T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

TAVLAMA BENZETİMİ ALGORİTMASI İLE AKIŞ TİPİ ÇİZELGELEMEDE TOPLAM AKIŞ ZAMANININ MİNİMİZASYONU

YÜKSEK LİSANS TEZİ

Arş.Gör. Gökay AKKAYA

Enstitü Anabilim Dalı : ENDÜSTRİ MÜHENDİSLİĞİ

Enstitü Bilim Dalı : YÖNEYLEM ARAŞTIRMASI

Haziran 1997

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

TAVLAMA BENZETİMİ ALGORİTMASI İLE AKIŞ TİPİ ÇİZELGELEMEDE TOPLAM AKIŞ ZAMANININ MİNİMİZASYONU

YÜKSEK LİSANS TEZİ

Arş.Gör. Gökay AKKAYA

Enstitü Anabilim Dalı : ENDÜSTRİ MÜHENDİSLİĞİ

Enstitü Bilim Dalı : YÖNEYLEM ARAŞTIRMASI

Bu tez .../..../ 1997 tarihinde aşağıdaki jüri tarafından Oybirliği/Oyçokluğu ile kabul edilmiştir.

Yrd. Doç. Dr. Harun R. YAZGAN	Yrd. Doç. Dr. Emin GÜNDOĞAR	Yrd. Doç. Dr. İsmail H. CEDİMOĞLU
Jüri Başkanı	**Jüri Üyesi**	**Jüri Üyesi**

ÖNSÖZ

Günümüzde üretim, hayatımızın en önemli bir gereksinimi durumundadır. Ama bu üretim, nasıl? Ne şekilde? Ne seviyede? yani kısaca en optimal ve verimli şekilde nasıl olmalıdır? İşte bu sorunun cevabı için dünyada bir çok teknik geliştirilmektedir. Özellikle son yıllarda sezgisel algoritma adını verdiğimiz teknikler geliştirilmiştir. İşte Tavlama Benzetimi yaklaşımı da bu tekniklerden biridir ve son yıllarda Tabu Search, Genetik Algoritma ile birlikte çok sık kullanılmaktadır.

Her araştırma programı gibi burada da incelemeler, yeni sorunlar ve çalışma konularını ortaya çıkartmıştır.

Tezin hazırlanmasında sırasında her türlü desteği sağlayarak bana büyük yardımları dokunan danışmanım Yrd. Doç. Dr. Harun R. YAZGAN' a, benden yardımlarını esirgemeyen başta bölüm başkanımız Doç Dr. Harun TAŞKIN olmak üzere bütün Endüstri Mühendisliği Bölümü hocalarıma ve arkadaşlarıma, bu çalışmamda bana destek olan ev arkadaşlarım, Fatih TEPEDELEN' e, Mehmet YAZICI' ya, Emin TEPEDELEN' e ve Aydın KURT' a, benim bu günlere gelmemde çok büyük emekleri olan başta annem Bergüzar AKKAYA ve babam Hüseyin AKKAYA olmak üzere, bütün aile fertlerime teşekkürlerimi sunarım.

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar Kelimeler : Tavlama Benzetimi, kombinatoriyal optimizasyon, iş çizelgeleme, sezgisel algoritma.

Bu tez, üretim planlamanın NP-hard problemi olan ve bu konuda birçok uygulama ve algoritmanın geliştirildiği iş çizelgelemede, toplam akış zamanının minimize edilmesi konusunu içermektedir. Burada da bu tekniklerden biri ve son yıllarda geliştirilen Tavlama Benzetimi algoritması yaklaşımı ile toplam akış zamanı minimize edilmiştir.

SUMMARY

Keyword : Simulated Annealing, Combinatorial Optimization, Job Scheduling, Heuristic Algorithm.

In this thesis, scheduling problem which is an NP-hard, has been solved. The objective of the problem is to minimization of total flow time. Simulated Annealing algorithm has been applied in order to achieve the objective of the problem

BÖLÜM 1. GİRİŞ

Tavlama Benzetimi (TB), istatistiksel mekanizmalardan alınan düşüncelerin kullanılmasıyla kombinatoriyal optimizasyon problemlerinin çözümü için son yıllarda kullanılan yeni bir sezgisel metod algoritmasıdır. SA' nın başlıca uygulandığı bazı konular şunlardır:

- İş çizelgeleme problemleri,
- Gezgin satıcı adam problemleri,
- Envanter problemleri,
- İstatistiksel problemler,
- Optimizasyon problemleri,
- Planlama problemleri,
- Üretim ve imalat problemleri,
- Araç rotalama problemleri,
- Atama problemleri,
- Yapay sinir ağlar problemleri,
- Yöneylem araştırması problemleri vb.

İş çizelgeleme problemi, hangi işin hangi makinada hangi sırayla ve hangi zamanda işleme konulması gerektiğine karar verme olayı olarak tanımlanabilir. İş çizelgeleme problemlerinin üretim yapıları iki başlık altında sınıflandırılabilir :
Atölye tipi çizelgeleme
Akış tipi çizelgeleme
Bu çalışmada akış tipi üretim çizelgeleme problemine Tavlama Benzetimi algoritmasının bir uygulaması yapılmıştır.

BÖLÜM 2. İŞ ÇİZELGELEME

2.1. Çizelgeleme Problemlerinin Sınıflandırılması

Çizelgeleme problemlerini sınıflandırmak için aşağıdaki notasyonlar kullanılır. n/m/A/B burada;

n : İş sayısı

m : Makina sayısı

A : Problemin tipini gösterir. Eğer m=1 ise burası boş bırakılır. A; F,P,G,B değerlerini alabilir. Bunlar;

F : Akış tipi çizelgeleme problemi; bütün işler için makina sırasının aynı olduğu durumdur.

P : Permutasyon tipi çizelgeleme problem; bu durumda sadece makin aynı değil, aynı zamanda her bir makina için iş sırasının da aynı olduğu durumdur.

G : Genel atölye tipi çizelgeleme problemi; teknolojik kısıtları oluşumu üzerinde hiçbir kısıtın bulunmadığı durumdur.

B : Çizelgenin değerlendirileceği performans kriteri demektir.

2.2. Genel Atölye Tipi Çizelgeleme Problemi

Atölye-tipi üretim çizelgeleme probleminde; m tane $\{M_1, M_2,.....M_m\}$ makinada işlenmek üzere n tane $\{J_1,J_2,....J_n\}$ iş mevcuttur. Her bir işin, her bir makinada sadece ve sadece bir kez işlem gördüğü varsayılır. Makinada işin işlenmesine operasyon

denir ve i. işin j. makinadaki operasyonu O_{ij} olarak gösterilir. İşler, makinalarda belli bir sıra dahilinde işlenir ve bu sıra, teknolojik kısıt, iş seyri veya rota olarak adlandırılır. Genel atölye tipi üretim için teknolojik kısıtların oluşumuna dair hiçbir sınırlama yoktur. Her bir iş kendi işlem sırasına sahiptir ve diğer işlerin işlem sıralarından bağımsızdır.

Bununla birlikte bütün işler, aynı işlem sırasına sahip olduğundan özel bir durum ortaya çıkar. Böyle durumlarda problem akış-tipi çizelgeleme problemi olarak adlandırılır. Akış-tipi ve atölye tipi çizelgeleme arasındaki farklılıklar daha sonraki bölümlerde ayrıntılı olarak incelenecektir.

Her bir operasyon (O_{ij}), belli bir zaman uzunluğuna sahiptir. Bu zaman uzunluğu, işlem zamanı olarak adlandırılır ve p_{ij} olarak gösterilir. Basitleştirmek amacıyla, işlemin yapılacağı makinanın ayarlama ve hazırlık zamanının p_{ij} içinde bulunduğu varsayılır. Ayrıca işi makinaya taşımak amacıyla geçen zamanın da p_{ij} içinde bulunduğu varsayılır. Ayrıca p_{ij}' nin sabit ve önceden bilindiği varsayılır (French [1]).Yukarıdaki varsayımlara ek olarak, makinaların her zaman iş işlemeye müsait olduğu varsayılır. Fakat bu varsayım işler için geçerli değildir. Bazı işler, çizelgeleme başladıktan sonra bile işlenmek için elverişli durumda olmayabilir. i. işin işlenmek üzere atölyede hazır olduğu zamana i. işin hazırlık zamanı denir ve r_i ile gösterilir.

Genel olarak problem, işlerin makinalardan geçtiği bir sırasının bulunmasıdır. Bu sıra;

- Teknolojik kısıtlarla bağdaşır olmalı, yani olurlu bir çizelge olmalı,
- Bazı performans kriterlerine göre optimal veya optimale yakın olmalıdır.

Yukarıda bazı varsayımlar problemin tanımı içinde verildi. Bunlara ek olarak atölye tipi çizelgeleme problemini genelleştirmek için bazı tanımların yapılmasına gerek vardır. Bunlar şöyle sıralanır (French [1]).

1. **Her bir iş bütündür :** İş farklı operasyonlardan oluşmasına rağmen, aynı işin iki operasyonu hiçbir şekilde aynı anda işlenemez.

2. **İş Bölme Yoktur :** Her bir operasyon, başladığı zaman, diğer operasyon o makinada başlatılmadan önce tamamlanmalıdır.

3. **Her bir iş, her bir makinada bir tane olmak üzere, m tane farklı operasyona sahiptir :** İşin aynı makinada iki defa işlem görmesi olasılığı hesaba katılmaz.

4. **İş iptali söz konusu değildir :** Her bir iş tamamlanıncaya kadar işlenmelidir.

5. **İşlem zamanları çizelgeden bağımsızdır :** Burada iki durum varsayılmaktadır :

- Her bir hazırlık zamanı iş sırasından bağımsızdır. Yani işe ait makinayı ayarlamak için gereken zaman en son işlem gören işten bağımsızdır.

- Makinalar arasında işleri taşımak için gereken zaman ihmal edilebilmektedir.

6. **Ara stoğa izin verilir :** İşler bir sonraki makinanın boşalması için bekleyebilir. Bu önemli bir varsayımdır. Örneğin, çelik mil üretilirken demirin sıcak oluşundan dolayı iş, bir sonraki operasyon için beklemek zorunda kalabilir.

7. **Makinanın her bir tipinden sadece bir tane vardır :** İşleri işlenmesi sırasında aynı işi yapan birden fazla makinanın olmadığı varsayılır. Bu varsayım, beklemekten kaçınmak için belli makinaların çoğaltılması durumunu ortadan kaldırır.

8. **Makinalar boş kalabilir.**

9. **Hiçbir makina, aynı anda birden fazla operasyonu işleyemez.**

10. **Makinalar asla bozulmaz ve çizelgeleme periyodu boyunca kullanıma hazırdır.**

11. **Teknolojik kısıtlar önceden bilinir ve sabittir.**

12. **Rassallık söz konusu değildir. Özellikle ;**

- İşlerin sayısı bilinir ve sabittir,
- Makinaların sayısı bilinir ve sabittir,
- İşlem zamanları bilinir ve sabittir,
- Hazırlık zamanları bilinir ve sabittir,
- Belli bir problemi tanımlamak için gereken her türlü kalitatif değerler bilinir ve sabittir.

2.2.1. Performans kriterleri

Çizelgelemede amaçları ifade etmek her zaman kolay değildir. Amaçlar, çok karmaşıktır ve genellikle birbirleriyle bağdaşmazlar. Ancak, çizelgelemede ne derece başarılı olunduğunu karar vermek için bir takım kriterleri tanımlamak gereklidir. Aksi taktirde matematik olarak çizelge hakkında yargıya varmak zorlaşır.

Bazı durumlarda kararlaştırılmış teslim tarihlerine uymak tercih edilir. Aksi taktirde güvenilirlik kaybına uğranılacak ve maliyet söz konusu olacaktır. Bazen ise teslim tarihi çok önemli olmayabilir ve çizelgeleme periyodunun uzunluğu minimum yapılmak istenebilir. Çünkü bütün işler tamamlandıktan sonra makinalar başka işler için kullanılabilir. Böylece makinaların aylak (boş) kalma zamanları minimum olacaktır. Aykak makina, aylak sermaye yatırımı demektir. Bunlara ek olarak, stok maliyetleri minimum yapılmak istenebilir. Bununla sadece son ürünlerin stoklanması maliyeti kastedilmemekte, aynı zamanda makinalar arasında işlenmek üzere bekleyen yarı işlenmiş parçaların stoklanma maliyetler (ara stok maliyetleri) de kastedilmektedir.

Özetle, amaçlar çok farklı, çeşitli ve birbirleriyle çelişir olabilir. Ama çizelgeleme probleminin etkinliğinin ölçülmesi için tamamlanması gereklidir. Performans kriterlerini matematik terimlerle tanımlamadan önce bazı tanımlar ve notasyonlar aşağıda verilmiştir :

r_i : i işinin hazırlık zamanı

p_{ij} : i işinin j. makinada işlem zamanı

a_i : J_i işinin teslim süresi

q_i : J_i işine ait tahsisat. (Teslim süresi ve hazırlık zaman arasında işlenme için hesaba katılan zaman periyodu). $q_i = a_i - r_i$ (1)

W_{ik} : J_i işinin k'ıncı operasyonu için bekleme zamanı. k'ıncı operasyonla M_k makinası kastedilmemekte (k makina da olabilir), k. işlem sırasında gelen her hangi bir makine kastedilmektedir.

Böylece eğer J_i işinin teknolojik kısıtı $M_{j(1)}$, $M_{j(2)}$, $M_{j(3)}$, ...$M_{j(m)}$ ise, k. operasyon $Q_{ij(k)}$, $M_{j(k)}$ da işlem görendir. Böylece $W_{i(k)}$, J_i işinin $M_{j(k-1)}$ makinasında tamamlanması ile M_k makinasında işleme başlaması arasında boşta kalan zamandır.

