KARADENİZ TEKNİK ÜNİVERSİTESİ OF TEKNOLOJİ FAKÜLTESİ
YAZILIM MÜHENDİSLİĞİ BÖLÜMÜ

2019 – 2020 BAHAR DÖNEMİ
YZM-4032 META-SEZGİSEL OPTİMİZASYON DERSİ

ÖDEV KONUSU: Karga Arama Algoritması ve Performansını Arttırmaya Yönelik Çalışmalar

1. Karga Arama Algoritması (Crow Search Algorithm)

1.1. Giriş

Kargalar en akıllı kuşlar olarak kabul edilir. Vücut boyutlarına göre en büyük beyni içerirler. Kargalar yüzleri hatırlayabilir ve düşmanca yaklaştıklarında birbirlerini uyarabilir. Dahası, sofistike yollarla iletişim kurabilir ve birkaç ay sonrasına kadar yiyeceklerinin saklanma yerlerini hatırlayabilirler. Kargaların diğer kuşları izledikleri, diğer kuşların yiyeceklerini nerede sakladıklarını gözlemledikleri ve sahibi ayrıldıktan sonra çaldığı bilinmektedir. Eğer bir karga hırsızlık yapmışsa, gelecekteki bir kurban olmaktan kaçınmak için saklanma yerlerini taşımak gibi ekstra önlemler alacaktır. Bir hırsızın davranışını tahmin edebilmek için hırsız olma deneyimlerini kullanırlar. Saklamış oldukları yiyeceklerin çalınmasını önleyebilmek amacıyla en güvenli rotayı belirleyerek gizli yiyecek depolarının keşfedilmesini engellemeye çalışırlar. CSA, kargaların akıllı davranışlarına dayanarak geliştirilmiş popülasyon temelli bir meta-sezgisel algoritmadır.

CSA ilkeleri aşağıdaki gibidir:

- Kargalar sürü şeklinde yaşar.
- Kargalar yiyecekleri sakladıkları yerlerin pozisyonlarını ezberler.
- Kargalar hırsızlık yapmak için birbirlerini takip ederler.
- Kargalar önbelleklerini olasılıkla çalınması ihtimaline karşı korurlar.

Birkaç karga içeren d-boyutlu bir ortam olduğu varsayılmaktadır. Karga sayısı (sürü boyutu) N'dir ve arama alanındaki karga i'nin konum vektörleri $x^{i,iter}(i=1,2,\ldots,N;\ iter=1,2,\ldots,iter_{max})$ burada $x^{i,iter}=[\ x^{i,iter}_{1},\ x^{i,iter}_{2,\ldots},\ x^{i,iter}_{d}]$ ve $iter_{max}$ maksimum yineleme sayısıdır. Her karga, saklandığı yerin konumunun ezberlendiği bir anıya sahiptir. Yinelemede, karga i'nin saklanma yeri $m^{i,iter}$ ile gösterilir. Her karganın anısına en iyi deneyiminin konumu ezberlenmiştir. Kargalar çevrede hareket eder ve daha iyi yiyecek kaynakları arar. Yineleme olduğunda, karga j'nin saklanma yeri olan $m^{j,iter}$ ziyaret etmek istediğini varsayalım. Bu yinelemede karga, karga j'nin saklandığı yere yaklaşmak için karga j'yi takip etmeye karar verir. Bu durumda, iki durum olabilir:

Durum 1: Karga j karga i'nin onu takip ettiğini bilmiyor. Sonuç olarak karga i, karga j'nin saklandığı yere yaklaşacaktır. Bu durumda, karga i'nin yeni pozisyonu aşağıdaki gibi elde edilir:

$$x^{i,iter+1} = x^{i,iter} + r_i * fl^{i,iter} *(x^{j,iter} - x^{i,iter}) \qquad (1)$$

burada r_i, 0 ile 1 arasında eşit dağılımlı rasgele bir sayıdır ve $fl^{i,iter}$, yineleme yinelemesindeki karga i'nin uçuş uzunluğunu gösterir. Küçük fl değerleri yerel aramaya ($x^{i,iter}$ civarında) ve büyük değerler global aramaya ($x^{i,iter}$'den uzak) yol açar. Şekil 1 'de gösterildiği gibi, fl değeri 1'den küçük seçilirse, karganın i bir sonraki konumu $x^{i,iter}$ ve $x^{j,iter}$ arasındaki kesik çizgi üzerindedir. Şekil 1 (b) 'nin belirttiği gibi, fl değeri 1'den fazla seçilirse, karga i'nin bir sonraki konumu çizgi çizgisinde $x^{j,iter}$ 'yi aşabilir;

Durum 2: Karga j, karganın onu takip ettiğini biliyor. Sonuç olarak, önbelleğinin çalınmasını önlemek için, karga j, arama alanının başka bir konumuna giderek karga i'yi kandırır.

