T. C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GENETİK ALGORİTMALARLA PERMÜTASYON TİPİ İŞ SIRALAMA

YÜKSEK LİSANS TEZİ

End. Müh. Muharrem DÜĞENCİ

Enstitü Anabilim Dalı : Endüstri Mühendisliği

Enstitü Bilim Dalı : Endüstri Mühendisliği

Eylül 1996

T. C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GENETİK ALGORİTMALARLA PERMÜTASYON TİPİ İŞ SIRALAMA

YÜKSEK LİSANS TEZİ

End. Müh. Muharrem DÜĞENCİ

Enstitü Anabilim Dalı : Endüstri Mühendisliği

Enstitü Bilim Dalı : Endüstri Mühendisliği

Bu tez .... / .... / 19.... tarihinde aşağıdaki jüri tarafından Oybirliği/Oyçokluğu ile kabul edilmiştir.

Doç. Dr. Ercan ÖZTEMEL	Yrd. Doç. Dr. Cemalettin Kubat	Y. Doç. Dr. İ. Hakkı CEDİMOĞLU
Jüri Başkanı	**Jüri Üyesi**	**Jüri Üyesi**

ÖNSÖZ

Son yıllarda gerek teorisi gerekse uygulamalarıyla Dünya ve Türkiye bilimsel gündeminde en üst sıralarda yer alan konulardan biri Yapay Zeka ve Yapay Zeka teknikleridir. Türkçe kaynakların bulunmadığı, yabancı kaynaklara ulaşmanın zor olduğu ve daha da önemlisi bu sahada çalışmaları yönlendirecek bilim adamlarının sayısının az ve ulaşılmasının güç olduğu bir dönemden geçilmektedir. Böyle bir dönemde konunun birinci derceden otoritesi sayılan sayın hocam Doç. Dr. Ercan Öztemel'in danışmanlığında, Yapay Zeka tekniklerinden Genetik Algoritmalar konusunda tez yapmak benim için en büyük gurur vesilesidir. Bu vesile ile sayın hocam Doç. Dr. Ercan ÖZTEMEL'e sonsuz teşekkürlerimi arz ederim.

Bununla birlikte çizelgeleme konusunda kaynak temini ve yol göstermede yardımlarını esirgemeyen hocalarım; İ. Hakkı CEDİMOĞLU ve M.Fatih TAŞGETİREN'e, programlama konusunda fikir alışverişinde bulunduğum arkadaşım Mümtaz İPEK'e, Endüsti Müh. bölümü çalışanlarına ve Fen Bilimleri Enstitüsü sekreteri Fatma AYDIN'a teşekkürlerimi bir borç bilirim.

Ayrıca tez metninin yazılmasında yardımlarını esirgemeyen eşime teşekkür ederim.

Eylül 1996, Adapazarı

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Bu çalışmada, canlılardaki genetik süreci bilgisayar ortamına taşıyan ve başarılı uygulamaları ile araştırmacıların dikkatini üzerine çeken genetik algoritmaların (GA) iş sıralama problemlerine uygulanması gösterilmiştir. Oluşturulan örnek problemler hem GA hem de geleneksel sezgisel metodlar ile çözülmüş ve sonuçların bir karşılaştırılması yapılmıştır. Özellikle iş ve makina sayısının arttığı durumlarda, genetik algoritmaların diğer metodlardan daha başarılı sonuçlar verdiği gözlemlenmiştir.

ANAHTAR KELİMELER : Genetik algoritmalar, İş sıralama, Genetik arama

PERMUTATION JOB SEQUENCING USING GENETIC ALGORITHMS

SUMMARY

There has been an increasing trend in using genetic algorithms (GAs) in engineering. This has been realized by numerous successful applications. In this study, the application of genetic algorithms to job sequencing problems is introduced. Several problems are solved by both GAs and well-known heuristics. The results and their comparison are presented in the paper. It has been proven that GAs are more successful than the others, especially when the large numbers of jobs and machines are considered

KEYWORDS : Genetic algorithms, Job sequencinng, Genetic search

BÖLÜM 1. GİRİŞ

Teknolojinin hızla geliştiği günümüzde, bilgi ve bilginin kullanımı, problemlere başarılı çözümler için oldukça önemli bir araç olmuştur. Yapay zeka teknikleri "*bilgi yoğun*" teknikler olduklarından; yani bilgiyi kullanma ve bilgisayarlaştırma metodlarını ortaya koyduklarından, hayatın her alanında kullanılmakta olup çalışmalar laboratuarlardan gerçek hayata inmeğe başlamıştır. Özellikle endüstriyel hayata yapay zeka sistemlerinin daha verimli, daha ucuz, daha kaliteli ve daha kısa zamanda üretim gibi katkıları, onlara olan ilgiyi son zamanlarda giderek artırmıştır (Öztemel, 1995). Önceleri, uzman sistemler ile başlayan bu yöndeki gelişmeler, yapay sinir ağları ve genetik algoritmaların eklenmesi ile devam etmektedir. Bu çalışmada genetik algoritmaların endüstriyel problemlere nasıl çözüm getirdikleri, akış tipi atölye iş sıralaması problemi üzerinde gösterilecek ve sonuçların bilinen geleneksel çözüm metodları ile bir karşılaştırması yapılacaktır.

Genetik algoritmalar, temeli doğal yaşam sürecine dayalı bir çözüm araştırma metodudur. Bir optimizasyon problemini, sınırlı karakterlerden ve uzunluklardan oluşan kromozom denilen diziler halinde kodlayarak ve bu kromozomların oluşturduğu alternatif çözümler topluluğu üzerinde genetik işlemleri uygulamak suretiyle çözmeye çalışır. Genetik işlemleri uygulayacak en sık kullanılan genetik operatörler ise çaprazlama, mutasyon ve yeniden oluşturmadır (Lam , 1996).

Canlılardaki genetik süreci bilgisayar ortamına taşıyan genetik algoritma yaklaşımı üzerindeki çalışmalar, özellikle son 30 yıl içerisinde oldukca hızlı bir ivme kazanmıştır. Genetik algoritma çalışmaları, 1970'li yıllarda Holland'ın yaptığı araştırmalar ile yoğun olarak gündeme gelmeye başlamıştır. Onun öğrencisi olan Goldberg'in çalışmaları ise genetik algoritmaların bir çok alanda uygulanabileceğini göstermiştir (Goldberg, 1989).

Genetik algoritmalar daha çok, matematiksel modeli kurulamayan, çözüm alanı çok geniş ve probleme etki eden faktörlerin fazla olduğu problemlerin çözümünde kullanılmaktadır. İş çizelgeleme problemleri de endüstride üzerinde önemle durulan temel problemlerden birisi olması nedeniyle uygulama sahası olarak iş çizelgeleme problemleri seçilmiştir.

İş çizelgelemenin ana amacı, "atölyedeki işler hangi sıra ve/veya program ile yürürlüğe konulursa makina ve zaman kullanımı en iyi düzeyde gerçekleştirilebilir, işgücü verimliliği artırılabilir ve işler müşterinin teslimini istediği tarihe yetiştirilebilir" sorularına cevap aramaktır. Bu amaç doğrultusunda işlerin hangi makinalarda ne zaman işlem görecekleri ve makinaların boş ve dolu oldukları zaman dilimleri önceden belirlenerek ona göre bir program yapılmak istenmektedir. Teslim tarihleri de gözönünde bulundurularak yapılacak bu program, ana hedefi yakalamaya yarayacaktır.

