplo numérico

- Inicialización

Position	Function Value
$x_1 = (1.7667, -4.1337)$	20.2089
$x_2 = (4.8071, 0.6642)$	23.5502
$x_3 = (-2.9232, -4.0439)$	24.8983
$x_4 = (-4.3747, -4.7421)$	41.6270
$x_5 = (-1.6587, 0.5680)$	3.07408

- Actualización de DE, estrategia utilizada

\- DE/aleatorio/1/bin

aleatorio = vector a perturbar (vector target)

1 = nº de diferencias

bin = crossover

Evolución Diferencial – Ejemplo numérico

- 1ra. Generación

En DE el operador de mutación se utiliza para crear un nuevo individuo (v_i) para cada solución (Vector Padre) de la población.

$$v_i = x_o + F(x_1 - x_2)$$

$x_o \neq x_1 \neq x_2$ *aletaorios en este caso*

- Para comenzar la mutación seleccionamos como vector padre el 1er. Vector de la población

Say $x_1 = (1.7667, \ -4.1337)$ is the Parent Vector.
Corresponding function value is $f(x_1) = 20.2089$

For mutation, let *Target Vector* (selected randomly from the current population) is $x_4 = (-4.3747, -4.7421)$

$Randomly\ selected\ solution_1 = x_5$
$Randomly\ selected\ solution_2 = x_3$

Evolución Diferencial – Ejemplo numérico

- 1ra. Generación

Let *Scale Factor* = 0.5

Vector $\mathbf{v}_1$ is calculated as below:

$$
\begin{aligned}
\mathbf{v}_{11} &= x_{41} + 0.5 \times (x_{51} - x_{31}) \\
&= -4.3747 + 0.5 \times (-1.6587 - (-2.9232)) \\
&= -3.7425 \\
\mathbf{v}_{12} &= x_{42} + 0.5 \times (x_{52} - x_{32}) \\
&= -4.7421 + 0.5 \times (0.5680 - (-4.0439)) \\
&= -2.4362
\end{aligned}
$$

Vector mutado $\mathrm{v}_1 = (-3.7425,\ -2.4362)$

Evolución Diferencial – Ejemplo numérico

- Crossover

$$u_i(j) = \begin{cases} x_i(j) & \text{if } rand \leq C_r; \\ v_i(j) & \text{en otro caso.} \end{cases}$$

- Por ejemplo: rand = 0,75 elijo las componentes de v_i

El nuevo individuo u_i estará formado por ambas componentes de v_i

Recordemos: x_i (vector padre) y v_i vector mutado

$$x_1 = (1.7667, \ -4.1337) \qquad v_1 = (-3.7425, \ -2.4362)$$

$$u_1 = (-3.7425, \ -2.4362).$$

Nueva población el vector padre (x_1) o el vector intermedio (u_1)?

Reemplazamos f (x_1) = 20.2089, y f (u_1) = 19.9417

El nuevo individuo $x_1 = (-3.7425, \ -2.4362)$

Evolución Diferencial – Ejemplo numérico

- El mismo mecanismo se aplica para el segundo individuo de la población

$x_2 = (4.8071, 0.6642)$ con valor objetivo **23.5502**

- Elegimos para la mutación los tres vectores (x_0, x_1 y x_2):

$$\begin{aligned} &Target\ Vector = x_5 \\ &Randomly\ selected\ solution_1 = x_3 \\ &Randomly\ selected\ solution_2 = x_1 \end{aligned}$$

Obtenemos el siguiente vector mutado $v_2 = (-1.2491, -0.2358)$ que se acepta xq ambas variables están dentro del rango del espacio de búsqueda.

Se aplica el crossover y se obtiene u_2 $(-1.2491, -0.2358)$ con $f(u_2) = 1.6158$.

Como $f(u_2) < f(x_2)$ claramente el nuevo individuo de la población es

$$x_2 = (-1.2491, -0.2358)$$

Evolución Diferencial – Ejemplo numérico

- Para la solución x_3

Elegimos para la mutación los tres vectores (x_0, x_1 y x_2):

$$\begin{aligned} &Target\ Vector = x_4 \\ &Randomly\ selected\ solution_1 = x_1 \\ &Randomly\ selected\ solution_2 = x_5 \end{aligned}$$

Obtenemos el siguiente vector mutado $v_3 = (-5.4167,\ -6.2443)$

Ambas variables están fuera de los límites redefinimos v_3

$v_3 = (-5,-5)$.