Bütünüyle 1. ve 2. durumlar aşağıdaki gibi ifade edilebilir:

$$x^{i,\text{iter}+1} = \begin{cases} x^{i,\text{iter}} + r_i \times fl^{i,\text{iter}} \times (m^{j,\text{iter}} - x^{i,\text{iter}}) & r_j \geqslant AP^{j,\text{iter}} \\ \text{a random position} & \text{otherwise} \end{cases} \quad (2)$$

Burada r_j, 0 ile 1 arasında eşit dağılımlı rastgele bir sayıdır ve $AP^{j,\text{iter}}$ yineleme yinelemesinde karga j'nin farkındalık olasılığını gösterir.

Meta-sezgisel algoritmalar çeşitlendirme ve yoğunlaşma arasında iyi bir denge sağlamalıdır. CSA'da yoğunlaşma ve çeşitlendirme esas olarak farkındalık olasılığı (AP) parametresi tarafından kontrol edilir. Farkındalık olasılığı değerinin düşmesi ile CSA, bu bölgede mevcut iyi bir çözümün bulunduğu yerel bir bölgede arama yapma eğilimindedir. Sonuç olarak, küçük AP değerlerinin kullanılması yoğunlaşmayı artırır. Öte yandan, farkındalık olasılığı değerinin artmasıyla, mevcut iyi çözümlerin yakınında arama olasılığı azalır. Dolayısıyla, büyük AP değerlerinin kullanılması çeşitliliği artırır.

1.2. CSA Adımları

CSA'nın sahte kodu Şekil 2'de gösterilmektedir. CSA'nın uygulanması için adım adım prosedür bu bölümde verilmektedir.

1.2.1. Problemi ve ayarlanabilir parametreleri başlatın.

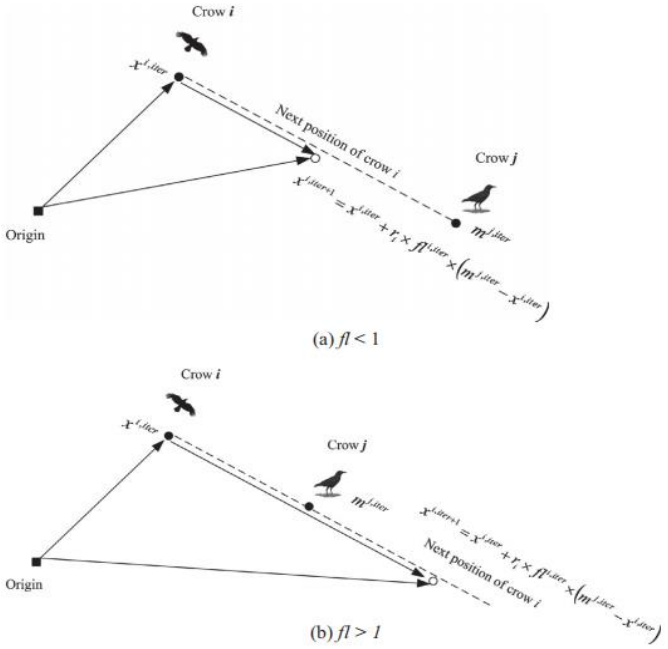
Optimizasyon problemi, karar değişkenleri ve kısıtlamalar tanımlanır. Ardından, CSA'nın ayarlanabilir parametreleri (sürü boyutu (N), maksimum yineleme sayısı ($iter_{max}$), uçuş uzunluğu (fl) ve farkındalık olasılığı (AP)) değerlenir.

1.2.2. Kargaların konumunu ve hafızasını başlatın.

N karga, sürünün üyeleri olarak rastgele bir d-boyutlu arama alanına yerleştirilir. Her karga sorunun uygulanabilir bir çözümünü belirtir yani kargalar arayıcıları temsil eder. d karar değişkenlerinin sayısıdır.

$$\text{Crows} = \begin{bmatrix} x_1^1 & x_2^1 & \dots & x_d^1 \\ x_1^2 & x_2^2 & \dots & x_d^2 \\ \vdots & \vdots & \vdots & \vdots \\ x_1^N & x_2^N & \dots & x_d^N \end{bmatrix} \quad (3)$$

Her karganın hafızası başlatıldı. İlk iterasyonda, kargaların hiçbir deneyimi olmadığı için, yiyeceklerini başlangıç pozisyonlarında gizledikleri varsayılmaktadır.