İş çizelgeleme problemlerini genel olarak, akış tipi atölye iş çizelgeleme ve atölye tipi iş çizelgeleme olmak üzere iki grupta toplayabiliriz. Bu çalışmada; atölye tipi çizelgeleme problemleri hakkında genel bilgi verilmiş ve daha sonra da akış tipi atölye iş çizelgelemenin özel bir durumu olan permütasyon tipi iş sıralama problemleri ve bu problemler için önerilen sezgisel yöntemlere birer örnekle değinilmiştir.

Tezin 2. ve 3. bölümlerinde sırası ile genel çizelgeleme problemleri, permütasyon tipi iş sıralama, sezgisel algoritmalar ve genetik algoritmalar tanıtılmaktadır, 4. bölümünde, genetik algoritmalar ile iş sıralama probleminin çözümü ve 5. bölümde sonuçların karşılaştırması yapılmaktadır.

BÖLÜM 2. İŞ ÇİZELGELEME

İş çizelgeleme problemi hangi işin hangi makinada ve hangi sırayla işleme konulması gerektiğine karar verme olayı olarak tanımlanabilir. İş çizelgeleme problemine çeşitli yaklaşımlarda bulunulmuştur. Bu yaklaşımlar iki başlık altında sınıflandırılabilir.

Atölye tipi çizelgeleme

Akış tipi çizelgeleme

Bu bölümde ilk olarak genel atelye tipi çizelgeleme problemleri ve özellikleri tanıtılacak daha sonra da akış tipi çizelgelemenin özel bir durumu olan permutasyon tipi çizelgeleme üzerinde durulmaktadır.

2.1 Genel Atelye Tipi Çizelgeleme Problemi

Atelye-tipi üretim çizelgeleme probleminde; m tane $\{M_1, M_2, \ldots M_m\}$ makinada işlenmek üzere n tane $\{J_1, J_2, \ldots J_n\}$ 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 atelye 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 olduğunda özel bir durum ortaya çıkar. Böyle durumlarda problem akış-tipi çizelgeleme problemi olarak adlandırılır. Akış-tipi ve atelye-tipi çizelgeleme arasındaki fark ilerki bölümlerde daha 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, işi

yürütmek üzere gereken makinayı ayarlama veya hazırlama için gerekli olan zamanın, yani, hazırlık zamanı, 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, 1982).

Dolayısıyla bu tezde önemli bir varsayım yapılmıştır. O da her operasyon zamanları teslim tarihleri gibi sayısal miktarlar deterministiktir ve çizelgeleyici tarafından önceden bilinir.

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. İşlenmek üzere i. işin atelyede hazır olduğu zamana i. işin hazır zamanı denir ve r_i ile gösterilir.

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

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

Yukarıda bazı varsayımlar problemin tanımı içinde verildi. Bunlara ek olarak atelye tipi çizelgeleme problemini genelleştirmek için bazı tanımların yapılmasına gerek vardır. Bunlar şöyle sıralanır (French, 1982).

1. **Herbir iş bir bütündür** : İş farklı operasyonlardan oluşmasına rağmen, aynı işin iki operasyonu hiçbir şekilde aynı anda işlenemez. Böylece bazı pratik problemleri tartışmamızın dışına çıkarabiliriz. Yani son ürün için montaj edilmeden önce alt parçaların aynı anda imal edildiği durumlar gibi.
2. **İş Bölme Yoktur** : Herbir operasyon, başladığı zaman, diğer operasyon o makinada başlatılmadan önce tamamlanmalıdır.
3. **Herbir iş, herbir 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** : Herbir iş tamamlanıncaya kadar işlenmelidir.

5. **İşlem zamanları çizelgeden bağımsızdır** : Burada iki şey varsayılmaktadır :
   - Herbir 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 edilmektedir.
6. **Ara stoğa izin verilir** : İşler bir sonraki makinanın boşalması için bekleyebililr. Bu önemli bir varsayımdır. Örneğin, çelik mil üretilirken demirin sıcak oluşundan dolayı iş bir sonraki operasyon için beklemek zorunda kalabilir.
7. **Makinanın herbir tipinden sadece bir tane vardır** : İşlerin işlenmesi esnasında aynı işi yapan birden fazla makinanın olmadığı varsayılır. Bu varsayım, "beklemek"ten 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 nicel değerler bilinir ve sabittir.

2.1.1. Performans ölçütleri

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

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

Kısaca, amaçlar çok farklı, çeşitli ve birbirleriyle çelişir olabilir. Ama çizelgeleme probleminin etkinliğinin ölçülmesi için tanımlanması gereklidir. Performans ölçütlerini kesin matematik terimlerle tanımlamadan önce bazı tanımlar ve notasyonların verilmesi gerekir. Bunlar:-

r_i : i işinin hazır zamanı

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

a_i : J_i işinin teslim tarihi

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

W_{ik} : J_i işinin k. operasyonu için bekleme zamanı. k. operasyonla M_k makinası kastedilmemekte (k.makina da olabilir), k. işlem sırasında gelen herhangi bir makina kastedilmektedir. Böylece eğer J_i işinin teknolojik kısıtı $M_{j(1)}, M_{j(1)}, M_{j(1)}, \ldots, M_{j(m)}$ 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 işinin akış zamanı (yani J_i işinin atelyede harcadığı zaman)

$$F_i = C_i - r_i \quad (4)$$

L_i : J_i nin gecikmesi (yani işin tamamlanma zamanı ile teslim tarihi arasındaki farktır.) $L_i = C_i - d_i$. (5)

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

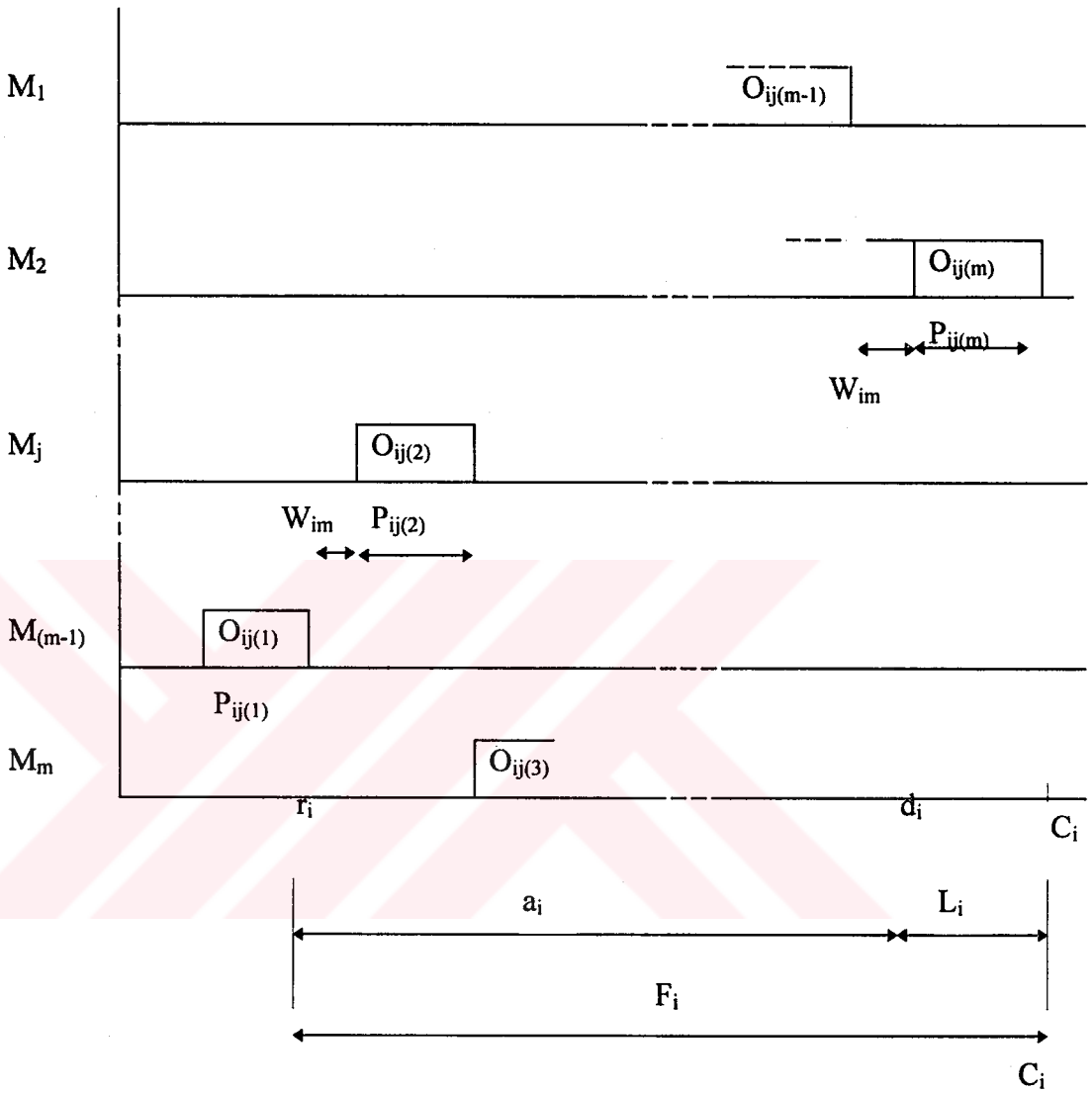
T_i : J_i işinin pozitif gecikmesi. $T_i = \max\{L_i, 0\}$ (6)