W_i : J_i' nin toplam bekleme zamanı. $W_i = \sum_{k=1}^{m} W_{ik}$ (2)

C_i: J_i' nin tamamlanma zamanı. (Yani J_i işinin işlenmesinin bittiği zamandır).

$$C_i = r_i + \sum_{k=1}^{m} (W_{ik} + P_{ij(k)}) \quad (3)$$

F_i: J_i' nin akış zamanı (J_i işinin atölyede harcadığı zaman). $F_i = C_i - r_i$ (4)

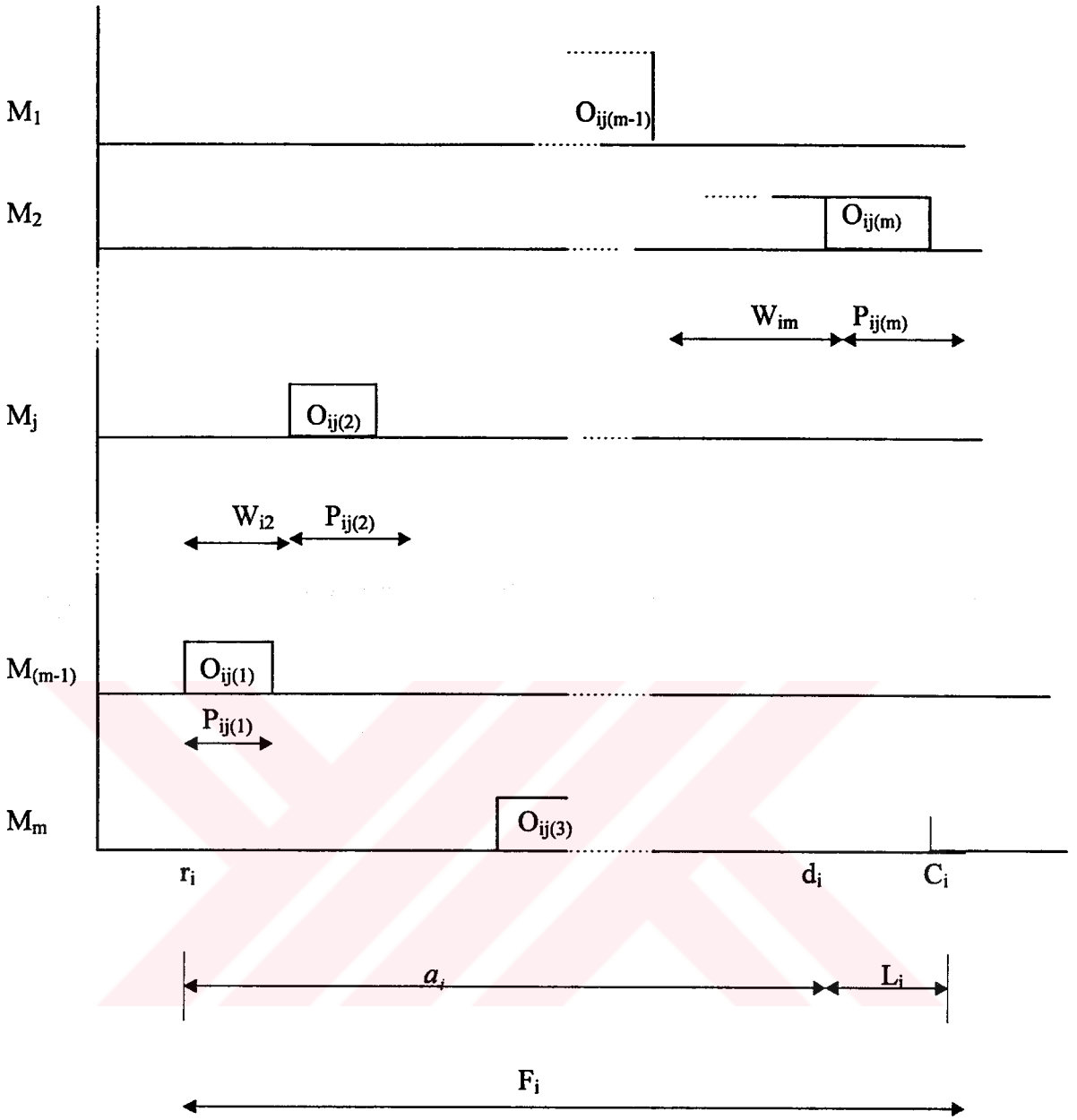
L_i: J_i' nin gecikmesi (İşin tamamlanma zamanı ile teslim tarihi arasındaki farktır

$$L_i = C_i - d_i \quad (5)$$

Eğer iş erken bitmiş ise, yani teslim tarihinden önce tamamlandıysa L_i negatiftir. Tam tersine eğer L_i pozitif ise, geç kalmış demektir ve buna işin pozitif gecikmesi denir. Böylece yeni iki kriter tanımlanmaktadır.

T_i : J_i işinin pozitif gecikmesi. $T_i = \max\{L_i, 0\}$ (6)

E_i : J_i işinin erkenliği. $E_i = \max\{-L, 0\}$ (7)



Şekil 2.1: Herhangi bir J_i işi için gant diyagramı. İşlem sıraları teknolojik kısıtlara bağlı olarak ($M_{(m-1)}$, M_j, M_m,..., M_1, M_2) şeklinde verilmiştir. Bekleme zamanları W_{i1}, W_{i3}' ün sıfır, W_{i2} ve W_{im}' nin ise farklı olduğu görülmektedir. Tamamlanma zamanı, teslim tarihinden sonra olduğundan $T_i=L_i$, $E_i=0$' dır.

Tipik bir iş için bu miktarlar Şekil 2.1.' deki Gant diyagramında gösterilmiştir. Burada zaman, hem bir zaman aralığını, hem de belli bir zaman noktasını göstermektedir. Bundan dolayı hazır zaman, tamamlanma zamanı, zamandaki bir noktayı ifade ederken, işlem zamanı, bekleme zamanı ve akış zamanı bir zaman aralığını gösterir.

Genellikle bu miktarların ortalamaları ve maksimum değerleri performans kriteri olarak ele alınır. Örneğin, $\overline{F}$ ortalama akış zamanını, C_{max} ise maksimum tamamlanma zamanını gösterir.

Diğer bir kriter olan M_j makinasındaki aylak zaman ise; $I_j = C_{max} - \sum_{i=1}^{n} p_{ij}$ şeklinde ifade edilir.

2.2.1.1. Tamamlanma zamanına dayanan kriterler

Buradaki temel kriterler, F_{max}, C_{max}, $\overline{F}$ ve $\overline{C}$' dir. F_{max}' ı enazlamak, çizelge maliyetinin doğrudan doğruya en uzun işle ilgili olduğunu göstermektedir. Maksimum tamamlanma zamanı C_{max}, çizelge maliyetinin işleri işleme sisteminin ne kadar zamanının, işlerin toplam kümesine ayrıldığına bağlı olduğunu gösterir. Hazırlık zamanlarının tümünün sıfır olması durumunda C_{max} ve F_{max} birbirine eşittir demektir. Gerçekte, eğer bir iş son derece geç hazır zamana sahipse, en kısa akış zamanına sahip olan iş C_{max}' da tamamlanır. Burada C_{max}'ı aynı zamanda toplam üretim zamanı veya yapım zamanı olarak da isimlendirmek uygun olacaktır. Ortalama akış zamanı $\overline{F}$ minimize etmek, çizelge maliyetinin doğrudan doğruya tek bir işi işlemek için geçen sürenin ortalamasıyla ilgili olduğunu ifade eder. Ortalama tamamlanma zamanı $\overline{C}$ minimize etmek, $\overline{F}$ minimize etmeye eşdeğerdir. Başka bir deyişle, minimum $\overline{C}$' ya erişen çizelge aynı zamanda minimum $\overline{F}$ da erişecektir. Bunun tersi de doğrudur. C_{max} ve F_{max} oldukça farklı performans kriterleri olmasına rağmen $\overline{F}$ ve $\overline{C}$ hemen hemen aynı şeyi ifade ederler. Bunun nedeni oldukça basittir. Bir sayı kümesinin maksimum değerini alan bir ifade, ortalamayı alan ifadelerden farklı değerlere sahip olacaktır.

Çok nadir görünmesine rağmen, bazı işlerin diğerlerinden daha önemli varsayımına dayanarak ağırlıklı performans kriterlerini kullanırlar. Bu durumda performans kriterleri aşağıda görüldüğü gibi ağırlıklı ortalamayı minimize etme esasına dayanır.

$\sum_{i=1}^{n} \alpha_i C_i$ *veya* $\sum_{i=1}^{n} \beta_i F_i$ n = İş sayısıdır.

Burada $\alpha_1, \alpha_2, \ldots\ldots \alpha_n$ ve $\beta_1, \beta_2, \ldots\ldots \beta_n$, toplamı 1' e eşit olan ağırlık faktörleridir.

2.2.1.2. Teslim tarihine dayanan kriterler

Çizelge maliyeti genellikle hedef tarihlerinin ne kadar yakalandığı ile ilgili olduğu için bu durumda uygun performans kriterleri sırasıyla ortalama gecikme ($\overline{L}$), maksimum gecikme (L_{max}), ortalama pozitif gecikme ($\overline{T}$), maksimum pozitif gecikme (T_{max}) olacaktır. $\overline{L}$'yı veya L_{max}'ı minimize etmek, bir işi erken tamamlamak için getiri söz konusu olduğu zaman uygundur. Bu durumda iş ne kadar erken tamamlanırsa getiri o kadar artacaktır. Tam tersine, $\overline{T}$ ve T_{max}' ı minimize etmek, erken tamamlanan işler herhangi bir getiri getirmiyorsa uygun olacaktır. Bu durumda ise geç kalan işler için ceza maliyetleri söz konusudur. Başka bir ifade ile, eğer ceza maliyetlerinden kaçınmak isteniyorsa $\overline{T}$ ve T_{max}' ı minimize etmek gerekmektedir.

Bazı durumlarda, bir işin geç kalmasıyla oluşan ceza maliyeti işin ne kadar geç kaldığına bağlı değildir. Bir dakika geç tamamlanan bir iş, yıllarca geç kalmış bir iş durumuna düşebilir. Bu gibi durumlarda uygun amaç, pozitif gecikmeli işler sayısını (n_t) minimize etmek olacaktır. Pozitif gecikmeli iş demek, daha önce ifade edildiği gibi teslim tarihinden sonra tamamlanan iş demektir.

2.2.1.3. Stok ve yarılanma maliyetlerine dayanan kriterler

Burada, işleme hazır olmayan veya makinalar arasında bekleyen işlerin ortalama sayısı ($\overline{N}_W$) veya tamamlanmamış olan işlerin sayısı ($\overline{N}_u$) minimize edilmeye çalışılır. Bu performans kriterleri, kabaca ara stok maliyetleriyle ilgilidir. Ayrıca tamamlanmış parçaların stok maliyetini azaltmak amacıyla $\overline{N}_c$ minimize edilmek

istenebilir. Eğer amaç, makinaların en verimli şekilde kullanılmasını sağlamak olursa, herhangi bir zamanda o an işlenmekte olan işlerin ortalama sayısını $(\overline{N_p})$ maksimize etmek amaçlanabilir. Alternatif olarak, makinaların etkin kullanımı amaçlandığında, ortalama aylak zamanı $(\overline{I})$ veya maksimum aylak zamanı (I_{max}) minimize edilmek istenebilir.

Son olarak, performans kriterleri, düzenli ve düzensiz olarak iki sınıfta incelenir. Düzenli kriter (R) , tamamlanma zamanlarında azalır olmayan bir kriterdir. Bu durumda R, C_1, C_2,....C_n' nin bir fonksiyonudur. Yani;

$C_1 \le C_1', C_2 \le C_2', ..., C_n \le C_n'$

Buradan

$R(C_1, C_2, ...C_n) \le R(C_1', C_2', ...C_n')$

elde edilir.

Bu tanımın temelindeki mantık şudur: İki çizelgeden birinci çizelgedeki bütün işlerin, ikinci çizelgedeki tüm işlerden hiçbir şekilde daha geç tamamlanmadığı varsayılır. Bu durumda, düzenli performans kriteri tanımı altında, birinci çizelge en az ikinci çizelge kadar iyi bir çizelgedir. Burada dikkat edilmesi gereken nokta düzenli performans kriterinin minimize edilmek istenmesidir.

$\overline{C}$, C_{max}, $\overline{F}$, F_{max}, $\overline{L}$, L_{max}, $\overline{T}$, T_{max} ve n_t hepsi düzenli performans kriterleridir.

2.3. Permutasyon Tipi İş Çizelgeleme

Bu tezde işlenen çizelgeleme tipi, permutasyon tipi sıralama olduğu için öncelikle bu çizelgeleme tipi hakkında bilgi vereceğiz. Permutasyon tipi sıralama, işlerin başlangıçta tümünün işlem görmeye hazır olduğu, her işin her makinada mutlaka operasyonu olduğu ve işlerin makinalardaki işlem görme sıralarının da aynı olduğu sıralama tipidir. Bu tip iş sıralama tekniği seri imalat yapan işletmelerde ortaya çıkan

problemlere çözüm üretmekte kullanılır. Bu tip üretimde genellikle makinalar bir hat boyunca dizilmiştir ve tüm işler bu hattı takip ederler.

Permutasyon tipi iş sıralama probleminde, atölyedeki tek kaynak kısıtı makinadır. İşler mevcut makinalara aynı sırada uğrarlar. Tüm işler t=0 anında işlenmeye hazır olduğu ve işlem sürelerinin bilindiği kabul edilir. Diğer varsayımlar ise şunlardır;

- Bir iş bir makinaya atandığında, bitirilmeden aynı makinaya başka bir iş atanmaz.
- Bir makinada bir operasyonun başlatılabilmesi için, işin bir önceki operasyonunun tamamlanmış olması gerekir.
- Makinalardaki aksama ve bozulmalar göz önünde bulundurulmaz.
- Makina hazırlama zamanları işlem sürelerinin içindedir.
- İşlerin makinalar arası transfer zamanı ihmal edilebilir.

Herhangi bir i. işin j. makinadaki işlem süresi P(i,j) ve iş sırası $\{J_1,J_2,....J_n\}$ olarak kabul edilirse işlerin makinalardaki tamamlanma zamanları $C(J_i,j)$ ve ona bağlı olarak maksimum tamamlanma (C_{max}) aşağıdaki gibi hesaplanır. Burada n işsayısı, m makina sayısıdır.

$C(J_1,1) = P(J_1,1)$

$C(J_i,1) = C(J_{i-1},1) + P(J_i,1)$ $\quad i = 2.....n$

$C(J_1,j) = C(J_1,j-1) + P(J_1,j)$ $\quad j = 2.....m$

$C(J_i,j) = \text{Max}\{C(J_{i-1},j), C(J_i,j-1)\} + P(J_i,j)$ $\quad i = 2....n, j = 2....m$

$C_{max} = C(J_n,m)$

Bu çalışmada, çeşitli sıralamaların toplam akış zamanlarını hesaplarken yukarıdaki C_{max} formülünü kullandık.

2.3.1. Permutasyon tipi çizelgeleme problemi çözüm metodları

Permutasyon tipi iş çizelgeleme problemlerinin çözüm algoritmalarına bakıldığında optimizasyon yöntemlerinin 3 ve daha az sayıda makina söz konusu olduğunda

çalıştıkları, makina sayısının 4 ve yukarısı olduğu durumlarda ise sezgisel yöntemlerin kullanıldığı görülür.

2.3.1.1. Johnson algoritması

Johnson algoritması permutasyon tipi iş çizelgeleme problemlerinde tamamlanma zamanına dayalı performans kriterlerine göre makine sayısına bağlı olarak optimal ve optimale yakın sonuçlar veren bir algoritmadır. İki makinada işlenmek üzere n iş ($n/2/F/F_{max}$ problemi) ve işlerin makinalara geliş sıralarının M_1,M_2 olduğunda maksimum akış zamanının nasıl enküçükleneceğini göstermektedir. Burada hazırlık zamanlarının sıfır olmasıyla $F_{max}=C_{max}$ olmaktadır. M_2 makinasında işleme en erken başlamak için; M_1 makinasında en küçük işlem zamanlı işlerin önce başlatılması uygun görülmektedir. Bir işin M_1 makinasında işlenmesi için M_1'in boş olması gerekmektedir. Johnson algoritması, iki makinalı permutasyon tipi iş sıralama probleminde, tüm bu kabuller bir arada düşünüldüğünde, M_1 makinasında kısa işlem süresine sahip olan işler son sıralara alınacak şekilde bir iş sırası (J_1, J_2,.....,J_n) aranmaktadır. Johnson tarafından geliştirilen algoritma bu özellikleri sağlamaktadır.

