Zeki Optimizasyon Teknikleri
Intelligent Optimization Techniques

Hazırlayan: M. Ali Akcayol
Gazi Üniversitesi
Bilgisayar Mühendisliği Bölümü

Bu dersin sunumları, “Singiresu S. Rao, Engineering Optimization: Theory and Practice, Wiley, 2009.” kitabı kullanılarak hazırlanmıştır.

İçerik

- Rastgele arama
- Stokastik hill climbing
- İteratif lokal arama
- Tabu arama

Rastgele arama

- Rastgele arama (random search) algoritması, **doğrudan arama yöntemidir** (ek bilgiye gerek duymaz, amaç fonksiyonu gradyan değeri vb.).
- Sürekli domain'de arama yapmak için **türeve ihtiyaç duymaz.**
- Tüm arama uzayında **uniform olasılık dağılımına göre örnekleme yapılır.**

Pseudocode for Random Search.

```
Input: NumIterations, ProblemSize, SearchSpace
Output: Best
Best ← ∅;
foreach iter_i ∈ NumIterations do
    candidate_i ← RandomSolution(ProblemSize, SearchSpace);
    if Cost(candidate_i) < Cost(Best) then
        Best ← candidate_i;
    end
end
return Best;
```


Rastgele arama

Sezgiseller

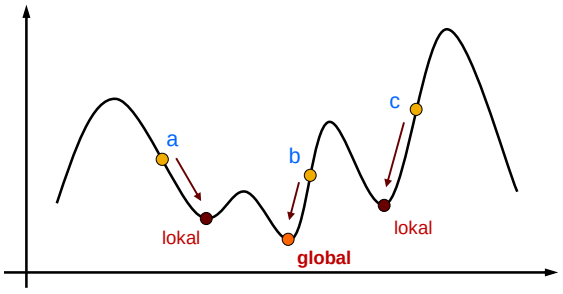
- Rastgele arama, sadece **aday çözüm oluşturma** ve **aday çözümü değerlendirme fonksiyonlarına** ihtiyaç duyar.
- Rastgele aramada, **hafıza yoktur** ve **geçmişten bağımsız** bir şekilde **örnekleme yapar** (blindly resample).
- Rastgele arama **düşük boyutlu arama uzaylarında** makul bir sürede **optimal çözüme yakınsama yapabilir.**
- **Aday çözüm oluşturma yöntemi probleme özgüdür** ve çok iyi belirlenmesi gerekir.
- **Rastgele arama sonucu**, lokal arama yapan **başka bir algoritma için** (Simple Hill Climbing) **başlangıç olarak alınabilir.**

İçerik

- Rastgele arama
- Stokastik hill climbing
- İteratif lokal arama
- Tabu arama

Stokastik hill climbing

- Stokastik hill climbing algoritması **lokal optimizasyon yapar.**
- Stokastik hill climbing algoritması **doğrudan arama yapar** ve **sürekli domain'de** arama yapmak için **türeve ihtiyaç duymaz.**
- **Deterministik hill climbing** komşu çözümü greedy yaklaşımı ile belirler ve **lokal optimumda kalma olasılığı yüksektir.**



Stokastik hill climbing

- **Basit hill climbing algoritmasının** (deterministik) **geliştirilmiş şeklidir.**

Pseudocode for Stochastic Hill Climbing

```
Input: Iter_max, ProblemSize
Output: Current
Current ← RandomSolution(ProblemSize);
foreach iter_i ∈ Iter_max do
    Candidate ← RandomNeighbor(Current);
    if Cost(Candidate) ≥ Cost(Current) then
        Current ← Candidate;
    end
end
return Current;
```


Stokastik hill climbing

Sezgiseller

- Stokastik hill climbing, komşulukların doğrudan gösterildiği **kesikli domain'de kullanılır.**
- Stokastik hill climbing combinatorial optimizasyon problemlerinin çözümünde kullanılır **(register atama, scheduling, timetabling, packet routing)**.
- Algoritma, **sürekli domain'de** komşu aday çözüm tanımlamak için **adım boyutu belirlenerek kullanılabilir.**
- Stokastik hill climbing, **diğer global arama algoritmalarının çözümünün iyileştirilmesinde kullanılabilir.**
- Algoritma iyi düzeyde sonuç almak için **tekrarlı bir şekilde yeniden başlatılabilir** (Multiple Restart Hill Climbing).
- Algoritma **eş zamanlı çoklu aday çözüm oluşturularak kullanılabilir** (Parallel Hill Climbing).