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

Tipik bir iş için bu miktarlar Şekil 2.1'deki Gant diyagramında gösterilmiştir.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} in ise sıfırdan farklı olduğu görülmektedir. Tamamlanma zamanı, teslim tarihinden sonra olduğundan bu iş için $T_i = L_i$ ve $E_i = 0$ dır.

Yukarıdaki terminolojide bazı şaşırtıcı ifadeler olabilir. Zaman kavramı iki farklı manaya sahiptir. Zaman, hem bir zaman aralığını, hemde belli bir zaman noktasını ifade eder. Dolayısıyla 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 alınır. Örneğin, $\overline{F}$ ortalama akış zamanı, C_{max} ise maksimum tamamlanma zamanını gösterir..



Şekil 2.1: Herhangi bir J_i işi için gant diyagramı.

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

Son olarak belli bir zamanda çeşitli durumlarda işlerin sayısını hesaplamak için bazı değişkenleri tanımlamak gerekirse.

$N_W(t)$: t zamanında işlenmeye hazır olmayan veya makinalar arasında bekleyen işlerin sayısı,

$N_p(t)$: t zamanında gerçekten işlenmekte olan işlerin sayısı,

$N_c(t)$: t zamanında tamamlanmış olan işlerin sayısı,

$N_u(t)$: t zamanıyla tamamlanacak olan işlerin sayısı.

Bu değişkenler arası ilişkiler şöyledir.

$$N_W(t) + N_p(t) + N_c(t) = n \qquad (\text{bütün } t\text{'ler için})$$

$$N_W(t) + N_p(t) = N_u(t) \qquad (\text{bütün } t\text{'ler için})$$

$$N_u(0) = n,$$

$$N_u(C_{max}) = 0.$$

$$\overline{N}_u = \frac{1}{C_{ma}} \int_0^{C_{max}} N_u(t)dt$$

Bazı performans kriterleri yukarıdaki miktar tanımlamaları yardımıyla açıklanabilir.

2.1.1.1.Tamamlanma zamanına dayanan kriterler

Bu kategorideki temel kriterler, $F_{max}, C_{max}, \overline{F}$ ve $\overline{C}$ dir. F_{max}'ı enazlamak, maksimum akış zamanı (F_{max}) , 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. Hazır zamanların tümünün sıfır olması durumunda C_{max} ve F_{max} birbirine eşit demektir. Gerçekte, eğer bir iş son derece geç hazır zamana sahipse, en kısa akış zamanına sahip olan işin C_{max} da tamamlanacağı açıktır. Burada C_{max}'ı aynı zamanda **toplam üretim zamanı** veya **yapım zamanı**

olarak da isimlendirmek uygun olacaktır. $\overline{F}$'yı minimize etmek, ortalama akış zamanı, çizelge maliyetinin doğrudan doğruya tek bir işi işlemek için geçen sürenin ortalamasıyla ilgili olduğunu ifade eder. $\overline{C}$'yı minimize etmek, ortalama tamamlanma zamanı, $\overline{F}$'yı minimize etmeye eşdeğerdir. Başka bir değişle, minimum $\overline{C}$'ya erişen çizelge aynı zamanda minimum $\overline{F}$ ya da erişecektir ve bunun tersi de doğrudur. C_{max} ve F_{max} oldukça farklı performans kriterler 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 yazarlar, bazı işlerin diğerlerinden daha önemli varsayımına dayanarak ağırlıklı performans kriterlerini kullanırlar. Bu durumda performans kriterleri aşağıda görüldüğü gibi ağırlıklı ortalamayı minimize etme esasına dayanır (Taşgetiren, 1988).

$$\sum_{i=1}^{n} \alpha_i C_i \quad \text{veya} \quad \sum_{i=1}^{n} \beta_i F_i$$

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

2.1.1..2. Teslim tarihine dayanan kriterler

Çizelge maliyeti genellikle hedef teslim tarihlerine ne kadar yaklaşıldığı 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. $\overline{L}$'yi veya L_{max}'ı minimize etmek, bir işi erken tamamlamak için bir ödül (getiri) söz konusu olduğu zaman uygundur. Bu durumda iş ne kadar erken tamamlanırsa ödül o kadar artacaktır. Tam tersine, $\overline{T}$ ve T_{max} 'ı minimize etmek, erken tamamlanan işler herhangi bir ödül getirmiyorsa uygun olacaktır. Bu durumda ise geç kalan işler için ceza maliyetleri söz konusu olacaktır. Başka bir deyişle, 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. Örneğin, yakıtı bitmeden bir dakika önce inmesi gerektiği gibi çizelgelenen bir uçağın geç kaldığını düşününüz. Sonuç felaket olacaktır. 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.

2.1.1.3. Stok ve yararlanma 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 istenilebilir. Eğer amaç, makinaların en verimli şekilde kullanılmasını sağlamak olursa, herhangi bir zamanda halıhazırda işlenmekte olan işlerin ortalama sayısını $(\overline{N_p})$, maksimize etmek amaçlanabilir. Alternatif olarak, makinaların etkin kullanımı amaçlandığında veya ortalama $(\bar{I})$ veya maksimum makina 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, $C_1, C_2, \ldots, C_n$'nin bir fonksiyonudur. Yani ;

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

Buradan;

$$R(C_1, C_2, \ldots, C_n) \le R(C_1', C_2', \ldots, C_n') \text{ elde edilir.}$$

Bu tanımın temelindeki mantık şudur : İki çizelgeden birinci çizelgedeki bütün işler ikinci çizelgedeki bütün işlerden hiçbir şekilde daha geç tamamlanmadığını varsayılsın. Bu durumda, düzenli performans kriteri tanımı altında, birinci çizelge en

az ikinci kadar iyi bir çizelgedir. Burada dikkat edilmesi gereken şey 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. Örneğin, performans kriteri C_{max}'ı gözönüne alalım.

$$R\{C_1, C_2, ..., C_n\} = C_{max}$$
$$= \max\{C_1, C_2, ..., C_n\}.$$

Düzenli performans kriteri tanım gereğince,

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

$$R\{C_1, C_2, ..., C_n\} = \max\{C_1, C_2, ..., C_n\}$$
$$\le \max\{C_1', C_2', ..., C_n'\}$$
$$= R(C_1', C_2', ..., C_n')$$

dolayısıyla C_{max} düzenli bir performans kriteridir.