Şekil 1 : CSA (a) fl <1 ve (b) fl> 1'de durum 1 akış şeması. Karga i, kesik çizgi üzerindeki her konuma gidebilir.

KARGA ARAMA ALGORİTMASININ SÖZDE KODU

Arama alanındaki N satır sürüsünün konumunu rastgele başlat
Kargaların konumunu değerlendirin
Her karganın hafızasını başlat

while iter<iter$_{max}$
for i = 1 : N(Sürünün tüm N kargaları)
İzlenecek kargalardan birini rastgele seç (örneğin j)
Bir farkındalık olasılığı tanımlayın
if $r_j >= AP^{j,iter}$
$x^{i,iter+1} = x^{i,iter} + r_i * fl^{i,iter} * (m^{j,iter} - x^{i,iter})$
else
$x^{i,iter+1}$ *= arama alanının rastgele konumu*
end if
end for

Yeni pozisyonların fizibilitesini kontrol edin
Kargaların yeni pozisyonunu değerlendirin
Kargaların hafızasını güncelleyin

end while

1.2.3. Uygunluk fonksiyonunu değerlendirme

Her karga için, pozisyonunun kalitesi, karar değişken değerlerini amaç fonksiyona ekleyerek hesaplanır.

1.2.4. Yeni konum oluşturun

Kargalar arama alanında şu şekilde yeni bir konum oluşturur: varsayılan karga i yeni bir konum oluşturmak istiyor. Bu amaç için, bu karga, sürü kargalarından birini (örneğin, karga j) rastgele seçer ve bu karga (m^j) tarafından gizlenen yiyeceklerin konumunu keşfetmek için izler. Karga i'nin yeni pozisyonu denklem 2'deki gibidir. Bu süreç tüm kargalar için tekrarlanır.

1.2.5. Yeni pozisyonların fizibilitesini kontrol edin

Her karganın yeni pozisyonunun fizibilitesi kontrol edilir. Eğer bir karganın yeni pozisyonu mümkün ise, karga pozisyonunu günceller. Aksi takdirde, karga mevcut pozisyonda kalır ve üretilen yeni pozisyona hareket etmez.

1.2.6. Yeni pozisyonlar için uygunluk fonksiyonunu değerlendirin

Her bir karganın yeni pozisyonu için uygunluk fonksiyonu değeri hesaplanır.

1.2.7. Belleği/Konumu güncelleyin

Kargalar hafızasını şu şekilde günceller:

$$m^{i,iter+1} = \begin{cases} x^{i,iter+1} & f(x^{i,iter+1}),\ f(m^{i,iter})\text{'den daha iyidir.} \\ m^{i,iter} & \text{o.w} \end{cases}$$

Burada f (.) amaç fonksiyon değerini belirtir.

Bir karganın yeni pozisyonunun fitness fonksiyonu değerinin, hafızaya alınan pozisyonun fitness amaç fonksiyonu değerinden daha iyi olması durumunda, karga hafızasını yeni pozisyonla günceller.

1.2.8. Sonlandırma kriterini kontrol edin

Adım 4–7 $iter_{max}$'a ulaşılana kadar tekrarlanır. Sonlandırma kriteri karşılandığında, objektif fonksiyon değeri açısından belleğin en iyi konumu optimizasyon probleminin çözümü olarak rapor edilir.

2. Yapılan Çalışmalar ve Sonuçları

- Bu çalışmada hazırlanmış olan tüm caseler 10 farklı problem üzerinde 25 kez çalıştırılmıştır.
- Popülasyon boyutu 50, problem boyutu 30'dur.
- CSA algoritmasında arama uzayını etkileyen iki parametre bulunmaktadır. Bunlar bir karganın takip edildiğini anlaması için kullanılan farkındalık olasılığı ve iki karga arasındaki uçuş uzaklığıdır. Bu parametreler algoritma başlangıcında belirlenip, iterasyon boyunca sabit kalmaktadır. Yapılan çalışmada algoritmanın bu parametrelerinin değerleri sırayla 0.1 ve 2 alınmıştır.
- " - " : Popülasyonun geriye kalan kısmına ilgili yöntemin uygulandığı anlamına gelir.
- " 1 " : İlgili yöntemin popülasyonun tamamına uygulandığı anlamına gelir
- "Csa" : Algoritmanın kendi yöntemini uyguladığı anlamına gelir.