İçerik

- Rastgele arama
- Stokastik hill climbing
- İteratif lokal arama
- Tabu arama

İteratif lokal arama

- İteratif lokal aramada, **çok sayıda aday komşu çözüm oluşturulur** ve lokal arama teknikleri ile **her birisi için lokal optimum bulunur.**
- Mevcut en iyi çözümün **sorun olan kısımlarını düzeltmek** için aday çözüm oluşturulur.

Pseudocode for Iterated Local Search.

```
   Input:
   Output: S_best
1  S_best ← ConstructInitialSolution();
2  S_best ← LocalSearch();
3  SearchHistory ← S_best;
4  while ¬ StopCondition() do
5  |  S_candidate ← Perturbation(S_best, SearchHistory);
6  |  S_candidate ← LocalSearch(S_candidate);
7  |  SearchHistory ← S_candidate;
8  |  if AcceptanceCriterion(S_best, S_candidate, SearchHistory) then
9  |  |  S_best ← S_candidate;
10 |  end
11 end
12 return S_best;
```

İteratif lokal arama

Sezgiseller

- İteratif lokal arama, **kesikli domain'de combinatoryal optimizasyon problemlerine** uygulanır.
- **Perturbation** fonksiyonu, **mevcut en iyi çözümü** geçmiş bilgileri kullanarak **sezgisel olarak değiştirir.**
- **Perturbation** fonksiyonu, **probleme özgüdür.**
- **Başlangıç** noktası **rastgele** veya önceki bilgilere göre **heuristic seçilebilir.**
- Aday çözümün kabul edilme şartı, **mevcut en iyiden daha iyi olmasıdır.**

İçerik

- Rastgele arama
- Stokastik hill climbing
- İteratif lokal arama
- Tabu arama

Tabu arama

- 1986 yılında Fred W. Glover tarafından geliştirilmiştir.
- **Lokal minimumu elimine edebilir** ve **global minimumu bulabilir.**
- Tabu arama, **kısa dönemli** ve **uzun dönemli hafızaya sahiptir.**
- **Kısa dönemli hafızada, yakın zamanda ziyaret edilen çözümlere ilişkin bilgi tutulur** ve kısa süre içinde tekrarlanması önlenir.
- **Uzun dönemli hafızada,** algoritmanın başlangıcından itibaren **elde edilen iyi çözümlere ilişkin bilgi tutulur.**
- Değerlendirme fonksiyonu kullanılarak **her iterasyon için en iyi komşu çözüm bulunur.**

Tabu arama

- **Tabu listesi kullanılarak kabul edilebilir değişimler kısıtlanır.**
- Böylece, mevcut çözümden hangi komşu çözümlere geçiş yapılabileceği belirlenir.
- Tabu listesi kullanılarak daha **önceki çözümlerin tekrar edilmesi** (cycling problem) **engellenir.**
- **Tabu listesi her iterasyonda güncellenir, ekleme ve çıkarma yapılabilir.**
- Tabu listesine yeni ekleme ve var olanı çıkarma işlemleri **önceden belirlenmiş kurallara göre yapılır.**

Tabu arama

Parametreler

- Başlangıç çözümü
  - **Rastgele, önceki deneyimlere göre** veya bir algoritma ile seçilebilir.
- Yeni komşu çözüm oluşturma fonksiyonu
  - **Problem yapısına bağlıdır** ve performansı doğrudan etkiler.
  - Bir değişkende değişiklik yapılarak yeni çözüm kümesi oluşturulur.
- Kısa dönemli hafıza (tabu listesi)
  - **Arama sırasında karşılaşılan durumları tutar** ve yakın zamanda tekrarını önler.
  - **Belirli süre sonra** tabu olan **olaylar listeden çıkarılır.**
  - Genellikle listeye ilk giren ilk çıkar stratejisi (FIFO) uygulanır.

Tabu arama

Parametreler

- Uzun dönem hafıza
  - **Bulunan en iyi çözümü tutar.**
  - Arama süresince bulunmuş **elit çözümleri tutar.**
- Tabu yıkma kriterleri
  - **Belirlenen kriterler sağlandığında** bir komşu çözüm **tabu listesinden çıkarılır** ve yeni çözüm olarak alınır.
  - Mevcut çözümden **daha iyi bir çözüm** tabu da olsa alınır.
- Algoritmayı durdurma şartları
  - **Belirli bir iterasyona ulaşıldığında** sonlanır.
  - **Daha iyi çözüm bulunamadığında** sonlanır.

Tabu arama

Yeni çözüm oluşturma

- **Mevcut çözümden daha iyi bir çözüme geçiş yapılır.**
- Yeni çözümü oluşturan olası değişimler;
  - Seçilen bir x_j değişkeninin **0 değerinden 1 değerine değişmesi**.
  - Seçilen bir x_k değişkeninin **1 değerinden 0 değerine değişmesi.**
  - $f(x)$ amaç fonksiyonu değerinden $f(x')$ amaç fonksiyonu değerine değişmesi.
  - Tanımlanan bir g fonksiyonunun $g(x)$ değerinden $g(x')$ değerine değişmesi.