2.1.2. 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 durum.

P : Permutasyon tipi çizelgeleme problemi; bu durumda sadece makina sırası aynı değil aynı zamanda herbir makina için iş sırasının da aynı olduğu durumdur.

G : Genel atelye tipi çizelgeleme problemi; teknolojik kısıtların oluşumu üzerinde hiçbir kısıtın bulunmadığı durum.

B : Çizelgenin değerlendirileceği performans kriteri demektir.

Bu tezin konusu permutasyon tipi çizelgeleme olduğu için diğer çizelgeleme tipleri detaylı olarak anlatılmayacaktır. Bunların detayları (French, 1982) da bulunabilir.

2.2. Permutasyon Tipi İş Sıralama

Permütasyon tipi iş sıralama, işlerin başlangıçta tümünün işlem görmeye hazır olduğu, her işin her makinada mutlaka operasyonu olduğu ve işlerin makinalardaki işlem görüş 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.

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

- Bir iş bir makinaya atandığında, bitirilmeden aynı makinaya başka bir iş atanamaz.
- 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.

Bu çalışmada, önce işlem süreleri 2 ile 6 dakika arasında uniform olarak dağılan 20-iş 4-makina, daha sonra işlem süreleri 4 ile 20 dakika arasında uniform dağılan 50-iş 10-makina problemleri ele alınmıştır. Herhangi bir i. işin j. makinadaki işlem süresi P(i,j) ve iş permütasyonu (iş sırası){J1,J2,..Jn} olarak kabul edilirse işlerin makinalardaki tamamlanma zamanları $C(J_i,j)$ ve ona bağlı olarak maksimum tamamlanma zamanı (C_{max}), ortalama iş akış zamanı ($\overline{C}$) ve ortalama pozitif gecikme ($\overline{T}$) aşağıdaki gibi hesaplanır.

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

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

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

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

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

$\overline{C} = \sum_i C(J_i, m) / n$ $\quad i = 1..n$

$T_i = \text{Max}\{0, C(J_i, m) - d_i\}$ $\quad i = 1..n$

$\overline{T} = \sum_i Ti / n$ $\quad i = 1..n$

Ortalama gecikme bulunurken işlerin teslim tarihleri (d_i), toplam işlem zamanına dayalı olarak teslim tarihi tesbit eden TWK yöntemi ile k=2 alınarak hesaplanmıştır. Burada k'nın 2 alınmasının nedeni yeteri miktarda pozitif gecikme üretilmesini sağlamaktır.

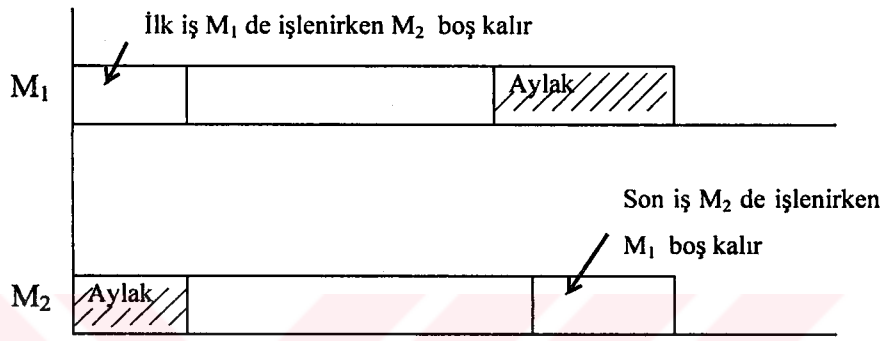
$$d_i = k * \sum_j P_{ij}$$

2.3. Akış Tipi Çizelgeleme Problemi Çözüm Metodları

Akış tipi atölye iş sıralama problemlerinin çözüm algoritmalarına bakıldığında optimizasyon yöntemlerinin 3 ve daha az sayıda makina söz konusu olduğunda çalıştıkları, makina sayısının 4 ve daha yukarısı olduğu durumlarda ise sezgisel yöntemlerin kullanıldığı görülür. Günlük hayatta makina sayısının 4 ten çok olduğu açıktır. Geliştirilen birçok sezgisel algoritmanın da, 2-makina için (özel şartlarda, 3-makina için) optimal sonucu veren Johnson algoritmasına dayandırılması nedeni ile önce Johnson algoritması daha sonra sezgisel yöntemler kısaca tanıtılacaktır.

2.3.1. Johnson algoritması

Johnson algoritması akış tipi atölye problemlerinde tamamlanma zamanına dayalı performans kriterlerine göre makina sayısına bağlı olarak optimal ve optimale yakın sonuçlar veren bir algoritmadır. İki makinada işlenmek üzere n iş (n/2/F/Fmax problemi) ve işlerin makinalara geliş sıralarının M_1, M_2 olduğunda maksimum akış zamanının nasıl enküçükleneceğini göstermektedir. Burada hazırlık zamanlarının sıfır olmasıyla Fmax=Cmax olmaktadır. M_2 makinasında işleme en erken başlamak için; M_1 makinasında en küçük işlem zamanlı işlerin önce başlatılması uygun görülmektedir (Şekil 2.2). Bir işin M_1 de işlenmesi için M_1'in boş olması gerekmektedir.



Şekil 2.2: Her iş sırasında oluşan atıl zamanlar

Bu bölümde Johnson algoritması, iki makinalı permütasyon tipi iş sıralama probleminde gösterilmektedir. Tüm bu kabulleri bir arada düşünüldüğünde M_1 makinasında kısa işlem süresine sahip olan işler ön sıralara M_2 makinasında kısa işlem süresine sahip olan işler son sıralara alınacak şekilde bir iş sırası $(J_1,J_2,J_3,...J_n)$ aranmaktadır. Johnson tarafından geliştirilen algoritma bu özellikleri sağlamaktadır.

Algoritma işlem sıralarını, uçlardan (baş ve son) ortaya doğru tespit eden bir yapıya sahiptir. Bunu yapmak için iki sayaca (k,h) ihtiyaç vardır. Başlangıçta k=1'dir ve 2,3,4,.. olarak artar ve işlem sırasındaki birinci, ikinci, üçüncü.... pozisyonları belirtir. Aynı şekilde h=n dir (n-1),(n-2)... olarak azalır ve n. ,(n-1). .... sıraları gösterir. Algoritmanın adımları şöyledir:

$a_i=P_{i1}$, j_i işinin M_1'deki işlem süresi

$b_i=P_{i2}$, j_i işinin M_2'deki işlem süresi olarak verilirse

Adım 1: Başlangıçta k=1, h=n olarak al.

Adım 2: Henüz sıraya konmamış işlerin listesini $(J_1,J_2,J_3,...,J_N)$ oluştur.

Adım 3: Henüz sıraya konulmamış işler arasında en küçük $a_i$ve b_i işlem zamanlarına sahip işleri tespit et.

Adım 4: J_i işi için en küçük işlem zamanı ilk makinada ise, yani a_i en küçükse

(i) J_i işini işlem sırasının k. pozisyonuna yerleştir.