2.3.1.2. Sezgisel Yöntemler

Sezgisel yöntemler makina sayısının 4 veya daha fazla olduğu durumlarda kullanılırlar ve optimal çözümü garanti etmezler. Her ne kadar optimal veya optimale yakın sonuçlar üretebilmekte iseler de iş sayısı (n) ve makina sayısı (m) arttığı durumlarda sezgisel yöntemlerin etkinliği de azalmaktadır. Aşağıda bu konuda en çok kullanılan algoritmaların isimleri verilmiştir :

1. Palmer'in Eğim Dizini (Palmer Slope Index-PSI) yöntemi
2. CDS algoritması
3. Dannenbring algoritması
4. NEH (Nawaz, Enscore and Ham) algoritması
5. HC (Johny C. Ho Yih.Long Chang) algoritması

BÖLÜM 3. TAVLAMA BENZETİMİ

3.1. Tavlama Benzetimi Algoritmasının Yapısı

Tavlama Benzetimi (TB) metastrateji sezgisel algoritmasının kökeni, istatistiksel kavramlardır. Bu algoritmayla ilk olarak Kirkpatrick [2] ile Cerny [3] ilgilenmiştir. Onlar, kombinatoriyal optimizasyon problemlerini (KOP) çözmek için, problem ile katıların tavlama prosesi arasındaki bir benzerliğe dayanarak bir TB algoritması sunmuşlardır. Metastrateji ve kombinatoriyal optimizasyon kavramları aşağıda daha geniş bir şekilde açıklanacaktır. Bu bağlamda, yoğunlaşan maddenin fiziğinde tavlama; ısı yolu sıcaklığının yüksek bir değere ulaşması ile eriyen bir katının (kristalin) bir işlemidir. Burada kristalin bütün molekülleri, sıvı fazda kendilerini rastgele düzenler. Eriyen kristalin sıcaklığı, donana kadar düşürülür. Eğer soğumada dış sıcaklık hızlı bir şekilde sıfıra doğru düşüyorsa ve her sıcaklık değeri için denge sıcaklığına ulaşılmasına izin verilmiyorsa, kristal yapı yaygın düzensizliklere ve hatalara kilitlenebilir. Bu olay hızlı söndürme olarak tanımlanır ki durağan bir yapıdaki sonuçtur.

Bu benzerlikten hareketle, katı maddenin durumu bir kombinatoriyal optimizasyon probleminin mümkün çözümüne karşılık gelir; durumların (katı, sıvı, gaz) enerjisi, çözümlerin amaç fonksiyonuna karşılık gelir; en düşük enerji de optimum çözüme karşılık gelir. Hızlı söndürmeye, çok yüksekten azalma yolu ile yerel optimizasyona benzer bir işlem olarak bakılabilir. Böylece yerel optimizasyonda, artan hareketler engellenir ve algoritma yerel bir minimum olur. Kristaller pratikte büyüdüğünde, bu kötü bir yerel optimumdur ve dikkatli bir tavlama işlemi ile bu durumdan sakınılır. Bu işlemde seviyelerin her sırasında sıcaklık yavaşça düşürülür. Sıcaklık dengesine ulaşana kadar kristal erir.

Metropolis [4], T sıcaklığının sabit bir değeri için sıcaklık denge gelişiminin bir benzetimi olan Monte-Carlo metodunu sunmuştur. Bu metod, aşağıdaki şu yolda kristalin durumlarını ardarda oluşturur: Kristalin şimdiki durumu verilir (S) ve aynı zamanda moleküllerinin pozisyonları da karakterize edilir. Küçük bir düzensizlikte, rastgele seçilen bir molekülün yeri değiştirilerek sorun giderilir. Eğer şimdiki S durumuyla, yeni üretilen S' durumu arasındaki enerji seviyesi farkı negatif ise S' kabul edilir ve işleme yeni durumdan (S') devam edilir. $\Delta \geq 0$ olduğunda bir $\theta \in [0,1]$ rassal sayısı seçilir ve eğer $\theta \leq e^{-\Delta/T}$ ise S' kabul edilir. Aksi durumda S durumu yine geçerli durumdur. Bu kabul kuralı, Metropolis (veya benzetim) kriterine dayandırılır. Tavlama Benzetimi, azalan sıcaklığın bir tavlama (veya soğuma) çizelgesi ile birleştirilmesinde benzetim tekniğinin kullanılmasına dayanır.

Tavlama Benzetimi, özel bir algoritmadan daha ziyade, bir yaklaşım tekniğidir. Belirli bir kombinatoriyal optimizasyon probleminin her hangi bir yaklaşımında, tercihlerimizde birçok kararı almak zorundayız. Johnson ve arkadaşları [5], bu tercihleri, problemin uygunluğuna göre soğuma çizelgeleri için, özel tercihler ve genel tercihler olarak iki sınıfa ayırmışlardır. Genel tercihler, Tavlama Benzetimi soğuma çizelgesinin elemanlarını tanımlar. Bir soğuma çizelgesi, aşağıdaki sorulara kesin bir cevap vermek zorundadır:

(i) T sıcaklık parametresinin ilk başlangıç değeri,
(ii) Soğuma oranı ve sıcaklık kuralı,
(iii) Her sıcaklıktaki iterasyon sayısının sağlanması,
(iv) Algoritmanın sonu (durdurma kriteri).

TB algoritmasının performansı, seçilen soğuma çizelgesinin dayanıklılığına bağlıdır. Uygun bir soğuma çizelgesi ile birçok kombinatoriyal optimizasyon probleminde yakın optimal çözümlere ulaşılabilir. Soğuma çizelgelerinin teoride ve pratikte birçok çeşidi literatürde pek çok yazar tarafından sunulmuştur. Bu soğuma çizelgeleri, Osman [6] tarafından üç kategoriye ayrılmıştır:

(a) Sıcaklık düzenli bir şekilde adım adım düşürülür, iterasyon sayısı, düşürülen sıcaklıktan önce belirlenir.

(b) Sıcaklık düzenli olarak düşürülmeye devam edilir, her iterasyonda sıcaklık yenilenir.

(c) Sıcaklık düzensiz olarak düşürülür, her sıcaklıkta zaman zaman artan her iterasyondan sonra sıcaklık düşürülür.

3.1.1. Kombinatoriyal optimizasyon ve metastrateji kavramları

Tavlama Benzetiminin, bir kombinatoriyal optimizasyon tekniği olduğunu yukarıda belirtmiştik. Bu nedenle kombinatoriyal optimizasyonun tanımını yapmak daha iyi olacaktır. Kombinatoriyal optimizasyon, kesikli fonksiyonların en optimal bir biçimde düzenlenmesi, gruplanması, sıralanması ve seçilmesi için bulunan matematiksel bir uygulamadır (Lawler [7]).

Tipik kombinatoriyal optimizasyon problemleri şunlardır: Verilen pozisyonlarda binaların optimal yerleşimi, müşterilerin en iyi şekilde gruplanması, tezgahlardaki işlerin optimal şekilde sıralanması, işlerin ilgili yerlere optimal bir şekilde atanması, yatırım imkanlarının optimalinin seçilmesi. Kombinatoriyal optimizasyonun önemli referansları, Lawler [7], Papadimitriou & Steiglitz [8], Nemhauser & Wolsey [9]' dir. KOP' nin bir örneği, mümkün çözümlerin sınırlı bir kümesinin dolaylı bir tanımlanmasında görülebilir. Bir amaç fonksiyonuna bu çözümlerden her biri için bir değer atanır. Optimal çözüm bu mümkün değerlerin minimumudur (veya maksimumudur). KOP' ler çok iyi tanımlanırlarsa, mümkün çözümler boş küme olmadığı müddetçe daima optimal bir çözüm vardır. Bu problemlerin pek çoğunda optimal çözüm bulmak çok zordur. Bu nedenle bu problemlerde yaklaşık (sezgisel) algoritmalar önemlidir. Bundan dolayı büyük boyutlu problemlerde hesaplama zamanı arasında uygun bir yaklaşık optimal çözüm sağlanabilir. Biz KOP' nin

çözümünde de bu sezgisel algoritmalardan Tavlama Benzetimi (TB) metodunu inceleyeceğiz.

Kombinatoriyal kelimesinin yerine bazen kesikli sözcüğü, optimizasyon kelimesinin yerine de programlama sözcüğü kullanılır. Bir KOP, mümkün çözümlerin bir kombinatoriyal (veya kesikli) kümesi üzerindeki bir amaç fonksiyonunun minimum (veya maksimum) bulmak için sorulur.

Bir KOP aşağıdaki şekilde tanımlanır :

KOP : Minimum f(x)

Kısıtlar $x \in X$

$X \subseteq \Omega$ ise, X, Ω uzayında mümkün bir çözüm olarak tanımlanır. Υ fonksiyonu gerçeklerin kümesi ise, $f : X \rightarrow \Upsilon$ amaç fonksiyon olarak adlandırılır. Bir mümkün çözüm $x \in X$ ise, başka bir $y \in X$ mümkün çözümü yoksa bu optimal bir çözümdür. X ve Ω kümeleri özel yaklaşımlar için uyumlu formların bir çeşitliliğini alır. Eğer X ve Ω kümeleri kombinatoriyal veya kesikli ise problem bir KOP' dir. Örneğin, a elemanlı sınırlı bir küme ailesi 2^a sınırlı eleman içerir. Bir KOP, araç rotalama probleminde olduğu gibi çok basit formda minimum maliyetli teslim rotalamanın bir kümesinin dizaynını, merkezi bir deponun meydana getirilmesini ve bitirilmesini, müşteriler kümesi olan I = (1,.....,n) kümesine hizmet sunmak için, değişken boyutlu hızlı araçlar kümesini V =(1,.....,v) içerir. Her müşteri, bir rotada işlenilen bir araç tarafından temin edilir. Her rotada müşterilerin toplam talepleri araç kapasitesini geçmemelidir.

Normal olarak, KOP' ni tanımlamak kolay, fakat çözmek zordur. Buna geçtiğimiz on beş yılda "Karmaşık Hesaplama " teorisinin gelişimi, bu ve diğer KOP' nin zorluğuna ilginç bir anlayış getirdi. Bu teori, zor veya kolaylığa göre sınıflanan problemler ve değerlendirilen algoritmalar için özenle sağlanan metodlara katkıda bulundu.

Yukarıda TB metodunun bir metastrateji sezgisel algoritması olduğunu belirtmiştik. Metastrateji sezgisel algoritması, yaklaşık çözümler için bir bilgisayar metodudur. Zor, büyük ve karmaşık KOP' nin optimal çözümlerini en etkili bilgisayar kullanılsa bile hesaplama zamanı gerçekçi olarak bulunmayabilir. Böyle bir algoritma, performansları arttırmak için farklı hareket ve organizasyonu çalıştırmada etkili olan yerel tarama metodu gibi alt bir tarama ile organize edilir ve yönetilir.

Geliştirilmiş bu sezgisel metodlarla elde edilen yerel optimum sonuçlar, arama mekanizmalarına ve mümkün çözümlere oldukça bağlıdır. Bu yerel optimum genellikle düşük kalitededir. Metastrateji yenilikçi yaklaşımlar, bu zayıf yerel optimumlarda yanlış sonuçlardan sakınmak için geliştirilir. Bu yaklaşımlarda, yerel minimum bir sonuçtan sonra yerel taramaya devam ederek, global bir minimum ortaya çıkarılabilir. Bundan sonra yerel tarama tekniklerinin versiyonları arttırılarak işleme devam edilebilir. Bu yaklaşımlar, aynı zamanda yeni problem alanlarına uygulandığında yaklaşık optimal çözümlerin şansını arttırır. Bu algoritmalardan son zamanlarda geliştirilen bazı sezgisel metodlar şunlardır:

1.Genetik algoritmalar

2. Yapay Sinir Ağları

3. Liste Arama (Tabu Search)

4. Tavlama Benzetimi

Biz bu tezde metastrateji sezgisel algoritmalarından Tavlama Benzetimi konusunu ele aldık.

3.1.2. Kombinatoriyal optimizasyon problemlerinde Tavlama Benzetimi metodunun performansı

Tavlama Benzetimi metodu 1983'den itibaren literatüre girmesiyle, birçok zor kombinatoriyal optimizasyon probleminde ve çeşitli alanlarda uygulandı. Kombinatoriyal optimizasyon problemine, Tavlama Benzetimi metodunun son yıllarda uygulandığı alanlar olarak, akış tipi üretim çizelgeleme problemleri için Osman [10]' un çalışmalarını, Kapasite Gruplama Problemleri (CCP) için Osman [6]' ın çalışmalarını, yine Osman [11]' ın Genel Atama Problemlerini (GAP) ve Osman' ın [12] Araç Rotalama Problemlerini (VRP) sayabiliriz. Tavlama Benzetimi metodunun teoride ve uygulamadaki başlıca kitap ve incelemeleri, Van Laarhoven [13], Collins [14], Aarts [15], Johnson [16] ve Eglese [17] sayılabilir.

3.2. Tavlama Benzetimi Metodunun İşleyişi

Önceki tanımlamalarda da belirtildiği gibi bütün Tavlama Benzetimi terminolojisi ve onun istatistiksel kavramlarla benzerliğinden hareketle TB, genelleştirilmiş tekrar edilen bir gelişim algoritması olarak görülebilir. TB algoritmasının adımları aşağıdaki gibidir :

ADIM 1 : Başlangıç bir rassal veya sezgisel bir S çözümü oluşturulur. T sıcaklığı için bir başlangıç T_s sıcaklığı ve diğer genel parametreler oluşturulur.

ADIM 2 : $S' \in N_\lambda (S)$ olmak üzere bir S' çözümü seçilir. $\Delta = C(S') - C(S)$ amaç değerlerinin farkları hesaplanır. Burada S' bir çözüm seti ve N_λ (S)'de çözüm uzayıdır.

ADIM 3 : Eğer ;

(i) S' , S ($\Delta < 0$) dan daha iyi veya S' , S den daha kötü, fakat şimdiki sıcaklıkda rassal işlem tarafından kabul edilebilir , yani $e^{-\Delta/T} > \theta$ ($0 < \theta < 1$) ise seçilir ve S' , S nin yerini alır.

Değilse ; Geçerli çözüm hala S dir.

ADIM 4 : Bir kurallar kümesine bağlı olarak T sıcaklığı yenilendiğinde şunları içerir:

(i) Soğuma çizelgesi kullanılır,
(ii) Yukarıdaki üç adımda gelişme olup olmadığına bakılır,
(iii) N_λ (S) ' de yakın bir çözüm olup olmadığı bütünüyle araştırılır.

ADIM 5 : Eğer "Durdurma Testi" başarılı ise işlem durdurulur, değilse ADIM 2 ye gidilir.

3.3. Tavlama Benzetimi Metodunun Uygulandığı Alanlar

3.3.1. Çizelgelemede Tavlama Benzetimi

Çizelgeleme problemleri, TB daha çok uygulandığı kombinatoriyal problemlerdir. TB, pratikteki bazı çizelgeleme problemleri kadar iyi sonuçlar veren tek-makina tipi üretim, akış tipi üretim, atölye tipi üretim ve hücre tipi üretim tekniklerine uygulanmıştır.

3.3.1.1. Tek-makina üretim çizelgeleme

Potts [18] tek-makina ağırlıklı gecikme problemlerini çözmek için TB ve diğer sezgisel metodları karşılaştırdı. Tek-makina ağırlıklı gecikme problemlerinde amaç;

$\min \sum_{i=1}^{n} w_i T_i$ 'dir.