Durumlar	Oran	Uygulanan Yöntem	Skor (Friedman)
CSA	1	Csa	13,042
Case_1	rand < 0.1 -	FdbPop Csa	12,732
Case_2	rand < 0.1 -	FdbPopArtDu Csa	12,292
Case_3	rand < 0.1 -	FdbPopDinamik Csa	12,182
Case_4	rand < 0.5 -	FdbPop FdbPopArtDu	12,632
Case_5	rand < 0.5 -	FdbPop FdbDinamik	12,9
Case_6	rand < 0.5 -	FdbPopArtDu FdbPopDinamik	12,144
Case_7	rand < 0.3 -	Csa FdbPopDinamik	11,888
Case_8	rand < 0.6 -	Csa FdbPopDinamik	11,924
Case_9	rand < 0.3 -	Csa FdbPop	12,852
Case_10	rand < 0.4 -	Csa FdbPopArtDu	12,336
Case_11	rand < 0.8 -	FdbPopDinamik Csa	12,412
Case_12	rand < 0.7 -	FdbPopDinamik FdbPop	13,1
Case_13	rand < 0.8 -	Csa FdbPopArtDu	12,95
Case_14	rand < 0.8 -	Csa FdbPopDinamik	12,612
Case_15	rand < 0.8 -	Csa FdbPop	12,498
Case_16	rand < 0.7 -	Csa FdbPop	12,184
Case_17	rand < 0.6 -	Csa FdbPopDinamik	13,076
Case_18	rand < 0.9 -	Csa FdbPopDinamik	12,168
Case_19	rand < 0.8 -	Csa FdbPopArtDu	12,444
Case_20	rand < 0.8 -	Csa FdbPop	12,514
FdbPop	1	FdbPop	12,206
FdbPopArtDu	1	FdbPopArtDu	12,092
FdbPopDinamik	1	FdbPopDinamik	12,82

Durumlar	Skor (Friedman)	Wilcoxson		
		İyi	Aynı	Kötü
CSA	13,042	0	10	0
Case_1	12,732	1	9	0
Case_2	12,292	1	9	0
Case_3	12,182	0	10	0
Case_4	12,632	0	10	0
Case_5	12,9	1	9	0
Case_6	12,144	0	10	0
Case_7	11,888	0	10	0
Case_8	11,924	1	9	0
Case_9	12,852	0	10	0
Case_10	12,336	0	10	0
Case_11	12,412	1	9	0
Case_12	13,1	0	10	0
Case_13	12,95	0	10	0
Case_14	12,612	0	10	0
Case_15	12,498	1	9	0
Case_16	12,184	0	10	0
Case_17	13,076	0	10	0
Case_18	12,168	0	10	0
Case_19	12,444	0	10	0
Case_20	12,514	0	10	0
FdbPop	12,206	0	10	0
FdbPopArtDu	12,092	1	9	0
FdbPopDinamik	12,82	0	10	0

3. Sonuç

Karga arama algoritmasında konumu güncellenen kargaların, hesaplanan yeni uygunluk değerleri hafızalarındaki uygunluk değerleri ile karşılaştırılır. Yeni uygunluk değeri hafızadan daha iyi ise güncelleme yapılır. Bu nedenle çözüm adaylarının birbirine benzemesi diğer bir deyişle popülasyonun bir araya toplanması durumu oluşabilir dolayısıyla algoritma yerel minimum tuzaklarına takılabilir. Bu noktada çeşitlilik yapılması gerekmektedir. Ancak optimizasyon problemlerinin yerine getirmesi gereken çeşitlilik ve komşuluk aramasının dengeli bir şekilde ayarlanması gerekir. Özellikle bu noktaya dikkat edilmelidir.

Yapılan çalışmada 24 adet case oluşturulmuştur. Oluşturulan caseler genel olarak baz algoritmaya karşı bir üstünlük sağlamıştır. Arama performanslarında iyileştirme olmuştur. Varılan diğer bir sonuç ise uygulanan yöntemlerin algoritmaya çeşitlilik kazandırdığıdır.

23 tane case'den 21 tanesi baz algoritmaya göre daha iyi sonuç vermiştir. Elde edilen en iyi sonuç Case_7'ye aittir ve bu sonuç11,888'dir. Csa'ya göre 1,154'lük bir başarı elde etmiştir.