Particle Swarm Algorithmus

Tutorial

für die Vorlesung

Advanced Software Engineering

des Studiengangs Angewandte Informatik

an der Dualen Hochschule Baden-Württemberg Mosbach

von

5703004, 1716504

25. April 2022

1 Metapher

Der Particle Swarm Optimization (PSO) Algorithmus imitiert das Verhalten von Zugvögeln, die in einem Schwarm fliegen. Dabei befolgen sie drei Grundregeln:

(i) Separation: Einzelne Vögel halten Abstand zueinander, um Kollisionen mit ihren Nachbarn zu vermeiden. Dadurch sind die Vögel gleichmäßig verteilt und es bilden sich keine lokalen Haufen.

(ii) Alignment: Einzelne Vögel fliegen in die durchschnittliche Richtung ihrer Nachbarn.

(iii) Cohesion: Einzelne Vögel fliegen in Richtung des Masseschwerpunkts ihrer Nachbarn.

Die theoretischen Grundlagen wurden dabei von Craig Reynolds bereits 1986 gelegt, der dieses Verhalten mit einem Programm namens *Boids* simuliert. Die Idee des Algorithmus besteht darin, dass einzelne Agenten (Partikel) miteinander interagieren, indem sie zunächst selbstständig den Suchraum erkunden und anschließend das neu erlangte Wissen über ein kollektives Gedächtnis miteinander teilen. Jeder Partikel hält dabei ein lokales Optimum (*pBest*), zusammen mit den entsprechenden Positionskoordinaten in dem Suchraum und steuert dies dem kollektiven Gedächtnis bei, sodass der Schwarm iterativ ein globales Optimum (*gBest*) ermitteln kann.

2 Strategie

Zunächst werden alle Agenten (Partikel) mit einer zufälligen Geschwindigkeit und Position initialisiert. Anschließend werden die jeweiligen Eigenschaften eines Partikels durch das Inertia, das kognitive individuelle Verhalten eines Partikels und das soziale Verhalten gegenüber aller Partikel aktualisiert. Ziel ist es, dass alle Agenten das Optimum in einem multidimensionalen Suchraum finden.

3 Prozedur

Der PSO nutzt eine vordefinierte Menge an Partikeln, die sich durch den Suchraum bewegen und dabei sowohl von ihrem letzten besten Punkt, dem letzten besten Punkt des Schwarms, als auch ihren nächsten Nachbarn beeinflusst werden. Dabei setzt sich nach jeder Iteration die Geschwindigkeit eines einzelnen Partikels aus der Summe der aktuellen Geschwindigkeit eines Partikels, dem individuellen kognitiven Verhalten eines Partikels und seinem sozialen Verhalten im gesamten Schwarm zusammen. Dabei werden die Parameter für das individuelle

kognitive und soziale Verhalten mit festen, zu parametrisierenden, Koeffizienten gewichtet und jeweils mit einem zufälligen Wert zwischen 0 und 1 multipliziert. Anschließend werden die Ergebnisse jeweils mit der Differenz aus dem jeweiligen Optimum und der aktuellen Position des Partikels multipliziert. Daraus lässt sich die folgende Formel für die Aktualisierung der Geschwindigkeit je Iteration ableiten:

$$v_i(t+1) = v_i(t) + (c_1 * \text{rand}() * (p_i^{best} - p_i(t))) + (c_2 * \text{rand}() * (p_{gBest} - p_i(t))) \quad (1)$$

Anschließend wird die Position in jeder Iteration aus der aktuellen Position eines Partikels und der zuvor berechneten Geschwindigkeit berechnet:

$$p_i(t+1) = p_i(t) + v_i(t) \quad (2)$$

4 Pseudocode

Der PSO benötigt zwei Eingabeparameter, die zu Beginn festgelegt werden. Das ist zum einen die Anzahl der Dimensionen (beim Travelling-Salesman-Problem beispielsweise die Anzahl der Orte), zum anderen die Anzahl der im Schwarm befindlichen Partikel. Dabei liegt die optimale Anzahl der Partikel in einem Bereich zwischen 10 und 30. Der Algorithmus liefert das Partikel, welches das globale Optimum gefunden hat.