Burada T_i, i işinin gecikmesi, w_i, i işi için atanan ağırlıktır. Bu problemde kısıt yoktur.

Potts, TB tekniği ile çalışan iki sezgisel metod kullandı. Bunlardan biri, SALM denilen genel bir sezgisel metoddur. Bu, soğuma cetvelinde kullanılması için Lundy [19] tarafından geliştirilen bir metoddur. Diğer sezgisel metod SAPW problem parametrelerine iyi bir şekilde uyarlanır. Bu parametreler, azalan adımlar ve tavlama aralığıdır. SALM sezgisel metodunda Z sıcaklığı k+1. iterasyon için şöyle hesaplanır;

$$Z_{k+1} = \frac{Z_k}{(1+\beta Z_k)}$$

Z' nin başlangıç ve son değerleri sırasıyla 1.000 ve 0.334' dür. β' nın değeri iterasyon sayısına bağlıdır (n = 50 için 10.000, n = 100 için 20.000). Başlayan işin sıralaması, rastgele ve komşu çözümler seçilir. Bu çözümler sıralamada rastgele seçilen iki işin değiştirilmesiyle oluşur.

Diğer sezgisel metod, SAPW, teslim tarihlerinin sıra ile karşılaştırılması- ki burada TTS (Teslim tarihi sırası) ≥ 0.4 ve bir gecikme faktörü ile - ki burada GF (Gecikme Faktörü) problemler için en erken teslim tarihi sezgisel algoritması tarafından oluşturulan işlerin sıralanması ile başlar. Gecikme faktörü, teslim tarihi sıkışıklığının bir ölçümüdür. Gecikme faktörünü (GF) ve teslim tarihinin sıra karşılaştırılmasını (TTS) çok görürüz.

Tek-makina problemlerinde erken ve gecikme cezalandırması doğrusal olmayan ceza fonksiyonlarına sahiptir. Böylece azalan iş tamamlama zamanları ceza fonksiyonunu arttırabilir. Bu problemlerde optimal çizelge V düzenindedir. Mittenthal [20] göstermiştir ki, TB ile birleştirilen iyi bir sezgisel metod, bu tür problemler için önceden sunulan sezgisel metodlara göre daha iyidir .V düzenindeki çizelgede yakın çözüm, çizelgenin iki kolu arasında değiştirilen işlerle bulunur. Mittenthal, çeşitli

soğuma oran değerlerini deneyip, sonuçlandırdığında, en uygun soğuma oranının 0.8 olduğunu görmüştür.

3.3.1.2. Akış tipi üretim çizelgeleme

Osman [10] bir akış tipi üretim çizelgeleme problemini TB olarak formüle etmiştir. Akış tipi üretim çizelgeleme probleminde n iş, m makina vardır. Bütün işler, önceden kararlaştırılan aynı sırada 1,2, ..., m makinayı dolaşır. Burada amaç, kısıt altındaki maksimum tamamlama zamanını minimize ederek işleri sıralamaktır. Bir iş her hangi bir makinada başlayabilir. Belli bir sıcaklıkta yapılan iterasyon sayısı yerine, sıcaklıklarda (Z) iterasyon sayılarını eşit olarak aldılar. Z' nin başlangıç değeri şöyle hesaplanır :

$$Z_1 = \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{p_{ij}}{(5mn)}$$

p_{ij} : Makinadaki i işi için işleme zamanı

m : Makina sayısı

n : İş sayısı

K. cı iterasyon için son sıcaklık ;

$$Z_k = 1$$

Osman dört tane Tavlama Benzetimi sezgisel metod geliştirmiştir.

Metod 1 : Birbiriyle değişme metodu. Bu algoritma da sıralamada rassal olarak seçilen iki iş birbiri ile yer değiştirir.

Metod 2 : Komşu işlerin yer değiştirmesi metodu. Burada da iki tane komşu iş birbiriyle yer değiştirir.

Metod 3 : Sıralı tarama metodu. Burada birbiriyle değişebilen bütün mümkün işler denenir.

Metod 4 : Rassal tarama metodu. Burada birbiriyle değişebilen bütün mümkün işler rassal olarak yapılır.

Ogbu [21] akış tipi üretim çizelgeleme problemini çözmek için TB tekniği uygulamıştır. Başlangıç sıralamaları için Palmer [22]' ın eğim indeksini ve Dannenbring [23]' in sezgisel metodlarını kullanmıştır. Yakın çözümler, başlangıç sıralamasında değiştirilme ve değiştirme operasyonları kullanılarak oluşturuldu. Kabul edilen sıcaklık olasılığı 0.20 den 0.0026 ya kadar geometrik olarak azaldı. Birbiriyle değiştirme yakın çözüm tekniği kullanıldığında, her sıcaklıkta iterasyon sayısı, 15 işten daha az problemler için $8(n_1)$, 15 iş problemi için $10(n_1)$ dir. Birbiriyle değiştirilenlerin sayısı, değiştirme tekniği kullanıldığında $5(n_2)$ ve $7(n_2)$ dir.

$$|n| = \frac{(n^2 - n)}{2}$$

$$|n2| = (n^2 - 2n + 1)$$

Burada n problemdeki iş sayısıdır. Ogbu TB tekniğinin daha iyi olduğunu bulmuştur. Bu iki TB tekniği arasında da, değiştirme tekniğinin büyük yapıdaki problemler için daha iyi performans sağladığı görülmüştür.

3.3.1.3. Atölye tipi üretim çizelgeleme

Van Laarhoven [24] atölye çizelgeleme problemleri için maksimum tamamlama zamanını minimum yapmak için TB kullanmıştır. Atölye çizelgeleme için aşağıdaki tanımlamanın yapılabileceğini düşünmüştür. J işlerin bir kümesi, M makinaların kümesi olarak verilsin. Her iş O operasyon silsilesini içersin. Bu operasyonlarda, verilen bir makinada, verilen bir uzunluğun aralıksız bir zaman periyodunda yapılan işlemleri gereklidir. Her makina bir zaman biriminde en fazla bir operasyonu yerine getirir. Bir çizelge, makinalarda zaman aralıklarına operasyonların bir dağıtımıdır. Problem minimum uzunluktaki çizelgeyi bulmak içindir. Eşitlik aşağıdaki gibidir :

$$\min \max_{v \le 0} S_v + t_v$$

$$s.t \quad S_v \ge 0 \quad \forall\, v \in O$$

$$S_w - S_v \ge t_v \text{ eğer } v \rightarrow w, \quad v, w \in O$$

$$S_w - S_v \ge t_v \qquad S_v - S_w \ge t_w \quad \text{eğer } M_v = M_w \quad v, w \in O$$

Burada S_v, t_v, v operasyonunun başlama ve işleme zamanlarıdır; $v \rightarrow w$ sıralamayı gösterir ki v, w den önce gelir; M_v ve M_w, v ve w operasyonlarının yapıldığı makinalardır.

Van Laarhoven, makine permutasyonlarının bir kümesinin aşağıdaki konfigürasyonla tanımlamıştır. Burada her parça sıra ile k makinasındaki operasyonlarda işlenir.

$$\Pi = \{\Pi_{i1} \ldots\ldots \Pi_{im}\}$$

Konfigürasyon sayısını şu bağıntı verir :

$\prod_{k=1}^{m} m_k !$ Burada m_k, k makinası ile işlenmiş olan operasyonların sayısıdır.

Yakın çözümler geçişlerle oluşturulur. Bir geçiş, bir kaç k makinasında ard arda gelen iki operasyonun değişimi ile oluşturulur. Bir artırımın kabul edilme olasılığı exp (- (C(j) - C(i)) / c ile verilir. Burada C(i), i. ci konfigürasyonla birleştirme maliyeti; c, derece derece azalan kontrol parametresidir.

3.3.1.4. Hücre tipi üretim çizelgeleme

Vakharia [25] hücresel imalat sistemlerde, çizelgeleme problemlerini çözmek için dal-sınır algoritması ve farklı iki tane sezgisel algoritma ile TB performansını

karşılaştırdı. Hücresel imalatta bir hücre, benzer işlenen ihtiyaçlara tahsis edilmiş makinaları içerir. Problem parça ailesindeki işlerin çizelgelemesini ve işlenmiş olan parça ailelerini kapsar. Burada amaç maksimum tamamlama zamanını minimize etmektir. Yakın çözümler komşu işlerin ve/veya ailelerin değiştirilmesi ile oluşturulmuştur. Soğuma çizelgesi iterasyon sayısının monoton bir şekilde azalan bir fonksiyonu olarak kullanılmıştır. Vakharia 10 tane rassal oluşturulan problemleri çözerek diğer bilinen sezgisel metodları TB ile karşılaştırdı. Sonuçta görülmüştür ki, büyük boyutlu problemler için, TB metodu daha iyi çözümler vermiştir.

Ayrıca Sridhar [26] hücresel imalat sistemlerine Tavlama Benzetimi yaklaşımını toplam akış zamanının minimizasyonuna uygulamıştır. Diğer klasik metodlar ve sezgisel algoritmalarla bu yaklaşımı karşılaştırmıştır. Sonuçta, Tavlama Benzetimi metodu ile toplam akış zamanının daha minimum olduğunu görmüştür.

3.3.2. Genel rotalama problemlerinde Tavlama Benzetimi

3.3.2.1. Gezgin satıcı problemi (GSP)

Cerny [3] gezgin satıcı probleminde (GSP) yaklaşık çözüm bulmak için TB kullanmıştır. Amaç fonksiyonu olarak gezgin satıcının toplam katettiği mesafenin minimizasyonunu kullanmıştır. Kısıt olarak her şehir en az bir kere ziyaret edilecektir. Cerny, N tane istasyon için GSP problemini D olarak tanımlanan N*N matrisi ile i. istasyondan j. istasyona kadar ki mesafede D (i,j) elementlerinden oluşan bir N*N matrisi olan D tanımlamıştır. Burada ki problem, minimum toplam uzunluk d için bir c permutasyonunu bulmaktır:

$$d = D(c_N . c_1) + \sum_{k=1}^{N-1} D(c_k . c_{k+1})$$

Cerny, optimal çözüme oldukça yakın çözümler elde edebileceğini, sunduğu algoritmayı kullanarak birkaç örneklerle göstermiştir. Bazı durumlarda, tam optimum

bulunmuştur. Örneklerin çoğunda ve eldeki diğer metodlarla karşılaştırma olmadığından, büyük iterasyon sayıları ile çözüm elde edilebilmiştir.

Golden [27] GSP' ne TB uygulamıştır. Komşu çözümler, turda rastgele seçilen iki yayın değiştirilmesi ile üretildi. Sağlanan yeni çözüm, aykırı bir turda sonuçlanmadı. Basit bir tavlama çizelgesi kullanıldı. Onlar keyfi bir yüksek sıcaklıkla başladılar ve daha sonraki sıcaklıklar 25 eşitlikte parçalara ayrılarak elde edildi. Golden'ın deneyleri gösterdi ki, GSP için CCAO sezgisel metodu, Tavlama Benzetimin'den çok daha hatasız ve daha hızlıydı. Sonuçta gördüler ki, (i) Hesaplama kuvveti pratikten daha hızlı büyüyen Tavlama Benzetimi uygulaması ile ilişkilendirilir; (ii) Çalıştırılma zamanları ve doğruluğu, kontrol parametrelerine tam duyarlı gibi göründü. CPU zaman miktarını karşılaştırmak için 2-OPT işleminin tekrarlanan uygulaması TB' den daha iyi sonuç verdi.

Randelman [28], GSP ' ne uygulanan TB' de birçok teorik uygulamaları almıştır. Randelman, turun uzunluğu TB' de soğuma oranı logaritmaya bağlı olduğunu göstermiştir.

Aarts [29] aynı zamanda GSP' ne uygulanan TB' nin teorik kısımlarını da ele almıştır. Aarts soğuma çizelgesindeki kontrol parametrelerinin bir fonksiyonundaki maliyetinin değişimini ve bekleyişini analiz etmiştir. 100 şehirde uygulanan TB algoritmasının sonuçlanan sayısal verilerini sunmuştur. Sonuçlandığında gördü ki; Tavlama Benzetimi algoritmasının ortalama durum performansını, gama dağılımına uyan optimum maliyetten sonuç maliyetine sapması ile açıklayabilmiştir.

3.3.2.2. Araç rotalama problemleri (ARP)

Alfa [30] araç rotalama problemlerini çözmek için Tavlama Benzetimi ile 3-opt sezgisel metodunu birleştirmiştir. Problemin amacı, aracın gezdiği toplam uzaklığı minimize etmekti. Dağıtım yapılan n tane düğüm noktası olsun. d_i i düğümlerindeki talep olsun (i=1,2,.....,n). i ve j düğümleri arasındaki tur maliyeti (mesafe veya tur

zamanı) t_{ij} dir ($t_{ij} < \infty$); $t_{ij} = \infty$ tanımlıdır. Eğer her k aracı, bir c_k kapasitesine sahipse, o zaman problemi birçok K aracı için tanımlamak gerekirse;

$$Enküçük\ Z = \sum_{k=1}^{K}\sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij} X_{ij}^{k}$$

Kısıtlar

$$\sum_{k=1}^{K}\sum_{i=1}^{n} X_{ij}^{k} = 1 \qquad \forall_j$$

$$\sum_{k=1}^{K}\sum_{j=1}^{n} X_{ij}^{k} = 1 \qquad \forall_i$$

$$\sum d_i X_{ij}^{k} \le c_k \qquad \forall_k$$

$$\sum_{i=1}^{n} X_{ip}^{k} - \sum_{j=1}^{n} X_{pj}^{k} = 0, \quad \forall p,k$$

$$X_{ij}^{k} = \{0,1\}$$

Burada X_{ij}^{k}, izin verilen alt turlar olarak tanımlanır. Araç rotalama probleminin birleştirilen metodla çözülmesi aşağıdaki prosedüre dayandırılır:

1. Büyük bir mümkün turla başlanır, bütün düğümlere bir turda uğranılacağı varsayılır.

2. Her iterasyonda (I, N +I) arasında üç ayrı rassal sayı üretilir. Burada N, depo sayısı, I iterasyon sayısıdır.

3. Yapılan turda rassal olarak üç hat seçilir. 3-opt. kuralı ile en çok sekiz tane yeni çözümler üretilir ve en iyi bir tanesi seçilir.

4. Eğer seçilen yeni çözüm şimdiki çözümden daha iyiyse, o zaman yeni çözüm bu kabul edilir ve devam edilir; aksi halde TB uygulamasına göre kesin bir yeni çözüm kabul edilir.

ARP için uygulanan Tavlama Benzetimi ile 3-opt. sezgisel metodunun birleştirilmesi ile, yazarlar iki durumda (N = 30 ve N = 50) çok iyi bilinen 3-opt sonuçları kadar iyi sonuçlar elde edebildiler.