Se aplica crossover y se obtiene el vector $u_3 = (-5,-5)$ f (u_3) = 50

Como f(x_3) < f(u_3) el nuevo individuo x_3 se mantiene en la nueva población (sobrevive)

Evolución Diferencial – Ejemplo numérico

- Aplicando el mismo procedimiento a las dos soluciones restantes obtenemos los siguientes nuevos individuos:

$$x_4 = (-4.3748,\ -4.3402) \qquad f(x_4) = 37.9765$$

$$x_5 = (-1.6587, 0.5681) \qquad f(x_5) = 3.0741$$

- Luego de la primera generación la población actualizada es:

Position	Function Value
$x_1 = (-3.7425,-2.4362)$	19.9417
$x_2 = (-1.2491,-0.2358)$	1.6158
$x_3 = (-2.9232,-4.0439)$	24.8982
$x_4 = (-4.3748,-4.3402)$	37.9765
$x_5 = (-1.6587,0.5681)$	3.0741

Evolución Diferencial – Ejemplo numérico

- 2da. Generación

Position	Function Value
$x_1 = (-1.3603\ -1.9917)$	5.8173
$x_2 = (-1.2491, -0.2358)$	1.6158
$x_3 = (-2.9232, -4.0439)$	24.8982
$x_4 = (0.5233, -0.0876)$	0.2815
$x_5 = (-0.4676, 0.7903)$	0.8433

Se modificaron x_1, x_4 y x_5.

La mejor solución es x_4 y es mejor que la obtenida en la primera generación.

Evolución Diferencial – Pseudocódigo

```
Procedure DE{
  t = 0;
  Initialize Pop(t);    /* of |Pop(t)| Individuals */
  Evaluate Pop(t);
  While (Not Done)
  {for i = 1 to |Pop(t)| do
         {parent1, parent2, parent3} = Select_3_Parents(Pop(t));
         thisGene = random_int(|Pop(t)|;
         for k = 1 to n do /* n genes per Individual */
            if (random < p) *P is crossover constant in [0,1]*
               Offspring(i) = parent1(i) + mu (parent2(i) - parent3(i));
            else
               Offspring(i) = Individual(i) in Pop(t);
          end /* for k */
         Evaluate(Offspring(i));
       end /* for i */
     Pop(t+1) = {j | Offspring(j) is_better_than Individual(j)}
                {k | Individualk is_better_than Offspringk};
     t = t + 1;}
```

Aplicaciones de DE

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 15, 2019, accepted May 23, 2019, date of publication May 28, 2019, date of current version June 17, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2919507

An Improved Differential Evolution Algorithm for Optimal Location of Battery Swapping Stations Considering Multi-Type Electric Vehicle Scale Evolution

SHOUXIANG WANG[1], (Senior Member, IEEE), LU YU[1], (Student Member, IEEE), LEI WU[2], (Senior Member, IEEE), YICHAO DONG[1], AND HONGKUN WANG[1]

[1]Key Laboratory of Ministry of Education, Tianjin University, Tianjin 300072, China
[2]Stevens Institute of Technology, Hoboken, NJ 07030, USA

Aplicaciones de DE

FOCUS

CrossMark

A self-feedback strategy differential evolution with fitness landscape analysis

Ying Huang[1,2] · Wei Li[3] · Chengtian Ouyang[3] · Yan Chen[4]

Published online: 16 August 2018

International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 12 (2018) pp. 10845-10854

An Efficient Differential Evolution for Engineering Design Problems

Raghav Prasad Parouha

Assistant Professor, Department of Mathematics,
Indira Gandhi National Tribal University Amarkantak
Madhya Pradesh – 484887 India.

Aplicaciones de DE

Smart Traffic Lights: A First Parallel Computing Approach

D. Souravlias*, G. Luque†, E. Alba†, K.E. Parsopoulos*
*Department of Computer Science and Engineering,
University of Ioannina, GR-45110 Ioannina, Greece
Email: {dsouravl,kostasp}@cse.uoi.gr
†Universidad de Málaga, Andalucía Tech
E.T.S.I. Informática, Campus Teatinos,
29071 Málaga (España)
Email: {gabriel,eat}@lcc.uma.es

***Abstract*—Optimal traffic light scheduling is a fundamental problem in modern urban areas. It has severe impact on traffic flow management, energy consumption and vehicular emissions, as well as on urban noise. The vast number of traffic lights in modern cities increases the complexity of the scheduling problem and, at the same time, urgently needs for efficient algorithms that optimize the light cycle programs. In this work, we propose a solution for the traffic light scheduling problem by using Differential Evolution, and investigate the benefits of parallelism on this complex problem. For understanding the impact in the city, the popular micro-simulator SUMO is used. We evaluate our approach on close-to-reality problem scenarios consisting of**

automatic traffic control signals. A number of industrial solutions have been proposed for this problem, such as the Cross Zlín [1] and ATC [2]. These solutions focus on the real-time configuration of a single traffic light junction. Also, they require the existence of infrastructures that provide online information about the changing traffic situations. We here go in a different direction, because the increasing number of vehicles requires the transition from the local control of a single intersection to a holistic approach considering a large urban area, and because optimizing the existing traditional

Aplicaciones de DE

applied sciences

MDPI

Article

Differential Evolution: A Survey and Analysis

Tarik Eltaeib * and Ausif Mahmood

Computer Science and Engineering Department, University of Bridgeport, Bridgeport, CT 06614, USA; mahmood@bridgeport.edu

* Correspondence: teltaeib@bridgeport.edu; Tel.: +1-678-237-6229

Received: 26 July 2018; Accepted: 4 October 2018; Published: 16 October 2018