Zu Beginn des Algorithmus wird die Population und das globale Optimum mit dem Standardwert 0 initialisiert. Anschließend erhalten entsprechend der festgelegten Größe der Population nacheinander die Partikel eine zufällige Geschwindigkeit und Position im Suchraum. Anschließend werden für das jeweilige Partikel die Kosten abhängig von ihrer Position berechnet. Die aktuelle Position entspricht zu Beginn dem lokalen Optimum (*pBest*). Wenn die Kosten des lokalen Partikels kleiner als die bereits bekannten Kosten im kollektiven Gedächtnis sind, dann wird die globale beste Position mit der lokalen Position des Partikels überschrieben. Quelltext 1 bildet dieses Verhalten in einem Pseudocode ab.

Quelltext 1: Initialisierung der Parikel

```
population = 0
p_gBest = 0
for i = 1 to populationSize do
    p_velocity = randomVelocity()
    p_position = randomPosition(populationSize)
    p_cost = cost(p_position)
    p_best = = p_position
    if p_cost <= p_gBest then
        p_gBest = p_best
    end if
```

```
done
```

Anschließend wird die jeweilige Geschwindigkeit und Position der Partikel aktualisiert, die Kosten an der jeweiligen Position neu berechnet und ggf. als neues lokales Optimum gesetzt, bis die Abbruchbedingung erfüllt ist. Eine Abbruchbedingung kann beispielsweise das Erreichen einer maximalen Anzahl an Iterationen sein oder ein Fehlerwert, der unter einen vordefinierten Schwellwert fällt. Dann kann davon ausgegangen werden, dass ein globales Optimum gefunden wurde. Sind die Kosten der aktuellen Position eines Partikels kleiner als die des bisherigen globalen Optimums, wird auch dieser Wert angepasst. Quelltext 2 drückt dieses Verhalten im Pseudocode aus.

Quelltext 2: Aktualisierung der Partikel

```
while !stopCondition() do
    for all p in population do
        p_velocity = updateVelcoity(p_velocity, p_gBest, p_best)
        p_position = updatePosition(p_position, p_velocity)
        p_cost = cost(p_position)
        if p_cost s<= p_gBest then
            p_best = p_position
            if p_cost <= p_gBest then
                p_gBest = p_best
            end if
        end if
    done
done

return p_gBest
```

5 Exploration and Exploitation

Bei der Exploration erkunden Partikel den noch unbekannten Suchraum nach einem neuen Optimum. Wie stark dieser Suchraum von den Partikeln erkunden wird, hängt maßgeblich von seinem kognitiven individuellen Verhalten ab, das durch die Konstantebeeinflusst c_1 wird.

Die Exploitation sorgt vor allem dafür, dass vielversprechende, bereits bekannte Regionen genauer von den Partikeln untersucht werden. Diese Regionen werden durch das kollektive Gedächtnis des Schwarms festgelegt. Wie stark ein Partikel exploriert, wird damit maßgeblich durch die Konstante c_2 beeinflusst.

Für den Erfolg des PSO ist eine ausgewogene Balance zwischen Exploration und Exploitation unabdingbar. Ein harmonisiertes Flugverhalten bez. der Position und Geschwindigkeit

der Partikel wird durch eine Konstante, dem Inertia gewichtet. Durch das Inertia kann die Exploration und Exploitation balanciert werden.

6 Entscheidungsregeln für Schwarmverhalten

Die Inertia lässt ein einzelnes Partikel mit derselben Geschwindigkeit in dieselbe Richtung fliegen, wie zuvor.

- $inertia = 0$: Die eigene momentane Geschwindigkeit und Position v^t eines Partikels hat keinen Einfluss auf den Geschwindigkeits- und Positionsvektor $v^{(t+1)}$ der nächsten Iteration.
- $inertia = 0.5$: Die eigene Position und Geschwindigkeit eines Partikels fließt in die Berechnung der nächsten Geschwindigkeit und Position mit ein, ist allerdings nicht das alleinige Kriterium für das Flugverhalten eines Partikels. Dadurch wird das Flugverhalten von Partikeln in einem Schwarm simuliert.
- $inertia = 1$: Die momentante Geschwindigkeit und Position hat maximalen Einfluss auf den Geschwindigkeits- und Positionsvektor eines Partikels in der nächsten Iteration.