3.3.3. Planlama problemlerinde Tavlama Benzetimi

3.3.3.1 Genel fabrika planlama problemleri

Jajodia [31] hücresel imalat ortamında hücreler arası ve hücreler dışı planlama problemlerine TB yaklaşımını uygulamıştır. Hücreler arası planlama problemi, atölyedeki alanlara makina hücrelerinin yerleştirilmesini sağlar. Hücre dışı planlama problemi, her hücredeki makinaların planlamasını sağlar. Burada amaç, bu hücreler arasında parçaların imal edilmesinde toplam gezilen mesafeyi minimize etmektir. Aşağıdaki eşitlikte, problem, minimize edilmiş k hücrelerinin ilgili pozisyonlarının sınırlaması olarak başlatılabilir.

$$Z = \sum_{i=1}^{k} \sum_{j=i+1}^{k} T_{ij} d_{ij} \qquad \text{k= Hücre sayısı}$$

Burada T_{ij}, m_i, ve m_j hücreleri arasındaki transfer edilen palet sayısıdır ve d_{ij} uzaklıktır. Uzaklıklar, aşağıda tanımlandığı gibi 'Manhattan' uzaklığı kullanılarak hesap edilmiştir:

$$d_{ij} = |x_i - x_j| + |y_i - y_j|$$

Burada (x_i, y_i) ve (x_j, y_j), m_i ve m_j hücrelerinin geometrik merkezlerinin koordinatlarıdır. İlk çözüm bir k*k matrisindeki hücrelerin rastgele yerleştirilmesi ile oluşturulur. Komşu çözümler, bir diğer kullanılmayan önceki pozisyona bir hücrenin hareketi ile veya herhangi iki hücrenin pozisyon değiştirmesi ile oluşur. İterasyon sayısı çözüm sayısı ve adım sayısı göz önüne alındığında azalan sıcaklık sırasıyla, 100k, 10k, 100' dür. Burada k problemdeki hücre veya makina sayısıdır. Sıcaklık geometrik olarak azalır. Çözüm kalitesi, Kusiak [32] tarafından sunulan alt sınır için son amaç fonksiyon değerinin oranı olarak tayin edilmiştir. Kusiak sekiz problemlik

bir kümede TB ile diğer bilinen sezgisel algoritmaları karşılaştırmıştır ve TB'nin performansının diğer metodlardan daha iyi olduğunu bulmuştur. Kusiak aynı zamanda TB algoritmasının, parametrelerin ilk seçimlerine veya problemin trafik (düzen) şartlarına duyarsız olduğunu bulmuştur.

Heragu [33] tek-sıralı ve çok-sıralı planlama problemlerine TB uygulamıştır. Planlama problemlerinin amaçlarından biri de yapılar arasındaki taşıma malzemeleri için, malzeme iletişim sistemleri tarafından ihtiyaç duyulan toplam zamanı minimize etmektir. Heragu basit TB'nin 2-yol ve 3-yol algoritmalarındaki değişen cezalar ile beraber Karışık Tavlama Benzetimi (KTB)'ni göz önüne almıştır. 2-yol değişim algoritması, ilk çözümdeki iki pozisyon arasındaki değişiklikleri göz önüne alır. 3-yol değişim algoritması ilk çözümdeki üç pozisyon boyunca olan değişiklikleri göz önüne alır. Değişen ceza algoritması, Heragu [34]'nun kısıtlı ABS modelini bir ceza metodunun bir kez kullanılması ile kısıtsız hale dönüştürdü.

Kullanılan parametre değerleri, ilk sıcaklık = 999.0, soğuma oranı = 0.90, iterasyon sayısı = 100n ve her adımda kabul edilen çözüm sayısı = 10n dir. Burada n, problemdeki binaların sayısıdır. Heragu 850' den daha fazla problem çözmüştür ve KTB algoritmasının, bazı tek-sıralı problemlerin daha önceki çözümlerinden daha iyi çözümler ürettiğini bulmuştur. Diğer problemler için KTB en iyi bilinen çözümleri üretti. Bununla beraber, KTB karşılaştırılan diğer metodlar için daha fazla hesaplama zamanına ihtiyaç duydu.

3.3.3.2. Esnek imalat sistemlerinde (EİS) planlama problemleri

Kouvelis [35] EİS ortamında tek sıralı planlama problemlerini (TSPP) çözmek için TB ni kullanmıştır. TSPP de parçalar sisteme soldan girer ve sağ taraftan ayrılırlar. Problem, iş istasyonlarının bir sıralamasını bulmaya yöneliktir. Aşağıdaki eşitlikte, problem, atanan α (α_1, α_2,.....α_N) vektörünü bulmaktır. Burada α_i, i istasyonlarının yeridir. Amaç fonksiyon aşağıdaki gibidir:

$$\min f(x) = \sum_{i=1}^{N}\sum_{j=1}^{N} W_{ij}\, d(\alpha_i, \alpha_j)\, L(\alpha_i, \alpha_j)$$

Burada W_{ij}, i istasyonundan j istasyonuna kadar yol alınan parçaların sayısıdır; d (α_i, α_j), i ve j istasyonları arasındaki mesafedir ve

$$L(\alpha_i, \alpha_j) = \begin{cases} 1 & \text{eğer } \alpha_i > \alpha_j \\ 0 & \text{diğer} \end{cases}$$

İlk çözümler, Kouvelis tarafından sunulan sezgisel algoritma ile elde edilmiştir. Komşu çözümler, iki pozisyonun rassal değişimi ile oluşturuldu. Kouvelis, komşu örnek oranlarının (0.1-0.95), farklı iterasyon sayısının (0.1-0.95), farklı soğuma oranlarının (0.1-0.9) ve farklı başlangıç olasılıklarının farklı değerlerini kullanarak simulasyon sayısını elde etmiştir. Kouvelis, simulasyon sonuçlarının yardımı ile planlama problemlerine TB'nin uygulanması için mümkün bir ilk değer sunmuştur.

Koulevis [36] kısıtlarla ayrılmış alanlarla planlama problemlerine TB'ni uygulamıştır. Pozitif alanlara ayrılmış kısıtlar, her siparişten sonra yer ayrılan makinalara ihtiyaç duyarlar. Negatif alanlara ayrılmış kısıtlar, yakın çevrede olmayan makinalara ihtiyaç duyarlar. Bir çözüm konfigürasyonu aday yerlere makinaları atar. Komşu çözümler, makina yerlerinin 2-yol değişimi yöntemi ile oluşturulur. Bir konfigürasyonun maliyeti, bağlayıcı maliyetini ve malzeme iletişim maliyetlerini kapsar. TB' nin iki yaklaşımı ele alınmıştır. İlkinde, başlangıç aday çözümü alanlara ayrılan kısıtlara dahil edilmiştir. Komşu çözümler hem değiştiği gibi, hem de bölgelere ayrılan kısıtlar kuralı ihlal etmediler. İkinci yaklaşım olarak, Başlangıç çözümü, bölgelere ayrılan kısıtları dahil etmediler, fakat kuralı ihlal eden bölgeye ceza tayin ettiler. Kouvelis, farklı başlangıç kabul edilen olasılıkları (0.05-0.95), soğuma oranlarını (0.1-0.99), incelenen komşu çözüm oranlarını (0.05-0.95) ve iterasyon sayılarını kullanarak simulasyonları çalıştırmıştır. Bu simulasyon analizlerine dayanarak, bölgelere ayrılmış kısıtlarla planlama problemleri için parametre değerlerinin bir kümesini kullanılmasını sunmuştur. Koulevis aynı zamanda bölgelere ayrılmamış kısıtlarla planlama problemleri için de bir fikir ortaya koymuştur. Kısıtlar, en uzak

akış oranı ile veya bu oranlarla birleştirilen maliyetlerle makinalar için oluşturulabilir. Bu işlem öncesi ve TB'nin uygulanışı çözüm etkinliğini önemli derecede ilerletebilir.

Sharpe [37], TB ni yapı planlama problemlerinin çözümünde de kullanmıştır. Yapı planlama problemleri, konumlara bir yapıdaki modüllerin atanmasını sağlar. Bu yapılar, modüller arasındaki aktiviteyi (Buna malzeme nakli ve insan hareketleri dahildir) minimize eder. Komşu çözümler, rassal olarak değişen iki modül tarafından oluşturulmuştur. Scriabin [38] ile Nugen [39] iyi bilinen problemlerine TB uygulamasında hızlı ve yavaş soğumayı uygulamışlardır. Ve sonuçta yavaş soğuma şartları altında daha iyi çözümler üretmişlerdir. Bu teknik, mevcut tekniklerin en iyisiydi.

3.3.4. Karesel atama problemleri (KAP)

Birçok yapı yerlere atanmak zorunda olduğunda ve yerine bağlı olarak yapılar arasında etkileşim olduğunda, planlama problemlerinde ve yapı yerleşimlerinde KAP' tan yararlanma İhtiyaci doğmuştur. KAP Aşağıdaki şekilde formüle edilir :

$$\min f(x) = (1/2)\sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{j=1}^{n}\sum_{h=1}^{n} c_{ik.jh}.x_{ik}x_{jh}$$

Kısıtlar

$$\sum_{i=1}^{n} x_{ik} = 1, \quad k = 1,2,.....,n$$

$$\sum_{i=1}^{n} x_{ik} = 1 \qquad i = 1,2,.....,n$$

$$x_{ik} \in \{0,1\} \quad \forall i,k$$

c_{ikjh} = k yerinin i yapısının ve h yerinin j yapısının yerleşim maliyeti,

x_{ik} = Eğer i yapısı k yerine yerleşirse 1, diğer durumlarda 0' dır.

Burkard [40] TB kullanarak KAP' ta bir çözüm elde etmiştir. Burkard, sonlu bir N ={1, 2, ..., n} kümesi ile iki tane (n*n) boyutlu, A = (a_{ij}) (uzaklıklar matrisi) ve B = (b_{kl}) (bağlantı matrisi) matrislerini tanımlayarak problemi formüle etmiştir. Burada amaç N kümesinin bir ϕ permutasyonunu bulmaktır. N kümesinin toplamı,

$$\sum_{i=N} \sum_{j=N} a_{ij} b_{\phi(i)\phi(j)}$$ minimum olur.

KAP' ın mümkün çözümlerinin kümesi, N kümesinin bütün permütasyonlarının kümesidir. Komşu çözümler, permütasyondaki iki elementin değişimi ile oluşturulur. Tavlama, tam bir devirde birçok değişimin rassal bir değişebilir ölçümü ile kontrol edilir.

Tam devir yapılırken değişimler yoksa işlem durur. Birçok tekrar da bazı k tamsayısı için $O(n^k)$ operasyonlar kümesi olabilir. Bu tekrarlardan çözümü kaliteli ve etkili olan seçilir ve böylece akla uygun yüksek değerli bir küme olabilir. Soğuma oranı 0.5 oldu. Burkard sonuçta sunulan işlemin hesaplanabilir etkinlikte olduğunu görmüştür. Bu yaklaşım, yüksek kaliteli çözümler olarak ve çok kısa bir zamanda iyi alt optimal çözümler olarak bulabilir. Bundan başka, bu yaklaşım KAP için diğer hızlı ve sofistik sezgisel algoritmalarla iyi bir şekilde karşılaştırıldı.

Burkard [41] KAP'ın bu problemin sınıflamasını ve matematiksel formülasyonunu birkaç uygulamaya tanımlayarak KAP için bir inceleme sunmuştur. Tavlama Benzetimi algoritması, Bender'in algoritmasına dayanan, karşı ve bağımsız sezgisel algoritmasının artan derecelerinin bir metodu ile karşılaştırıldı. Her ikisinin sonuçlarında görüldü ki, TB, KAP' ı çözmek için çok güçlü bir araçtır.

Bonomi [42] de KAP' ı çözmek için TB' ni kullanmıştır. Bonomi azalan sıcaklığın optimal çözümlerin bir komşu çözümde daralan rassal arama işlemine karşılık

geldiğine dikkat çekmiştir. İlk sıcaklık başladığında, istatistiksel dengeye gelmesine müsaade etmiştir. İşleme devam edildiğinde azalan dereceler şöyleydi :

$T_{n+1} = \delta T_n$, $\delta = 0.925$, $n = 1, 2, \ldots, 60$.

Wilhelm [43] KAP' larda TB performansını test ettiler. Son zamanlarda bazı büyük yapıdaki problemlerin (n = 50 ve n = 100) oluşturulması derecesinde iyi olan bir standart problemler kümesi kullanmıştır. Sonuçları diğer geleneksel yapıdaki sezgisel algoritmalarla karşılaştırmıştır. Tavlama çizelge sıcaklıklarının kümesi t_i = (10) $(0.9)^{i-1}$ ile tanımlanır. i = 1, 2, ..., r (t_1 başlangıç sıcaklığı t_r son sıcaklıktır). Yeni çözümler, iki yapının rassal seçilmesi ile ve onların yerlerinin değişimi ile oluşturulur. Ortak bir problem için TB çözümlerinin kalitesi, KAP' a geleneksel yaklaşımlarla elde edilen çözüm kalitesinden daha iyiydi. Bu sonuçları elde etmek için, parametrik değerlerin farklı kombinasyonlar altındaki çalışmaları icra edildi. Böylece TB'nin birçok kontrol parametresine ve durdurma kurallarına daha duyarlı oldukları görüldü.

3.3.5. Parti hacmi problemleri

Çok seviyeli parti hacmi problemi, çok aşamalı üretimde, üretim miktarlarının tanımlanması problemidir. Böylece hazırlık maliyetlerinin ve tutma maliyetlerinin toplamı minimum edilir. Problem, bir karışık tamsayılı doğrusal programlama problemi olarak gösterilebilir.

X_{it} = t zamanında i parçasının üretim miktarı
I_{it} = t zamanındaki i parçasının envanter miktarı
S_i = i parçası ile birleştirilen hazırlık maliyeti kümesi

$$v_{it} = \begin{cases} 0, & \text{Eğer t zamanında i parçası hazır değilse} \\ 1, & \text{Diğer durumlarda} \end{cases}$$

h_i = i parçasının tutma maliyeti

a_{ij} = Bir önceki zaman periyodunda i parçasının üretilmesi ile başlatılan j parçası için oluşturulan talep

Amaç fonksiyon

$$\min Z = \sum_{i=1}^{N} \sum_{t=1}^{r} \left(S_i Y_{it} + p_{ij} x_{it} + h_i I_{it} \right)$$

Kısıtlar

$$I_{it} + x_{ij} - \sum a_{ij} x_{jt} - d_{it} = I_{it}$$

$x_{it} - MY_{it} \leq$

$I_0 = 0 \quad \forall i$

$x_{it} \geq 0 \quad \forall i,t$

$I_{it} \geq 0 \quad \forall i,t$

$y_{it} \in \{0,1\} \; \forall i,t$

$M > 0$ (çok büyük bir sayı)

Kuik [43] TB'ni çok seviyeli parti hacmi problemlerini çözmek için uygulamıştır. Komşu çözümler geçiş mekanizmaları ile oluşturuldu. Bu geçiş mekanizmaları, problemin değişen parametreleri, parça-periyot hazırlıkları, yapıdaki parça seviye sayısı ve parça-periyot talebidir. Tavlama işleminin sıcaklığı, başlangıç minimum 20 derece ile başlayıp, β / α oranı ile onu takip eden daha yüksek sıcaklıklarla arttırılır. Burada $\alpha \in \{ 0.85, 0.90, 0.985 \}$ ve β şimdiki sıcaklıklardır. İterasyon sayısı olarak 80, 100 ve 120 olarak ele alınmıştır. Kuik küçük problemler için TB sezgisel algoritmasının iyi sonuçlar sağladığını görmüştür.