(ii)J_i işini sıralanmamış işler listesinden sil

(iii)k'yı bir artır, k=k+1.

(iv)Adım 6'ya git.

Adım 5: J_i işi için en küçük işlem zamanı 2. makinada ise, yani b_i en küçükse;

(i) J_i işini işlem sırasının h. pozisyonuna yerleştir.

(ii)J_i işini sıralanmamış işler listesinden sil.

(iii)h'ı bir azalt, h = h - 1.

(iv)Adım 6'ya git.

Adım 6: Hala sıralanmamış iş varsa Adım 3'e git, aksi taktirde sıralama tamamlanmıştır.

Eğer en küçük işlem zamanı birden fazla işe ait ise sıra ile atamalar yapılır.

Örnek: Tablo 2.1'de işlem zamanları verilen $7/2/P/F_{max}$ problemi üzerinde Johnson algoritmasının uygulanması;

Tablo 2.1: $7/2/P/F_{max}$ Problemi işlem zamanları

İş	İşlem Zamanları a_i (M_1)	b_i (M_2)
1	6	3
2	2	9
3	4	3
4	1	8
5	7	1
6	4	5
7	7	6

İş 4 çizelgelenir:	4	--	--	--	--	--	--
İş 5 çizelgelenir:	4	--	--	--	--	--	5
İş 2 çizelgelenir:	4	2	--	--	--	--	5
İş 3 çizelgelenir:	4	2	--	--	--	3	5
İş 1 çizelgelenir:	4	2	--	--	1	3	5
İş 6 çizelgelenir:	4	2	6	--	1	3	5
İş 7 çizelgelenir:	4	2	6	7	1	3	5

Böylece iş sırası (4-2-6-7-1-3-5) olarak bulunur. Yukarıda iki tane seçenekli atama vardır. İş 4'ü ilk sıraya atamadan iş 5'i son sıraya atayabiliriz. Bu öncelik sonuç sırasını değiştirmez. Aynı şekilde 6. iş sırasına, iş 3 yerine iş 1 de atanabilir. Bu değişiklik iş sırasını (4-2-6-7-3-1-5) olarak değiştirmesine rağmen maksimum akış zamanını değiştirmez.

2.3.2. Sezgisel Yöntemler

Sezgisel yöntemler makina sayısının 4 veya daha fazla olduğu durumlarda kullanılırlar ve optimal çözümü garanti etmezler (French, 1982). Her ne kadar optimal veya optimale yakın sonuçlar üretebilmekte iseler de iş sayısı (n) ve makina sayısının (m) arttığı durumlarda sezgisel yöntemlerin etkinliği de azalmaktadır. Aşağıda bu konuda en çok kullanılan algoritmaların kısaca özeti verilmiştir.

2.3.2.1. Palmer'in eğim dizini (Palmer Slope Index - PSI) yöntemi

Palmer (1965) her işe bir dizin değeri tanımlayarak bu değere dayalı bir algoritma geliştirdi. Bu yöntem, işlem zamanı ilk makinalarda kısa olanları öne, uzun olanları ise sona alacak bir eğim dizini tanımlayarak ve bu eğim dizini değerine göre azalan bir iş sırası tanımlayıp optimale yakın sonuç elde etme esasına dayalı bir yöntemdir. Bu algoritmada işlerin önceliği hesaplanan S_i eğim dizinine göre belirlenir. Eğim dizini, S_i, genel n/m/F/F_{max} problemi için şu şekilde hesaplanır:

$$S_i = -(m-1)p_{i1} -(m-3)p_{i2}-...+(m-3)p_{i(m-1)}+(m-1)p_{im}$$

$$S_i = \sum_{j=1}^{m}\left[m-(2j-1)\right]p_{ij}$$

Belirli bir n/7/F/F_{max} problemi için;

$$S_i=-6p_{i1}-4p_{i2}-2p_{i3}+0p_{i4}+2p_{i5}+4p_{i6}+6p_{i7}$$

Burada m makina sayısını, Pij ise i. işin j. makinadaki işlem süresini gösterir. Büyük eğim dizini değerleri sondaki makinada işlem süresi uzun olan işlere verilmektedir. Sıralamada öncelik büyük S_i değerli işe verilmektedir. Aynı S_i değerli işler için işler yanyana listelenir bu işler arasında bir öncelik sözkonusu olmaz.

İşlem zamanları makinadan makinaya artma eiğilimi gösterdiğinde ilk terimlerin toplamının küçük ve negatif, sonraki terimlerin toplamının büyük ve pozitif olduğu görülmektedir. Bu da bu eğim dizininin istenen özelliğe sahip olduğunu gösterirr.

Bu algoritma işlem zamanları Tablo 2.2. de verilen örnek bir 4/3/F/F_{max} problemine uygulanırsa:

Tablo 2.2 : 4/3/F/F_{max} Problemi İşlem Zamanları

İş J_i	İşlem Zamanları p_{i1}	p_{i2}	p_{i3}
1	1	8	4
2	2	4	5
3	6	2	8
4	3	9	2

Eğim dizinleri;

$S_1 = -2*1+0*8+2*4=6$

$S_2 = -2*2+0*4+2*5=6$

$S_3 = -2*6+0*2+2*8=4$

$S_4 = -2*3+0*9+2*2=-2$

S_i değerleri büyükten küçüğe sıralandığında bu algoritma iş sırası (1-2-3-4) veya (2-1-3-4)'ü tavsiye etmektedir. Alternatif iki sıranın F_{max} performans kriterine göre değerlendirildiğinde her iki sıranında aynı değere (F_{max}=28) sahip oldukları görülür.

2.3.2.2. CDS algoritması

Champbell ve arkadaşları (1970) tarafından geliştirilen bu teknikte, n-iş m-makina problemini önce, (m-1) adet n-iş 2-makina problemi şekline dönüştürülür. Daha sonra Johnson algoritmasına göre problemleri tek tek çözülerek en iyi sonucu (C_{max}'ı en küçük olanı) veren iş sırası ana problemin de (n-iş m-makina) iş sırası olarak kabul edilir. Dönüştürme işlemi aşağıda gösterildiği gibidir. Birinci makinadaki işlerin işlem zamanları $a_i^{(k)}$, ikinci makinadaki işlerin işlem zamanları $b_i^{(k)}$ olarak gösterilir ise:

$$a_i^{(k)} = \sum_{j=1}^{k} P_{ij}$$

$b_i^{(k)} = \sum_{j=m-k+1}^{m} P_{ij}$ olarak hesaplanır.

Burada $a_i^{(k)}$, k. iki makina probleminde i. işin birinci makinedeki işlem zamanını, $b_i^{(k)}$ ise aynı problemde i. işin ikinci makinadaki işlem zamanını gösterir. Bir örnekle açıklamak gerekirse Tablo 2.3 de gösterildiği gibi 4-iş 3-makina probleminin çözümü için; önce iki tane (3-1=2) 4-iş 2-makina problemi haline getirilir ve bunlar Johnson algoritmasına göre çözülür.