Die *Personal Influcence* betimmt, in wie weit ein Partikel zu seiner vorherigen Position zurückkehrt, die besser ist als die momentane. Die *Social Influence* hingegen lässt das Partikel in die Richtung seines besten Nachbarn fliegen.

Wird die *cognitive ratio* auf 1 gesetzt und die *social ratio* auf 0, so ignoriert ein Partikel die besten Ergebnisse seiner Nachbarn und damit des Schwarms und sucht nur in seiner eigenen Umgebung nach einem Optimum. Damit gibt es kein Schwarmverhalten zwischen den Partikeln; jedes Partikel findet ein lokales Optimum.

Wird hingegen die *social ratio* auf 1 gesetzt und die *cognitive ratio* auf 0, so tritt ein Partikel lediglich in die Fußstapfen seiner Nachbarn. Das hat zur Folge, dass keine neuen Position gesucht werden, es findet keine Exploration mehr statt.

Ein ausgewogenes Verhältnis zwischen Inertia, *Personal Influcence* und *Social Influcence* ist für den Erfolg des PSO notwendig. Dann findet eine individuelle Suche nach einer besten Position statt, zudem wird das kollektive Wissen des Schwarms in einer schwarmübergreifenden Suche nach einem globalen Optimum berücksichtigt.

7 Parameterabhängigkeiten

Die *Größe des Schwarms* bestimmt die initale Diversität eines einzelnen Partikels im Schwarm. Je größer ein Schwarm ist, desto mehr neue Positionen können in dem Suchraum pro Iteration erkundet werden. Allerdings wird dadurch auch die Komplexität der Berechnungen pro Iteration erhöht, weshalb ein zu großer Wert nicht zielführend ist. Dann gleicht das Verhalten des Algorithmus einer zufälligen parallelen Suche nach dem Optimum. Für diesen Parameter sollte deshalb ein Wert zwischen 10 und 30 gewählt werden.

Die *Anzahl an Nachbarn* bestimmt, wie stark einzelne Partikel miteinander interagieren. Je kleiner die Anzahl der Nachbarn eines Partikels ist, desto weniger Interaktion findet zwischen den Partikeln statt. Dadurch konvergiert der Algorithmus mit weniger Nachbarn langsamer, allerdings wird das tatsächliche globale Optimum eher von einem Schwarm gefunden. Zu viele Nachbarn birgen die Gefahr, dass nicht das globale Optimum gefunden wird.

Die *Anzahl an Iterationen* muss für jede Problemstellung angepasst werden. Zu wenige Iterationen können den Algorithmus frühzeitig abbrechen, obwohl das globale Optimum noch nicht gefunden wurde, zu viele Iterationen führen zu einer längeren Laufzeit des Algorithmus, die vielleicht gar nicht notwendig ist, da sich das globale Optimum nur noch geringfügig oder gar nicht mehr verändert.

8 Zusammenspiel *gBest* und *pBest*

Mit einer Konfiguration, in der die Anzahl der Partikel auf 20 gesetzt wird, die Inertia den Wert 0.729844 annimmt und sowohl das individuelle kognitive Verhalten (*cognitive ratio*) und das soziale Verhalten (*social ratio*) auf den Wert 1.496180 gesetzt werden, ist das Verhältnis zwischen Exploration und Exploitation ausgewogen, weil sowohl die individuelle Suche eines Partikels nach einem Optimum (effektives Schwarmverhalten), als auch das kollektive Gedächtnis des Schwarms (koordiniertes Schwarmverhalten) gleichermaßen berücksichtigt werden. Somit hat der Schwarm bereits nach wenigen Iterationen das globale Optimum *gBest* gefunden.

Ein ausgewogenes Verhalten zwischen effektivem und koordiniertem Schwarmverhalten ist notwendig, da sonst entweder jeder Partikel unabhängig nach einer Lösung sucht, was der Komplexität einer zufälligen Suche gleichkommt oder aber alle Partikel in die Fußstapfen ihrer Nachbarn treten und dadurch den Suchraum nicht weiter explorieren, wodurch das globale Optimum nicht gefunden werden kann.