Kuik [44] sunulan bu algoritmayı çok seviyeli kapasitelenen parti hacmi problemlerine uygulamıştır. Kapasitelenen problemler, kapasitelenemeyen bir benzer amaç fonksiyon gibi yarar sağlar. Bununla beraber üst sınırlar değişik yapıların üretim kapasitelerinde yerleştirilmiştir. Yapılan deneysel sonuçlar gösterdi ki, TB karşılaştırılan doğrusal programlamaya dayanan sezgisel algoritmadan iyi bir

performans sağlamakta ve ikincinin etkinliği TB'deki elementlerin birleştirilmesi ile gelişebilir.

3.4. Tavlama Benzetimi Algoritmasının Avantajları

Tavlama Benzetimi algoritmasının başlıca avantajları şunlardır :

1. TB optimizasyon problemleri için genel bir metodtur.
2. TB' nin performansı problemin büyüklüğünden bağımsızdır.
3. Problemlere göre optimal soğuma çizelgesi oluşturulur.
4. Eğer bir yakın optimal çözüm yeterli ise çoklu zaman soğuma çizelgesi oluşturulur.
5. Maliyet fonksiyonunun global minimum değeri daha az sayıda iterasyonla belirlenebilir.

3.5. Tavlama Benzetimi Algoritmasının Çeşitleri

3.5.1. Hızlı Tavlama Benzetimi (HTB)

HTB konveks fonksiyonların minimizasyonunda kullanılır. TB'de olduğu gibi, fonksiyon düz bir fonksiyon değildir. TB ile minimize edilen fonksiyonlar da aday noktalar bir Gauss dağılımı ile oluşturulur. Geman [45] yeterli ve gerekli global minimuma ulaşmak için sıcaklığı şu formüle göre güncelleştirmiştir .

$$T_{(k)} = \frac{1}{\ln(1+k)}$$

Burada k iterasyon sayısıdır. Burada, Szu [46] tarafından Cauchy dağılımı kullanılmıştır :

$$H_{(x)} = \frac{T_{(k)}}{T_{(k)}^{\,2} + x^2}$$

Bu algoritma aynı zamanda klasik TB algoritmasından daha hızlı bir şekilde global minimumu bulduğunu belirtmişlerdir.

3.5.2. Genelleştirilmiş Tavlama Benzetimi

Genelleştirilmiş TB algoritması kombinatoriyal optimizasyon problemlerini daha kolay çözmektedir. Bu algoritmada, Bohachevsky [47] klasik TB da kullanılan kabul edilebilirlik olasılığı olan a_1 yerine, yeni bir kabul edilebilirlik olasılığı olan a_2 kullanmıştır. Burada ;

$a_1 = \min (1, e^{-\Delta E / T})$

$a_2 = \min (1, \exp^{(-\Delta E / E^*T)})$' dir.

Burada maliyet fonksiyonu $E_{opt} = 0$ olarak kullanılmıştır. Böylece global minimumun dışına çıkma olasılığı sıfıra çok yakındır. Yerel minimumun dışına çıkma olasılığı çok azdır. Çünkü $E_{yerel\ minimum} > 0$' dır.

Genelleştirilmiş SA algoritmasının ilginç bir özelliği de sadece tek bir sıcaklıkla yapılmasıdır. Bu eşitlik aşağıda verilmiştir.

$0.5 < \exp (-\Delta E / E^*T) < 0.9$

Burada üssel değer T sıcaklığına bağlı olarak belirlenir.

BÖLÜM 4. TAVLAMA BENZETİMİ YAKLAŞIMI İLE İŞ SIRALAMA

Tavlama Benzetimi, alternatif çözümlerin çok fazla olduğu ve bu alternatiflerin birer birer denenerek en iyi çözümün bulunmasının oldukça fazla zaman aldığı problemlerde, klasik çözüm algoritmalarına göre daha iyi sonuçlar verebilmektedir. Parça tasarımı, iş çizelgeleme, gezgin satıcı vb. problemler, bunlara örnek olarak verilebilir. Bu problemlerde çözüme etki eden parametreler arttıkça problemin çözümü de o derece güçleşmekte ve klasik yöntemlerle çözülmesi imkansızlaşmaktadır. İş sıralaması problemi de bu problemlerden biridir. Bu bağlamda Tavlama Benzetimi iş sıralamasında rahatlıkla uygulanabilen bir yaklaşımdır. Aşağıda bu problemin çözümü için bir yaklaşım sunulmuştur.

4.1. Problemin Tanımlanması

Burada Tavlama Benzetimi algoritmasının çizelgeleme problemlerine adapte edilmesi anlatılacaktır. Bu algoritmada kullanılan notasyonların ve terminolojilerin tanımı aşağıda verilmiştir.

n : Çizelgelenecek iş sayısı

m : Makina sayısı

t_{ij} **:** j makinasındaki i işinin işlem zamanı

F : Sıralanan işlerin toplam akış zamanları

T : Tavlama Benzetimi prosesinin sıcaklığı

r : Sıcaklık düşürme faktörü

L : Herhangi bir sıcaklıktaki iterasyon sayısı

DONMA SAYACI : Çözümün donup donmadığını kontrol etmek için kullanılır. Sayaç 5'e ulaştığında veya sıcaklık 8' e düştüğünde çözümün donduğu varsayılır ve sayaç tekrar sıfırlanarak, aşağıda tanımlanacak olan $F(S') \leq F(G)$ ilişkisine bakılır.

KABUL : Belirli bir sıcaklıkta kabul edilen sıralamaların zamanlarının sayısını tutmak için kullanılan sayaçtır.

TOPLAM : Belirli bir sıcaklıkta toplam sıralamaların sayısını tutmak için kullanılan sayaçtır.

ORAN : Herhangi bir sıcaklıkta kabul edilen sıralamaların yüzde oranlarını depolar ve daha iyi sıralama bulmada başarılıdır. Eğer herhangi bir sıcaklıkta ORAN < %15 ise DONMA SAYACI 1 arttırılır. Dolayısıyla sıralama donma durumuna biraz daha yaklaşır.

{G} : O ana kadar karşılaşılan en iyi sıralamayı depo ederek sıralar {G} her zaman güncelleştirilir, DONMA SAYACI yeniden sıfırlanır, böylece daha iyi çözümler elde edilebilir.

{S} : Kullandığımız sıralama değiştirme yöntemindeki kabul edilen sıralamayı depo ederek sıralar ve bunu sırayla yapar.

{S' } : {S}'den elde edilen kullandığımız sıralama değiştirme yöntemindeki sıralamayı tutar.

$F_G, F_S, F_{S'}$: Sırasıyla {G}, {S}, {S' }' nün toplam akış zamanlarıdır.

PERTURB ({S},{S' },$F_{S'}$) : {S} sıralaması yeni bir sıralama elde etmek için bir yöntem ile değiştirilir. Bunlar sırasıyla { S' }' de depo edilir ve bunun da toplam akış zamanı $F_{S'}$ ' dür. Biz burada bu yöntemlerden 'rassal olarak birbirleriyle değişmesi' ve 'komşu işlerin birbirleriyle değişmesi' yöntemlerini kullandık.

DELTA ({G}, { S' }) : {S' } ile {G} birbirleriyle karşılaştırılır. Buna göre;

$$\Delta G = \frac{F_{S'} - F_G}{F_G}$$

DELTA ({S}, { S'}) : {S'} ile {S} birbirleriyle karşılaştırılır. Buna göre;

$$\Delta S = \frac{F_{S'} - F_S}{F_S}$$

P : Kabul edilme olasılığı istatistiksel kavramlarla ortaya çıkarılır ve daha iyi bir çözüm olup olmayacağı belirlenerek, çözüm kabul veya reddedilebilir.

U : 0-1 arasında düzgün dağılan rassal bir sayıdır. Bir çözümün kabulünü kontrol etmek için kullanılır.

U' : 0-1 arasında düzgün dağılan rassal bir sayıdır. ' komşu işlerin birbirleriyle değişmesi ' yönteminde komşu pozisyonunu seçmek için kullanılır.

4.2. Sıralama Değiştirme Yöntemleri

Bu çalışmada iki tane sıralama değiştirme yöntemi kullanılmıştır:

1. Birbirini takip eden (komşu) işlerin birbirleriyle değişmesi yöntemi
2. Rassal olarak işlerin birbiriyle değişmesi yöntemi

4.2.1. Birbirini takip eden (komşu) işlerin birbiriyle değişmesi yöntemi

Bu yöntemde işler değişirken birbirine komşu olan işle yer değiştirmektedir. Bunu bir örnekle şöyle açıklayabiliriz :

Elimizde beş iş ve bunların sıralaması da { 4-1-3-2-5 } olsun. Bu sıralamamız S sıralaması olsun. Bu yönteme göre, 1 ile iş sayısı (n) arasında (burada n = 5) bir iş pozisyonu seçilir. Örnek olarak burada 2 numaralı iş pozisyonunu seçelim. Bu pozisyona karşılık gelen iş, iş 1' dir. Bu yönteme göre, bu iki numaralı iş pozisyonu ya 1, ya da 3 numaralı iş pozisyonu ile yer değiştirecektir; yani ya iş 4 ile veya iş 3 ile yer değiştirecektir. Bunu seçmek için de 0-1 arasında düzgün dağılan bir sayı seçilir. Bu seçimin nasıl olduğu aşağıda algoritma açıklanırken daha açık bir şekilde verilecektir. Farz edelim ki burada pozisyon 3 seçilsin, yani iş 3 seçilsin. Bu durumda iş 1 ile iş 3 yer değiştirmiş olup, yeni sıralamamız {4-3-1-2-5} olur. Bu sıralama da S' sıralamamız olur. Eğer seçtiğimiz iş pozisyonları 1 veya 5 olsaydı, 1 için sadece sağındaki pozisyonun, 5 için sadece solundaki pozisyonun seçileceği aşikardır. Bir örnek verirsek; yine başlangıç sıralamasında 1 numaralı iş pozisyonunu yani iş 4' ü seçelim. Buna göre, iş 4 sadece iş 1 ile değişebilir. Böylece yeni sıralama {1-4-3-2-5} olacaktır.

4.2.2. Rassal olarak işlerin birbiriyle değişmesi yöntemi

Bu yönteme göre, iş pozisyonları tamamen rassal seçilir ve bu seçilen işler birbirleriyle yer değiştirir. Yine yukarıdaki örnekteki başlangıç sıralamasını ele alalım. Buna göre S sıralamamız { 4-1-3-2-5} olur. Eğer 1 ve 4 numaralı pozisyonlar yer değiştirirse, iş 4 ve iş 2 yer değiştirmiş olacaktır. Böylece S' sıralamamız da {2-1-3-4-5} olacaktır. Yani iş seçimi tamamen rassaldır.

Bu tezde bu yöntemlerin her ikisi de kullanılarak birbirleriyle bir karşılaştırma da yapılmıştır.

4.3. Tavlama Benzetimi Yaklaşımı ile İş Sıralama Algoritması

Bu yaklaşımda ilk önce her hangi bir klasik yolla bir başlangıç sıralaması oluşturulur. Daha sonra oluşturulan bu sıralamayı burada sunduğumuz Tavlama Benzetimi

yaklaşımı ile geliştirmeye başlarız. Bu yaklaşımın adım adım algoritması aşağıda verilmiştir. Bu algoritmanın bilgisayar kodlaması C dilinde ve Ek A'da verilmiştir.

ADIM 1. Başlangıç {G} ve {S} sıralamaları ile bunlara ait toplam akış zamanları elde edilir.

ADIM 2. T = 200; KABUL = 0; TOPLAM =0; DONMA SAYACI =0 olarak başlangıç değerleri verilir.

ADIM 3. Eğer ((DONMA SAYACI = 5) veya (T ≤ 8)),

ise ; ADIM 14' e git.

Değilse ; ADIM 4' e geç

ADIM 4. Bu adımda 'komşu işlerin birbirleriyle değişmesi ' yöntemi kullanılarak, {S' } ve $F_{S'}$ bulunur. İlk önce 0-1 arasında düzgün dağılan bir U_1' sayısı seçilir. Bundan sonra k iş pozisyonunun seçimine geçilir.

Burada k = int (n* U_i') olarak seçilir. Yine 0-1 arasında düzgün dağılan bir U_{i+1}' sayısı seçilir. Eğer $U_{i+1}' \leq 0.5$

ise; k.cı iş pozisyonu (k-1) ci ile yer değiştirir ve S' sıralaması oluşturulur.

Değilse; k.cı iş pozisyonu (k+1) ci ile yer değiştirir ve S' sıralaması oluşturulur.
(Eğer k = 1 ise U_{i+1}' seçilmez. 1 pozisyonundaki iş, 2 pozisyonundaki iş ile yer değiştirir. Aynı şekilde k= n ise bu sefer (n-1) pozisyonundaki iş ile yer değiştirir).
ΔS hesaplanır = DELTA ({S}, {S'}). Eğer (ΔS ≤ 0);

ise; ADIM 5' e geç

değilse; ADIM 8' e git.

ADIM 5. Atama işlemleri yapılır

{S} yerine {S'}; F_S yerine $F_{S'}$ atanır. KABUL = KABUL + 1

ADIM 6. ΔG hesaplanır = DELTA ({G}, {S'}). Eğer ($\Delta G \leq 0$);

ise; ADIM 7' ye geç.

Değilse ; ADIM 10' a git.

ADIM 7. Atama işlemleri yapılır.

{G} yerine {S'}; F_G yerine $F_{S'}$ atanır. DONMA SAYACI = 0; ADIM 10' a git.

ADIM 8. P = exp ($-\Delta S / T$); rassal bir U sayısı seçilir. Eğer (U > P)

ise ; ADIM 10' a git.

Değilse ; ADIM 9' a geç.

ADIM 9. Atama işlemleri yapılır.

{S} yerine {S'}; F_S yerine $F_{S'}$ atanır. KABUL = KABUL + 1

ADIM 10. TOPLAM = TOPLAM + 1

ADIM 11. Eğer (TOPLAM > 4*n) veya (KABUL > n));

ise ; ADIM 12' ye geç.

Değilse ; ADIM 4' e git.

ADIM 12. ORAN = (KABUL / TOPLAM)*100

Eğer ORAN ≤ 15 ;

ise ; DONMA SAYACI = DONMA SAYACI + 1

ADIM 13' e geç.

Değilse ; ADIM 13' e geç.

ADIM 13.

T = 0.9 *T;

KABUL = 0

TOPLAM = 0

ADIM 3' e git.

ADIM 14.

Çözüm donmuştur.

DUR.