Tablo 2.3. : 4-iş 2-makina probleminin iki ayrı 4-iş 2-makina problemine dönüştürülmesi

Ana Problem işlem zamanları				I. iki makina problemi k=1		II. iki makina problemi (k=2)	
işler (i)	P_{i1}	P_{i2}	P_{i3}	$a_i^{(1)} = P_{i1}$	$b_i^{(1)} = P_{i3}$	$a_i^{(2)} = P_{i1} + P_{i2}$	$b_i^{(2)} = P_{i3} + P_{i2}$
1	1	8	4	1	4	9	12
2	2	4	5	2	5	6	9
3	6	2	8	6	8	8	10
4	3	9	2	3	2	12	11

Johnson algoritması birinci 4-iş 2-makina problemine uygulandığında, 1-2-3-4 iş sırası elde edilir. İkinci 4-iş 2-makina problemi için de Johnson algoritması uygulandığında 2-3-1-4 iş sırası elde edilir. Birinci çözümün önerdiği iş sırası ile ana problemi çözdüğümüzde $F_{max}=28$, ikinci sırayı uyguladığımızda ise $F_{max}=29$ olarak bulunur. Bu durumda daha küçük maksimum akış zamanını ($F_{max}=28$) veren iş sırası (1-2-3-4) ana problemin de çözümü olarak kabul edilir.

2.3.2.3. Dannenbring algoritması

Dannenbring (1977), Palmer'in (1965) geliştirdiği PSI ve Campbell, Dudek ve Smith'in (1970) geliştirdikleri CDS algoritmalarının avantajlarını birleştirerek yeni bir sezgisel metod tanımladı. Onun fikri CDS algoritmasında olduğu gibi problemi yalnızca bir adet 2-makina problemine dönüştürerek Johnson algoritmasını uygulamak fakat aynı zamanda 2-makina problemine dönüştürmek için birleştirilen işlem zamanlarının da Palmer'in eğim dizinine göre hesaplanmasını öngörmektedeir. Bu durumda hem CDS'nin tek bir 2-makina problemine çözüm arama avantajından hem de PSI'nın bir işe ait birden fazla makinadaki işlemleri tek bir değer ile temsil etme avantajından faydalanılmaktadır. İlk makina için sıkıştırılmış zamanlar;

İlk makina için

$$a_i = mp_{i1} + (m-1)p_{i2} + .. + 1p_{im}$$

$$= \sum_{j=1}^{m} (m-j+1)p_{ij}$$

İkinci makina için ;

$$b_i = p_{i1} + 2p_{i2} + .. + mp_{im}$$
$$= \sum_{j=1}^{m} jp_{ij}$$

Eğer ilk makinalarda gerçek işlem zamanları kısa olursa bu ilk makina için hesaplanan sıkıştırılmış işlem zamanı a_i'nin de küçük olmasına neden olacaktır. Aynı şekilde son makinalardaki işlem sürelerinin kısalığı da b_i'nin küçük olmasına etki edecektir. Örnek olarak Tablo 2.4 de verilen 4-iş 2-makina problemi ele alınırsa;

Tablo1.4. : 4-iş 3-makina problemi ve toplanmış işlem zamanları

	İşlem Zamanları			Toplanmış İşlem Zamanları	
J_i	P_{i1}	P_{i2}	P_{i3}	$a_i=3p_{i1}+2p_{i2}+p_{i3}$	$b_i=p_{i1}+2p_{i2}+3p_{i3}$
1	1	8	4	23	29
2	2	4	5	19	25
3	6	2	8	30	34
4	3	9	2	29	27

Hesaplanan a_i ev b_i değerlerine göre Johnson Algoritması uygulandığında iş sırası (2, 1, 3, 4) maksimum akış zamanı da $F_{max} = 28$ olarak elde edilir.

2.3.2.4. NEH (Nawaz, Enscore and Ham) algoritması

NEH algoritması, işleri toplam işlem sürelerine dayalı olarak sıralayan bir algoritmadır. İlk olarak işler tezgahlardaki toplam işlem zamanlarına göre büyükten küçüğe doğru sıralanır. Toplam işlem süreleri eşit olan işler için ayrı bir öncelik kuralı yoktur (Nawaz, M. ve diğerleri, 1983). Bu aşamadan sonra algoritmanın çalışma şekli aşağıdaki gibidir.

Sıralanan n iş içerisinden toplam işlem zamanları en yüksek olan iki iş seçilir. Bu işler kendi aralarında alternatif iki sıraya konularak sadece bu iki işin tamamlanma

süreleri hesaplanır (kısmi tamamlanma). Kısmi tamamlanma süresi küçük olan alternatif iş sırası seçilerek bu iki işin birbirlerine olan göreceli sırası (önceliği) böylelikle belirlenmiş olur. Bu göreceli sıra daha sonraki adımlarda da korunmak üzere sabitlenir. Daha sonra toplam işlem zamanı en büyük olan üçüncü iş seçilir ve üç alternatif yere ilk, orta veya son sıralara yerleştirerek bu alternatiflerin kısmi tamamlanma süreleri hesap edilir. Kısmi tamamlanma zamanı en küçük olan göreceli sıra bu işlerin birbirlerine olan önceliğini gösterir ve daha sonraki adımlar için bu süre sabitlenir. Bu işlem her seferinde yeni bir iş eklenerek bütün işlerin birbirlerine göre sıralamalarının belirlenmesine kadar tekrarlanır. Algoritmadaki iterasyon sayısı her adımda k-1 (k atanmış işler sayısı) olmak üzere n iş için toplam;

$$\frac{n(n+1)}{2}-1 \quad \text{dir.}$$

Algoritmanın adım adım işleyişi aşağıdaki gibidir.

Adım 1. Her i işi için, i. işin j. makinadaki işlem süresi P_{ij} olmak üzere toplam işlem sürelerini bul.

$$P_i = \Sigma_J P_{ij} \qquad (j= 1,2,...,m)$$

Adım2. İşleri Pi değerlerine göre büyükten küçüğe sırala.

Adım3. Adım 2'de sıralanan işlerden ilk ikisini al (k=2). Mümkün iki sıra için (1-2 veya 2-1) kısmı tamamlanma zamanlarını bul. Tamamlanma zamanı kısa olan alternatifi seçerek diğer adımlarda da bu sırayı koru, k'yı bir artır (k=k+1)

Adım 4. Adım 2'de üretilen listenin k. pozisyonundaki işi al. Bu işi bir önceki adımda yerine yerleştirerek kısmi tamamlanma süresi en küçük olan alternatifteki sıraya yerleştir. (3. iş ilave edildiğinde 3-1-2, 1-3-2, 1-2-3 olmak üzere 3 alternatif sıra vardır.)

Adım5. Eğer k=n ise sıralama tamamlanmıştır. Değilse bütün işler tamamlanıncaya kadar *Adım 4'ü* tekrarla.

Bu adımları Tablo 2.5 de verilen örnek bir 4-iş 5-makina problemi üzerinde göstererek çözelim.

Tablo 2.5 : 4-iş 5-makina problemi işlem zamanları

		Makinalar(m)				
		1	2	3	4	5
İşler (n)	1	5	9	8	10	1
	2	9	3	10	1	8
	3	9	4	5	8	6
	4	4	8	8	7	2

Adım 1. Her bir işin toplam işlem zamanları bulunur.

$P_1 = 5 + 9 + 8 + 10 + 1 = 33,$

$P_2 = 9 + 3 + 10 + 1 + 8 = 31,$

$P_3 = 9 + 4 + 5 + 8 + 6 = 32,$

$P_4 = 4 + 8 + 8 + 7 + 2 = 29.$

Adım 2. Büyükten küçüğe sıralanır. iş1 - iş3 - iş2 - iş4

Adım 3. İş1 ve iş3 alınır ve bu iki iş arasındaki en optimal sıra bulunur. Bu işlemler Tablo 2.6 ve Tablo 2.7 de gösterildiği gibi hesaplandığında en küçük maksimum akış zamanı ($\min C_{max}$) 42 olan iş3 - iş1 sırasının en iyisi olduğu görülür. Sonraki adımlarda da iş1 ve iş3'ün birbirlerine karşı göreceli sırası hep aynı (3-1) kalacaktır. Yani iş1, iş3'den sonra yapılacaktır. Daha sonra k = 3 olarak alınır.