Tabu arama

Kısa dönemli hafıza

- Kısa süre önce karşılaşılan bilgileri tutar.
- **Önceki çözümlere** kısa sürede tekrar **dönüşü engeller.**
- **Her iterasyonda yenilenir.**
- **Yeni yapılan çözüm geçişlerinin tersini tabu yapar.**
- Her yeni tabu için **tabu süresi (t) atanır** ve i.iterasyonda eklenen tabu durumu i+t **süresi sonunda kaldırılır.**

Tabu arama

Uzun dönemli hafıza

- **Aramayı iyileştirir** ve aşağıdaki özelliklere sahiptir;
  - **Elit çözümlerin** ve niteliklerin **listesini tutar.**
  - İlerde bu elit çözümleri oluşturan niteliklere yeniden dönüş yaparak **bu bölgelerin daha detaylı araştırılmasını sağlar.**

Tabu arama

Tabu kısıtlamaları

- **İstenmeyen çözüm değişimi** ve **yasakları** uygulamak için kullanılır.
- Herhangi bir iterasyonda, x_k **çözümünden** x_{k+1} **çözümüne geçildiğinde,** bu **çözümün tersi tabu yapılabilir.**
- Örnek kısıtlamalar:
  - x_j değişkeninin **1 değerinden 0 değerine değişmesi** (daha önce 0 değerinden 1 değerine değişmişse)
  - x_j değişkeninin **0 değerinden 1 değerine değişmesi** (daha önce 1 değerinden 0 değerine değişmişse)
  - Tanımlanmış bir $g(x)$ f**onksiyonunun** y_k **değerinden** y_n **değerine değişmesi**
    (daha önce y_n değerinden y_k değerine değişmişse)

Tabu arama

Tabu süresi belirleme

- **Tabu süresi çok kısa ise, arama** sürekli **yerel optimumlarda gerçekleşir.**
- **Tabu süresi çok uzun ise,** hareketlerin çoğu tabu olduğundan **bulunan çözümlerin kalitesi düşer.**
  - **Statik:** Sabit bir değer seçilir ($t = 7$ veya $t = \sqrt{n}$ gibi) (n problem boyutunun bir ölçüsüdür, olası aday çözüm sayısı).
  - **Basit dinamik:** t_{min} ve t_{max} sınırları **arasında** sistematik veya rastgele bir sayı seçilerek t değeri belirlenir.
    t_{min} ve t_{max} statik değer ataması ile seçilebilir.
  - **Niteliğe bağlı:** t_{min} ve t_{max} niteliğin kalitesine bağlı olarak dinamik olarak belirlenir.
    t değeri basit dinamik kuraldaki gibi seçilebilir.

Tabu arama

Tabu yıkma şartları

- Tabu yıkma şartları oluşursa, tabu olan bir geçişin yapılması sağlanır.
- Yokluğa göre tabu kaldırma
  - **Tüm mümkün hareketler tabu ise**, tabu süresinin **bitmesi en yakın olan alınır.**
- Amaca göre tabu kaldırma
  - **En iyi çözümden daha iyi çözüm veren hareket** tabu olsa da alınır.
- Etkiye göre tabu kaldırma
  - Düşük etkili bir hareket, **yüksek etkili bir harekete yol açacaksa** tabu olsa da seçilir.

Tabu arama

Algoritmayı sonlandırma şartları

- Optimum çözümü bilinen problemlerde **optimum sonucun bulunması.**
- Belirlenmiş olan **maksimum iterasyona ulaşılması.**
- Mevcut en iyi çözümde **belirli bir süre daha fazla iyileştirme yapılamaması.**

Tabu arama

Tabu Search Pseudo Code

```
S_best Tabu_Search(TabuList)
begin
  S_current, S_candidate, S_best ← ConstructInitialSolutions();
  LongTermMemory (LTM) ← InitializeMemoryLTM();
  ShortTermMemory (STM)← InitializeMemorySTM();

  repeat
    S_candidate = SelectBestNeighbor();

    if ((not move(STM, S_candidate)) OR (Aspiration(S_candidate))
    then
      S_current = S_candidate;
      S_best = UpdateBest(S_best, S_candidate)
      STM = UpdateShortTermMemory(STM + S_current)
      LTM = UpdateLongTermMemory(LTM + S_best)
    end if
  until StopCondition()

  return S_best
end
```

Ödev

- Tabu arama algoritmasının uygulamasını içeren bir makale araştırma ödevi hazırlayınız.