BÖLÜM 5. TARTIŞMA ve SONUÇLAR

Bu çalışmada, Tavlama Benzetimi algoritmasının permütasyon tipi iş sıralama problemlerinin çözümünde etkin bir şekilde kullanıldığı gösterilmeye çalışılmıştır. Bunu yapmak için de aşağıda adları geçen iki metodtan yaralanılmıştır:

1. Birbirini takip eden (komşu) işlerin birbiri ile değişmesi yöntemi (KİBDY)

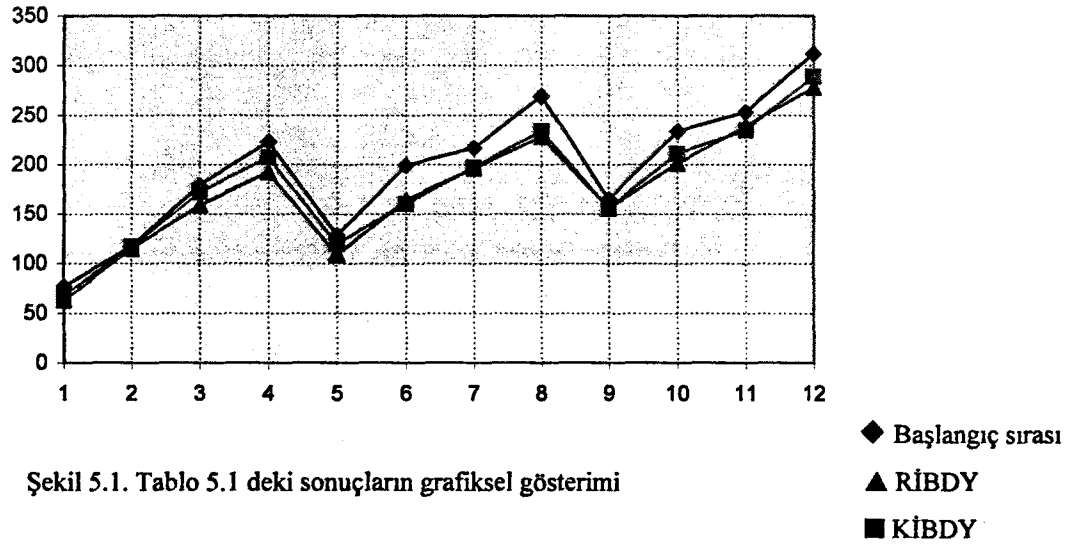
2. Rassal olarak işlerin birbiri ile değişmesi yöntemi (RİBDY)

Bu iki metod ile bir çok deneyler yapılarak toplam akış minimize edilmiştir. Bu problemler, çeşitli sayıdaki iş ve makina problemleridir. Bu sonuçlar Tablo-5.1' de gösterilmiştir.

Bu çalışmada ilk önce, bir başlangıç sıralaması seçiliyor. Burada bu sıralama rassal olarak seçilmiştir. Tavlama Benzetimi algoritması bundan sonra devreye girerek bu başlangıç sıralamasını geliştirmeye başlıyor. Bütün problemlerde ele alınan başlangıç sıralamalarının toplam akış zamanları, yukarıdaki iki yönteme Tavlama Benzetimi algoritması uygulanarak daha minimize hale getirilmiştir. Bu uygulamada sıcaklık değerleri arttırılarak çeşitli sonuçlar elde edilmiştir.

Tablo 5.1. T=200 derece için, C_{max} performans kriterine göre problemlerin çözümü

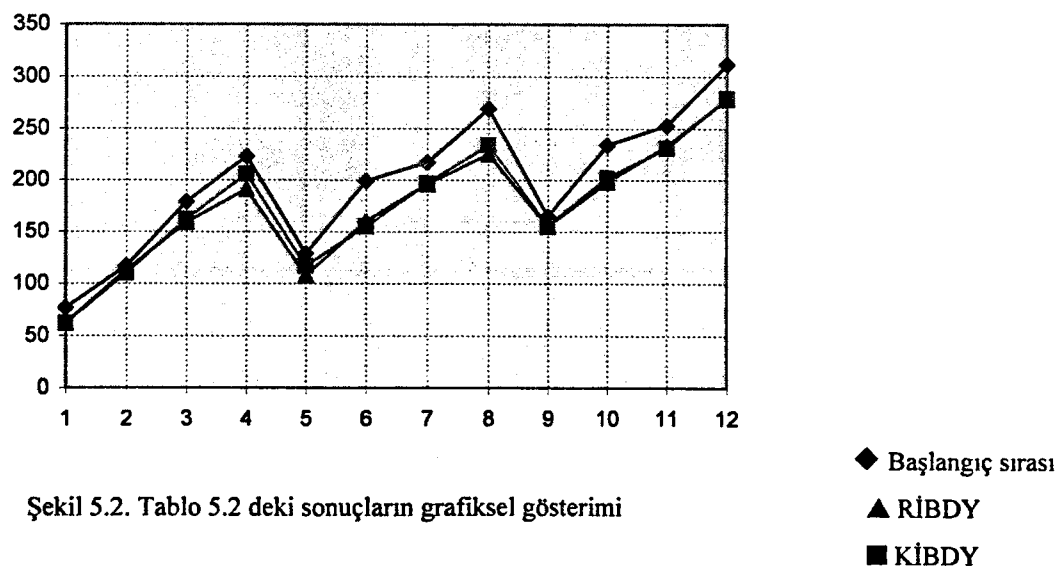
Problem Büyüklüğü		Başlangıç Sıralaması	Tavlama Benzetimi Algoritması	
n	m		KİBDY	RİBDY
5	5	77	68	63
5	10	117	117	115
5	15	179	172	159
5	20	223	207	192
10	5	129	120	109
10	10	199	160	164
10	15	217	197	196
10	20	269	234	228
15	5	165	156	156
15	10	234	211	201
15	15	253	235	239
15	20	312	289	278



Şekil 5.1. Tablo 5.1 deki sonuçların grafiksel gösterimi

Tablo 5.2. T=500 derece için, C_{max} performans kriterine göre problemlerin çözümü

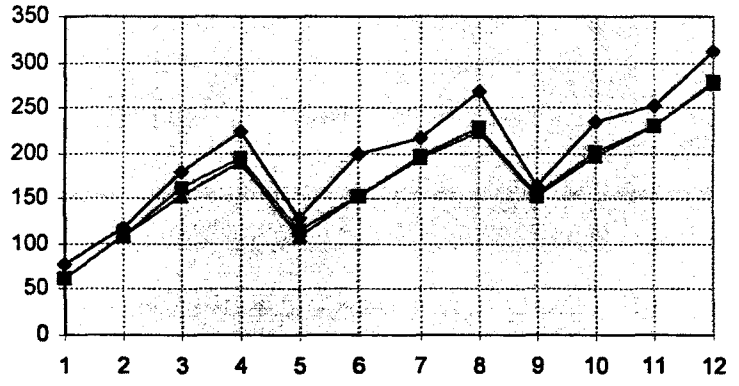
Problem Büyüklüğü		Başlangıç Sıralaması	Tavlama Benzetimi Algoritması	
n	m		KİBDY	RİBDY
5	5	77	62	62
5	10	117	110	113
5	15	179	162	159
5	20	223	206	191
10	5	129	117	108
10	10	199	155	160
10	15	217	197	196
10	20	269	234	225
15	5	165	156	155
15	10	234	202	198
15	15	253	231	233
15	20	312	279	278



Şekil 5.2. Tablo 5.2 deki sonuçların grafiksel gösterimi

Tablo 5.3. T=1000 derece için, C_{max} performans kriterine göre problemlerin çözümü

Problem Büyüklüğü		Başlangıç Sıralaması	Tavlama Benzetimi Algoritması	
n	m		KİBDY	RİBDY
5	5	77	61	61
5	10	117	109	109
5	15	179	162	153
5	20	223	195	191
10	5	129	115	108
10	10	199	153	153
10	15	217	197	196
10	20	269	228	223
15	5	165	154	152
15	10	234	202	197
15	15	253	230	230
15	20	312	279	276



Şekil 5.3. Tablo 5.3 deki sonuçların grafiksel gösterimi

Tablo5.1, 5.2 ve 5.3'de de görüldüğü gibi, başlangıç sıralamalarının toplam akış zamanları Tavlama Benzetimi ile geliştirildiğinde daha da minimize olmuş durumdadırlar. Ayrıca sıcaklık her defasında arttırıldığında toplam akış zamanları daha da düşmektedir. Her sıcaklık artımında yerel aramalar artacağı için optimal sonuca yaklaşımda daha fazla olacaktır. Tavlama Benzetimi birçok alternatifi değerlendirdiği için - yani yerel aramaları çok yaptığı için- bu sonuçlar optimal değerlere çok yakın olmaktadır.

Tavlama Benzetimi algoritmasının daha iyi sonuçlar verebileceğini anlamak için iş ve makine sayıları devamlı arttırılmıştır. Görüldüğü gibi iş ve makine sayısının artması ile TB daha iyi sonuçlar vermiştir. Bu da göstermektedir ki, iş ve makine sayısının artması diğer klasik ve sezgisel yöntemlerin verimliliğini azaltırken, TB' da böyle bir şey söz konusu olmamaktadır.

TB'nin diğer yöntemlere göre dezavantajı, çözümlere rassal olarak başlaması ve bunun sonucunda da , problemin çözümü için daha fazla iterasyona gerek duymasıdır.

KAYNAKLAR

[1] FRENCH, S., Sequencing and Scheduling, John Wıley and sons, New York,1982.

[2] KIRKPATRICK, S., GELATT, CD. and VECCHI, PM., " Optimization by Simulated Annealing", Science , Vol 220, pp. 671-680, 1983.

[3] CERNY, V., A " Thermodynamically Approach to The Travelling Salesman Problem: An efficient Simulated Annealing Algorithm", Journal of Optimization Theory and Applications,Vol 45, pp. 41-45, 1985.

[4] METROPOLIS, W., ROENBLUTH, A., TELLER, A. and TELLER, E., "Equation of the State Calculations by Fast Computing Machines", Journal of Chemical Physics, Vol 21, pp. 1087-1092, 1953.

[5] JOHNSON, S., ARAGON, C., MCCGEOCH, L. and SCHEVEON, C. "Optimization by Simulated Annealing: An Experimental Evaluation", Part 1, Operations Research, Vol 37, pp. 865-892, 1989

[6] OSMAN, IH. and CHRISTOFIDES N., " Simulated Annealing and Descent for Algorithms Capacitated Clustering Problems " , Presented at EURO X, Beograd, Yugoslavia, 1989.

[7] LAWLER, EL. " Combinatorial Optimization: Networks and Matroids ", Holt, Rinehart and Winston, Newyork, 1976.

[8] PAPADIMITRIOU, CH. and STEGILITZ, K., Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall, Newyork, 1982.

[9] NEMHAUSER, GL. and WOLSEY LA., Integer and Combinatorial Optimization, John Wiley, Chichester, 1988.

[10] OSMAN, IH. and POTTS, CN., " Simulated Annealing for Permutation Flow-Shop Scheduling ", Omega, Vol 17, pp. 551-557, 1989.

[11] OSMAN, IH. and CHRISTOFIDES N., " Simulated Annealing and Tabu Search Techniques for the Generalised Assignment Problem ", Presented at the 12th Triennial Conference on Operations Research of the International Federation of Operations Reserach Societies, IFORS' 90, Athens, Greece, 1990.

[12] OSMAN IH. , " Metastrateji Simulated Annealing and Tabu Search Algorithms for Combinatorial Optimization Problems ", Ph.D Thesis, Management School, Imperial College, London, UK, 1991.

[13] VAN LAARHOVEN, PJ. and AARTS, EHL., " Simulated Annealing: Theory and Applications ", Reidel, Dordrecht, 1987.

[14] COLLINS, NE., EGLESE, RW and GOLDEA, BL, "Simulated Annealing- an Annotated Bibliography ", American Journal Mathmetical Management Science, Vol 8(3), pp. 209-307, 1988.

[15] AARTS, E. and KORTS, J., " Simulated Annealing and Boltzman Machines ", Wiley & Sons, Aarts Chichester, 1988.

[16] JOHNSON, S., ARAGON, C., MCCGEOCH, L.and SCHEVEON, C., " Optimization by Simulated Annealing: An Experimental Evaluation ", Part 1, Operations Research, vol 37, pp. 865-892, 1989.

[17] EGLESE, RW., " Simulated Annealing: A tool for Operational Research ", Europan Journal Operations Research, vol 46, pp. 271-281, 1990.

[18] POTTS, CN and VAN WASSENHOVE, LN., " Single Machine Tardiness Sequencing Heuristics ", IIE Transactinos, vol 23, pp. 346-354, 1991.

[19] LUNDY, M. AND MEES, A., " Convergence of an Annealing Algorithm ", Mathmatical Programming, vol 34, pp. 111-124, 1986.

[20] MITTENDHAL, J., RAGHAVACHARI, M. And RANA A., " A Hybrid Simulated Annealing Approach for Single Machine Scheduling Problems With Non-Regular Penalty Functions ", Computers Operation Research, Vol 20, Pp. 103-111, 1993.

[21] OGBU, FA. and SMITH, DK., "The Application of the Simulated Annealing Algorithm to the Solution of the n/m/c Subscript Max Flowshop Scheduling ", Computers Operations Research, Vol 17, pp. 243-253, 1990.

[22] PALMER, DS., " Sequencing Jobs Through a Multistage Process in the Minimum Total Time- A Quick method of Obtaining a Near Optimum ", Operations Research Q., vol. 16, pp. 101-107, 1965.

[23] DANNENBRING DG., " An Evaluation of Flow-Shop Sequencing Heuristics, Management Science ", vol. 23, pp. 1174-1183, 1977.

[24] VAN LAARHOVEN PJM., AARTS, EHL. and LENSTRA, JK., " Job Shop Scheduling by Simulated Annealing ", Operations Research, vol. 40, pp. 113-125, 1992.

[25] VAKHARIA, AJ. and CHANG, Y., " A Simulated Annealing Approach to Scheduling a Manufacturing Cell ", Naval Research Logistics, vol. 37, pp. 389-407, 1990.

[26] SRIDHAR, J. and RAJENDRAN C., "Scheduling in a Cellular Manufacturing system : A Simulated Annealing approach ", International Journal Production Research, vol. 31, no. 12, pp. 2927-2945, 1993.

[27] GOLDEN, B. and SKISCIM, C., " Using Simulated Annealing to Solve Routing and Location Problems ", Naval Research Q., vol. 33, pp. 261-279, 1986.

[28] RANDELMAN, L. and GREST, G., " N-city Traveling Salesman Problem: Optimization by Simulated Annealing ", Journal Statistical Physics, vol. 45, pp. 885-890, 1986.

[29] AARTS, E., KORST, J. and VAN LAARHOVEN, P., " A Quantitative Analysis of the Simulated Annealing Algorithm: A Case Study for the Travelling Salesman Problem ", Journal Statist. Physics, vol. 50, pp. 187-206, 1988.

[30] ALFA, A., HERAGU, S. and CHEN, M., " A 3- opt Based Simulated Annealing Algorithm for Vehicle Routing Problems ", Computers Industrial Engineering, vol. 21, pp. 635-639, 1991.

[31] JAJODIA, SI., MINIS, I., HARHALAKIS, G. and PROTH, J., " CLASS: Computerized Layout Solutions Using Simulated Annealing ", International Journal Production Research, Vol. 30, pp. 95-108, 1992.

[32] KUSIAK, A. and HERAGU, SS., " The Facility Layout Problem ", Europan Journal Operations Research, Vol. 29, pp. 229-251, 1987.

[33] HERAGU, SS. and ALFA, AS., " Experimental Analysis of the Simulated Annealing Based Algorithms for the Layout Problem ", Europan Journal Operations Rsearch, vol. 57, pp. 190-202, 1992.

[34] HERAGU, SS. and KUSIAK, A., " Efficient Models for the Facility Layout Problem ", Europan Journal Operations Research, vol. 53, pp. 1-13, 1991.

[35] KOUVELIS, P. and CHIANG, W., " A Simulated Annealing Procedure for Single Row Layout Problems in the Flexible Manufacturing Systems ", International Journal Production Research, vol. 30, pp. 717-732, 1992.

[36] KOUVELIS, P., CHIANG, W. and FITZSIMMONS, J., " Simulated Annealing for Machine Layout Problems in the Presence of Zoning Constraints ", Europan Journal Operations Research, vol. 57, pp. 203-223.,1992.