Tablo 2.6. : Deneme sırası 1-3 için maksimum akış zamanı

		Makinalar				
		1	2	3	4	5
İşler	1	5/5	9/14	8/22	10/32	1/33
	3	9/14	4/18	5/27	8/40	6/46

Tablo 2.7. : Deneme sırası 3-1 için maksimum akış zamanı

		Makinalar				
		1	2	3	4	5
İşler	3	9/9	4/13	5/!8	8/26	6/32
	1	5/14	9/23	8/31	10/41	1/42

Adım 4. Adım 2 de belirlenen listedeki işlerden üçüncü pozisyondaki işi (iş2) al ve bu işi son adımda belirlenen 3-1 deneme sırasındaki tüm alternatif pozisyonlarda yerine koyarak optimal yeni sırayı bul. Daha önce belirlenen iki sıranın üç alternatif yerine yerleştirilebilir; başına, ortasına ve sonuna. Bu sıralar Tablo 2.8., Tablo 2.9. ve Tablo 2.10. da test edilmiş bunlar arasında 3-1-2 sırası en küçük maksimum akış zamanı ($minC_{max}$) 50 ile en iyi sonucu vermiştir.

Tablo 2.8. : 3-1-2 sırası için maksimum akış zamanı

		Makinalar				
		1	2	3	4	5
	3	9/9	4/13	5/18	8/26	6/32
İşler	1	5/14	9/23	8/31	10/41	1/42
	2	9/23	3/26	10/41	1/42	8/50

Tablo 2.9. : 3-2-1 sırası için maksimum akış zamanı

		Makinalar				
		1	2	3	4	5
	3	9/9	4/13	5/18	8/26	6/32
İşler	2	9/18	3/31	10/31	1/32	8/40
	1	5/23	9/32	8/40	10/50	1/51

Tablo 2.10. : 2-3-1 sırası için maksimum akış zamanı

		Makinalar				
		1	2	3	4	5
	2	9/9	3/12	10/22	1/23	8/31
İşler	3	9/18	4/22	5/27	8/35	6/41
	1	5/23	9/32	8/40	10/50	1/51

Adım 5. $i \neq n$, $i = 3 + 1 = 4$ ve sıralanmayan diğer işler için adım 4'den devam et.

Adım 4. Adım 2 de belirlenen iş listesinden dördüncü (i=4) sıradaki işi (iş4) al ve son 4. adımda birbirlerine karşı göreceli yerleri belirlenen 3-1-2 sırasında bu işi dört muhtemel yere yerleştirerek en iyi sırayı bul. İş4'ün 4 alternatif yere getirildiğindeki maksimum akış zamanları hesabı Tablo 2.11., Tablo 2.12., Tablo 2.13. ve Tablo 2.14. de gösterilmektedir. Bunlar içerisinde 4-3-1-2 sırası en küçük maksimum akış zamanı 54'ü vermektedir.

Tablo 2.11. : 3-1-2-4 sırası için maksimum akış zamanı

		Makinalar				
		1	2	3	4	5
İşler	3	9/9	4/13	5/18	8/26	6/32
	1	5/14	9/23	8/31	10/41	1/42
	2	9/23	3/26	10/41	1/42	8/50
	4	4/27	8/35	8/49	7/56	2/58

Tablo 2.12. : 3-1-4-2 sırası için maksimum akış zamanı

		Makinalar				
		1	2	3	4	5
İşler	3	9/9	4/13	5/18	8/26	6/32
	1	5/14	9/23	8/31	10/41	1/42
	4	4/18	8/31	8/39	7/48	2/50
	2	9/27	3/34	10/49	1/50	8/58

Tablo 2.13. : 3-4-1-2 sırası için maksimum akış zamanı

		Makinalar				
		1	2	3	4	5
İşler	3	9/9	4/13	5/18	8/26	6/32
	4	4/13	8/21	8/29	7/36	2/38
	1	5/18	9/30	8/38	10/48	1/49
	2	9/27	3/33	10/48	1/49	8/57

Tablo 2.14. : 4-3-1-2 sırası için maksimum akış zamanı

		Makinalar				
		1	2	3	4	5
İşler	4	4/4	8/12	8/20	7/27	2/29
	3	9/1	4/17	5/25	8/35	6/41
	1	5/18	9/27	8/35	10/45	1/46
	2	9/27	3/30	10/45	1/46	8/54

Adım 5. i=n (4=4) olduğundan adımlar tamamlanmıştır.

Bu problem için mümkün n! = 24 iş sırası içerisinden NEH algoritması 9 aşamada 4-3-1-2 iş sırasını optimal olarak bulmuştur.

NEH algoritmasının uzun iterasyonlara sahip olması nedeni ile yapılacak normal bir bilgisayar programı da bu iterasyon fazlalığından dolayı problem çözümünde diğer algoritma programlarına göre daha fazla zamana ihtiyaç duymaktadır. Bu dezavantajı gidermek ve çözüme daha hızlı ulaşmak için Taillard NEH algoritmasının her bir adımı için özel notasyonlar tanımlamıştır (Taillard, 1990). Bu notasyonlar ve adımlar aşağıda verilmiş ve örnek bir 5-iş 2-makina problemi (Şekil 2.3) üzerinde gösterilmiştir;-

İş sırasındaki i. yere k. işin yerleştirilmesi sonrasında i. işin maksimum akış zamanı M_i'yi belirlemek için:

(1) İlk olarak i.işin j. makinadaki en erken tamamlanma zamanı e_{ij} hesaplanır; ilk işin ilk makinadaki başlama zamanı 0 dır. (Şekil 2.3.(a)),

$e_{0j}=0$, $e_{i0}=0$,

$e_{ij}= \max \{e_{i,j-1},e_{i-1,j}\}+t_{ij}$

$(i=1,..,k-1)\,(j=1,...,m)$

(2) Daha sonra i. işin j. makinada işleme başlamasından, tüm operasyonların tamamlanmasına kadar geçecek kalan süre q_{ij} hesaplanır. (Şekil 2.3. (b))

$q_{kj}=0,\ q_{i,mH}=0,$

$q_{ij}=\max\{q_{i,j+1}, q_{i+1,j}\}+t_{ij}$

$(i=k-1,...,1)\ (j=m,...,1)$

(3) Bu adımda k. işin iş sırasındaki i. pozisyona araya konulduğunda j. makinadaki kısmi en erken tamamlanma zamanı F_{ij} hesaplanır. (Şekil 2.3.(c)).