[37] SHARPE, R and MARKSJO, BS., " Facility Layout Optimization Using the Metropolis Algorithm ", Envir. Plann. B, vol. 12, pp. 443-453, 1985.

[38] SCRIABIN, M. and VERGIN, RC., " Comparison of Computer Algorithms and Visual Based Methods for Plant Layout ", Management Science, vol. 22, pp. 172-181, 1975.

[39] NUGENT, CE., VOLLMAN, TE. and RUML, J., " An Experimental Comparison of Techniques for the Assingment of Facilities to Locations ", Operations Research, vol. 16, pp. 150-173, 1968.

[40] BURKARD, R. and RENDL, F., " A Thermodynamically motivated Simulation Procedure for Combinatorial Optimization Problems ", Europan Journal Operational Research, vol. 17, pp. 169-174, 1984.

[41] BURKARD, R., " Quadratic Assignment Problems ", Europan Journal Operational Research, vol. 15, pp. 283-289, 1984.

[42] BONOMI, E. and LUTTON, J., " The Asymptotic Behavior of Quadratic Sum Assignment Problems: A statistical Mechanics Approach ", Europan Journal Operational Research, vol. 26, pp. 295-300, 1986.

[43] KUIK, R. and SALOMON, M., " Multi-level Lot Sizing Problem: Evaluation of a Simulated Annealing Heuristic ", Europan Journal Operations Research, vol. 45,pp. 25-37, 1990.

[44] KUIK, R., SALOMON, M., VAN WASSENHOVE, LN. and MAES J., " Linear Programming, Simulated Annealing and Tabu Search Heuristics for Lot Sizing in Bottleneck Assembly Systems ", IIE Transactions, vol. 25, pp. 62-72, 1993.

[45] GEMAN, S. And GEMAN, D., " Stochastic Relaxation, Gibbs Distribitions, and the Bayesian Restoration of Images ", IEEE Transactions of Pattern Analysis and Machine Intelligence, vol. PAMI-6, no. 6, pp. 721-735, 1984.

[46] SZU, H., " Fast Simulated Annealing ", in Neural Network for Computing, J. S. Denker (ed.), American Institute of physics, New York, 1986.

[47] BOHACHEVSKY, IO., JOHNSON, ME. and STEIN, ML., " Generalized Simulated Annealing for Function Optimization ", Technometrics, vol. 28, no. 3, pp. 209-217, 1986.

EKLER

Ek A: Birbirini Takip Eden (Komşu) İşlerin Birbiri ile Değişmesi Yöntemi ile İş Sıralamanın C Programlama Dilinde Kodlanması

```
/*BİRBİRİNİ TAKİP EDEN KOMŞU İŞLERİN BİRBİRİYLE DEĞİŞMESİ
YÖNTEMİ*/

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>
FILE *sonuc;
#ifdef __cplusplus

 int max (int value1, int value2);

 int max(int value1, int value2)
 {
   return ( (value1 > value2) ? value1 : value2);
 }

#endif



int main(void)
{
```

```
clrscr();
    int t      = 200;
    float accept  = 0;
    float total   = 0;
    float frcount = 0;
    float enb, enb1,enb2;
    int isler[100];
    int c[100][100];
    int opt[100][100] ;
    sonuc=fopen("out.dat","w");
    fprintf ( sonuc, "    Sıcaklık          Toplam Akış Zamanı");
    register int i,j,s;
    int n; printf("\ iş sayısı..", n);scanf("%d", &n);
    int m;
    printf ("\ makina sayısı = ", m);scanf("%d",&m);
    int k=0,h=0,a=0;
    printf ("Başlangıç sıralaması :");

    for (  i = 0; i < n ; i++ )
        {
           printf ("\n %d. iş=", i);
           fflush(stdin);
           printf ("", isler[i]);
           scanf("%d",&isler[i]);
    devam:  if ( isler[i] >= n   )
                {
                 printf (" lütfen iş sayısından büyük sayı girme");
                 printf (" lütfen yeniden giriniz\n %d.iş..:",i);
                 scanf("%d",&isler[i]);
                 if (isler[i] >= n ) goto devam;
                }
        }
```

```
for ( i = 0; i < n; i++ )
        {
           for ( int j = 0; j < m ; j++ )
                {
                 printf ("\n %d. iŸin %d.makinada operasyon s□resi=",i,j);
                 scanf("%d",&opt[i][j]);
                }

        }

   devam3: if ( frcount == 5 || t <= 8  ) goto don;
   else
        {
           c[isler[0]][0]=opt[isler[0]][0];
           for (i=1 ; i <n ; i++)
                {
                 for (j=1 ; j <m ; j++)
                   {
                        c[isler[i]][0]=c[isler[i-1]][0] +opt[isler[i]][0];
                        c[isler[0]][j]=c[isler[0]][j-1] +opt[isler[0]][j];
                        int x = c[isler[i-1]][j];
                        int y = c[isler[i]][j-1];
                        int z;
                        z = max(x, y);
                        c[isler[i]][j] = z + opt[isler[i]][j];
                        enb=c[isler[i]][j];
                        enb2=c[isler[i]][j];
                   }
                }
   printf ("\n Başlangıç sıralamasının toplam akış zamanı %f ",enb);
   printf ("\n Başlangıç sıralamasının toplam akış zamanı %f ",enb2);getch();
```

```
top1:isler[i]=0;
clrscr();
randomize();
int  yeni_sayi= rand() % 100;
a= yeni_sayi;printf("\n rs=%d",yeni_sayi);
int k= n * a/100.0 ;
printf (" \n iş no k= %d\n ",k);getch();
int sondeger = (n-1);
if ( k == sondeger )  h = k-1;
else
if ( k == 0 )   h = k+1;
else
    {
        randomize();
        int  yeni_sayi;
        float y,a;
        yeni_sayi= rand() % 100;
        int  b= yeni_sayi;
        y=b/100.0;
        printf ("\n seçilen rassal sayı =  %f\n ",y );

        if ( y <= 0.5 )   h=k-1;
        else
        h=k+1;
    }

printf("\n Değişilecek iş no h=%d\n",h);getch();
a=isler[k];
isler[k]=isler[h];
isler[h]=a;
printf ("\n Yeni Sıralama..:\n" );
for(i = 0 ; i < n ; i ++)
```

```
{
   printf("x[%d]=%d\n",i,isler[i]);getch();
}

   c[isler[0]][0]=opt[isler[0]][0];
   for (i=1 ; i <n ; i++)
       {
        for (j=1 ; j <m ; j++)
          {
              c[isler[i]][0]=c[isler[i-1]][0] +opt[isler[i]][0];
              c[isler[0]][j]=c[isler[0]][j-1] +opt[isler[0]][j];
              int x = c[isler[i-1]][j];
              int y = c[isler[i]][j-1];
              int z;
              z = max(x, y);
              c[isler[i]][j] = z + opt[isler[i]][j];
              enb1=c[isler[i]][j];
          }
       }
printf ("\n yeni sıralamanın toplam akış zamanı %f ",enb1);getch();
float DELTAS=(enb1 - enb)/enb;
printf ("\n Sıralamaların Karşılaştırılmas□=%f", DELTAS);
if ( DELTAS <= 0)
   {
      if ( enb1 <= enb ) enb=enb1;printf ;getch();
      accept = accept + 1;
      printf ("\n kabul toplamı=%f",accept);getch();
      float DELTAG = (enb1 - enb2)/enb2 ;
      printf (" \n delta G =%f",DELTAG);
      if ( DELTAG <= 0)
```

```
            {
             if ( enb2 <= enb ) enb=enb2;getch();
             frcount=0;printf ("\n donma=%f", frcount);
            }


        else  goto top;
    }

else
    {
        randomize();
        float  yeni_sayi;
        float y,a;
        yeni_sayi= rand() % 100;
        a= yeni_sayi;
        float u=a/100.0 ;
        printf (" \n seçilen yeni rassal sayı =  %f ",u );getch();
        float p = exp (-DELTAS / t);
        printf (" \n seçilen p sayısı =  %f ",p );getch();
        if ( u < p)
            {
             if ( enb1 <= enb ) enb=enb1;getch();
             accept = accept +1;
             printf ("\n kabul toplamı=%f",accept); getch();
            }
        else goto top;
    }

top: total = total + 1;
printf ("\n toplam toplamı=%f",total);
if ( total > 4 * n || accept > n )
```

```
        {
            float oran = (accept/total)*100;
            printf ( "\n kabulün toplama oranı= %f", oran);getch();
            if (oran <= 15 )
                {
                 frcount = frcount + 1;printf ("\n donma=%f", frcount);}
            t=0.9*t ;printf ("\n sıcaklık = %d",t);
            accept = 0;
            total = 0;
            goto devam3;
        }
    else
        {
            fprintf (sonuc, "\n    %d       %f            ", t, enb);
            goto top1;
        }

 }
    don: printf (" \n çözüm Donmuştur...");
    fclose(sonuc);
}
```

Ek B: Rassal Olarak İşlerin Birbiri ile Değişmesi Yöntemi ile İş Sıralamanın C Programlama Dilinde Kodlanması

```
/*RASSAL OLARAK İŞLERİN BİRBİRİYLE DEĞİŞMESİ YÖNTEMİ*/

#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
FILE *sonuc;
#ifdef __cplusplus

 int max (int value1, int value2);

 int max(int value1, int value2)
 {
   return ( (value1 > value2) ? value1 : value2);
 }

#endif



int main(void)
{
    clrscr();
    int t      = 200;
    float accept  = 0;     float total   = 0;
float frcount = 0;
    float enb, enb1,enb2;
    int isler[100];
    int c[100][100];
```

```
int opt[100][100] ;
sonuc = fopen ("out1.dat", "w");
fprintf ( sonuc, "\    Sıcaklık              Toplam Akış Zamanı " );
register int i,j,s;
int n; printf("\ iş sayısı..", n);scanf("%d", &n);
int m;
printf ("\ makina sayısı = ", m);scanf("%d",&m);
int k=0,h=0,a=0;

printf ("Başlangıç sıralaması :");

for (  i = 0; i < n ; i++ )
    {
        printf ("\n %d. iş=", i);
        fflush(stdin);
        printf ("", isler[i]);
        scanf("%d",&isler[i]);



        devam: if ( isler[i] >= n   )
        {
              printf (" lütfen iş sayısından büyük sayu girme");
              printf (" lütfen yeniden giriniz\n %d.iş..:",i);
              scanf("%d",&isler[i]);
              if (isler[i] >= n ) goto devam;
        }
        }
    for (  i = 0; i < n; i++ )
    {
        for ( int j = 0; j < m ; j++ )
```

```
        {
            printf ("\n %d. işin %d.makinada operasyon süresi=",i,j);
            scanf("%d",&opt[i][j]);
        }

    }

devam3: if ( frcount == 5 || t <= 8  ) goto don;

else
    {
        c[isler[0]][0]=opt[isler[0]][0];

        for (i=1 ; i <n ; i++)
        {
            for (j=1 ; j <m ; j++)
            {
            c[isler[i]][0]=c[isler[i-1]][0] +opt[isler[i]][0];
            c[isler[0]][j]=c[isler[0]][j-1] +opt[isler[0]][j];
            int x = c[isler[i-1]][j];
            int y = c[isler[i]][j-1];
            int z;
            z = max(x, y);
            c[isler[i]][j] = z + opt[isler[i]][j];
            enb=c[isler[i]][j];
            enb2=c[isler[i]][j];
        }
            }

printf ("\n Başlangıç sıralamasının toplam akış zamanı %f ",enb);
printf ("\n Başlangıç sıralamasının toplam akış zamanı %f ",enb2);getch();
```

```
top1:isler[i]=0;
randomize();
int  yeni_sayi= rand() % 100;
a= yeni_sayi;
int k= n * a/100.0 ;
printf (" \n iş no k= %d\n ",k);getch();
yeni_sayi= rand() % 100;
a= yeni_sayi;
int h=(n-1) * a/100.0 ;
printf("\n Değişilecek iş no h=%d\n",h);
a=isler[k];
isler[k]=isler[h];
isler[h]=a;
printf ("\n Yeni Sıralama..:\n" );

for(i = 0 ; i < n ; i ++)
    {
       printf("x[%d]=%d\n",i,isler[i]);getch();
       c[isler[0]][0]=opt[isler[0]][0];
    }

for (i=1 ; i <n ; i++)
    {
       for (j=1 ; j <m ; j++)
            {
             c[isler[i]][0]=c[isler[i-1]][0] +opt[isler[i]][0];
             c[isler[0]][j]=c[isler[0]][j-1] +opt[isler[0]][j];
             int x = c[isler[i-1]][j];
             int y = c[isler[i]][j-1];
             int z;
             z = max(x, y);
             c[isler[i]][j] = z + opt[isler[i]][j];
```

```
            enb1=c[isler[i]][j];
           }
    }

printf ("\n yeni sıralamanın toplam akış zamanı %f ",enb1);getch();


float DELTAS=(enb1 - enb)/enb;
printf ("\n Sıralamaların Karşılaştırılması=%f", DELTAS);
if ( DELTAS <= 0)
    {
       if ( enb1 <= enb ) enb=enb1;
       accept = accept + 1;
       printf ("\n kabul toplamı=%f",accept);getch();
       float DELTAG = (enb1 - enb2)/enb2 ;
       printf (" \n delta G =%f",DELTAG);
       if ( DELTAG <= 0)

           {
            if ( enb2 <= enb ) enb=enb2;
            frcount=0;printf ("\n donma=%f", frcount);
           }
       else  goto top;
    }

else
    {

       randomize();
       float  yeni_sayi;
       float y,a;
       yeni_sayi= rand() % 100;
```

```
        a= yeni_sayi;
        float u=a/100.0 ;
        printf (" \n seçilen yeni rassal sayı =  %f ",u ); getch();
        float p = exp (-DELTAS / t);
        printf (" \n seçilen p sayısı =  %f ",p );getch();

        if ( u < p)
            {
             if ( enb1 <= enb ) enb=enb1;
             accept = accept +1;
             printf ("\n kabul toplamı=%f",accept);getch();
            }
        else goto top;
    }

top: total = total + 1;
printf ("\n toplam toplamı=%f",total);getch();
if ( total > 4 * n || accept > n )
    {
        float oran = (accept/total)*100;
        printf ( "\n kabulün toplama oranı= %f", oran);getch();
        if (oran <= 15 )
            {
             frcount = frcount + 1;printf ("\n donma=%f", frcount);
            }

        t=0.9*t ;printf ("\n sıcaklık = %d",t);
        accept = 0;
        total = 0;
        goto devam3;

    }
```

```
    else
        {
            fprintf (sonuc, " \n     %d                    %f        ", t, enb);
            goto top1;
        }
  }
    don: printf (" \n çözüm Donmuştur...");
    fclose (sonuc);

}
```

ÖZGEÇMİŞ

1968 yılında Samsun'da doğdu. İlk ve orta dereceli tahsilini İzmir'de tamamladı. 1986 yılında İ.T.Ü. Sakarya Mühendislik Fakültesi Endüstri Mühendisliği Bölümün'ne girdi. 1991'de mezun oldu. 1995 yılında SAÜ Fen Bilimleri Enstitüsü Endüstri Mühendisliği EABD programına kayıt oldu. Halen Atatürk Üniversitesi Mühendislik Fakültesi Endüstri Mühendisliği Bölümü'nde Araştırma Görevlisi olarak çalışmaktadır.