$F_{i0}=0$

$F_{ij}=\max\{F_{i,j-1}, e_{i-1,j}\}+t_{kj}$

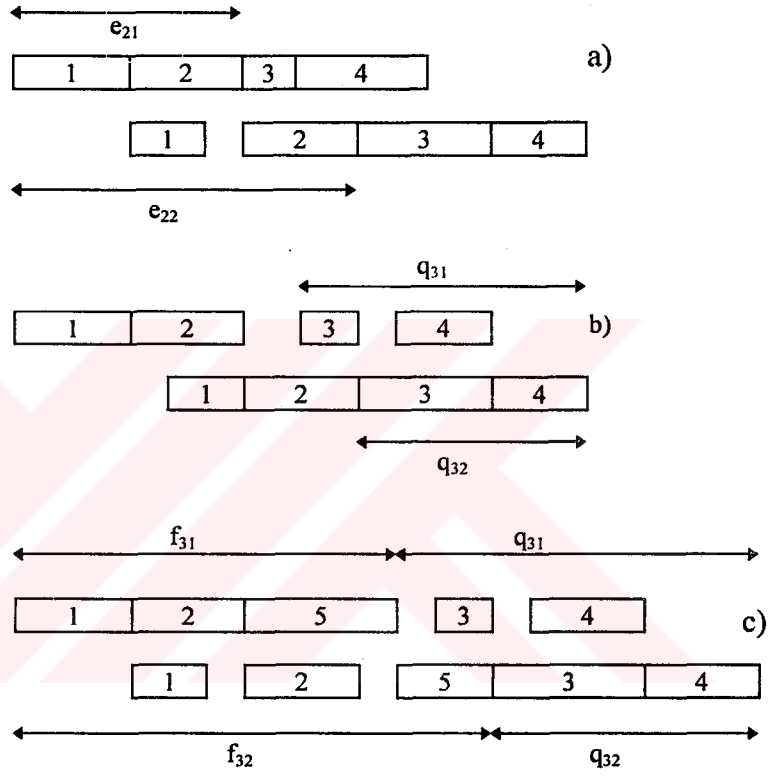
$(i=1,...,k)\ (j=1,...,m).$

(4) Son olarak i. pozisyonuna k işinin ilave edildiğindeki kısmi maksimum akış zamanı M_i hesaplanır.

$M_i=\max_j(F_{ij}+q_{ij})$

$(i=1,...,k)\ (j=1,...,m)$

Tüm bu hesaplamaların yapılması ile algoritma çok daha hızlı bir şekilde işletilir.



Şekil 2.3. : NEH Algoritmasının gösterimi; iş5'in üçüncü yere girişi

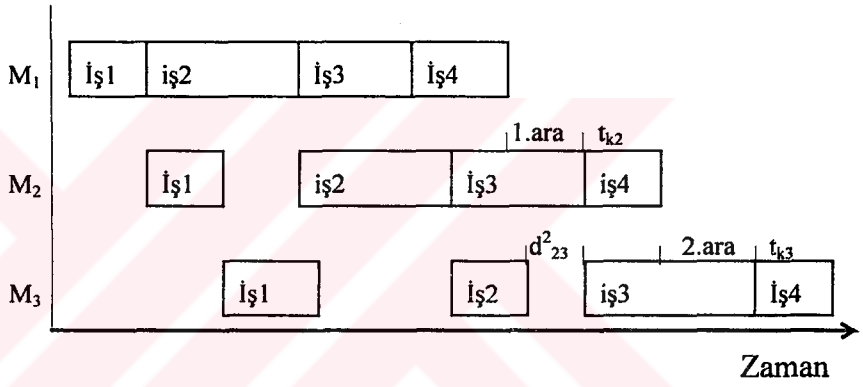
2.3.2.5. HC algoritması

Jhony C. Ho ve Yih.Long Chang (1991) tarafından geliştirilmiştir.Bu algoritma daha önce herhangi bir sezgisel yöntem ile sıralaması belirlenmiş işlerden birbirini takip eden işler arasındaki boşlukları en küçüklemeye çalışır. Şekil 2.4'deki üç makina probleminde de görüldüğü gibi makina 1 (M1) deki işler arasında atıl bir zaman yoktur. Programlanan iş sırasından en son atanan iş k olarak kabul edilir. 1.boşluk ise k işinin M1 deki işleminin bitmesi ile, M2'deki işleminin başlaması arasındaki fark olarak tanımlanır. Aynı şekilde k işinin M_2 makinasında işleminin tamamlanması ile M_3 makinasındaki işleminin başlaması arasında geçen süre ise 2.boşluk olarak tanımlanır. Daha sonra n-iş 3-makina problemi için maksimum akış zamanı şöyle hesaplanır.

$$F_{max} = \sum_{i=1}^{n} t_{i1} + t_{k2} + t_{k3} + 1.\text{boşluk} + 2.\text{boşluk}$$

Örnek 4-iş 3-makina problemi için hesaplamalar gant diyaramında gösterilmiştir. Yukarıdaki formülde ilk üç terim sabit olduğundan maksimum akış zamanını ancak birinci ve ikinci boşlukların toplamlarını minimum yaparak en küçükleyebiliriz. Boşlukların hesaplanması şu şekildedir.

$d_{ij}^{k} = t_{i,k+1} - t_{j,k}$ $\quad i,j=1,2,...,n,\ k=1,2,...,M-1$ ve $i \neq j$



Şekil 2.4. : 4-iş 3-makina problemi maksimum akış zamanı bileşenleri

Eğer sırada i işi j işinden önce ise, d_{ij}^k nin pozitif değeri j'nin i işi bitinceye kadar k+1 makinada en az d_{ij}^k kadar beklemesi gerektiğini gösterir. Örneğin 1. boşluk ve 2. boşluk, sırasıyla d_{34}^1 ve d_{34}^2 Şekil 2.4 de pozitif değerdedirler. Bu da iş 4'ün makina 2 ve 3 de bu boşluklar nedeni ile bekleyeceğini gösterir. Ancak, d_{23}^2 , Şekil 2.4'de negatiftir ve iş2'nin makina3 deki operasyonunun tamamlanmasından sonra iş3'ün makina2 deki operasyonunu tamamlayıp makina3'e gelmesini beklediği atıl süreyi gösterir.

Boşlukların minimize edilmesi için en negatif değerli iş ikililerinin sıranın sonlarına getirilmeye çalışılır. Sıranın sonunda bulunan negatif değerli işler sırada önceden hesaplanan pozitif boşlukların düşmesini sağlar. Sıranın başındaki negatif değerler zaman kaybı ihtimalini arttırır. Aynı şekilde en pozitif değerli iş ikilisini de sıranın başına yerleştirir. Çünkü böylelikle daha sonraki işler arasındaki negatif boşlukların sıkışma ihtimali daha da artar.

Negatif değerlerin etkisini azaltmak için d_{ij}^k ları toplamadan önce yeni bir faktör, faktör(k) tanımlanır. Bu yeni faktörün tanımı şöyledir.

Faktör (k)=((1.0-0.1)/(M-2))*(M-k-1)+0.1 k=1,2,...,M-1.

Burada işlemlerin başladığı ilk sıralardaki makinalara bağlı olan negatif d_{ij}^k ların, pozitif değerli toplam boşlukların küçülmesinde daha çok etkiye sahip olduğundan, bunun sonucu olarak yüksek ağırlıklar (örneğin faktör(k) '1 e daha yakın) küçük k'lara verilmiştir ve düşük ağırlıklar (faktör (k) nın 0'a yakın olması) büyük k değerlilere verilmektedir. Örneğin k=1 için faktör(k) 1'e eşittir, k=M-1 için faktör(k)=0.1 dir. Ortadaki makinalar için faktör(k), 0.1 ile 1.0 arasında lineer enterpolasyon ile belirlenir . Bu faktöre bağlı olarak, bir indirgeme fonksiyonu şöyle tanımlanabilir:

$$\delta_{ij}^k = \begin{cases} Faktör(k) & d_{ij}^k < 0 \text{ olduðunda} \\ 1 & diðer\ durumlarda \end{cases}$$

i,j=1,2,...,n ve k=1,2,...,M-1 için.

Tüm d_{ij}^{k} ları ve indirgeme faktörlerin birleştirldiğinde, tüm boşlukları temsilen düzenlenmiş bir boşluk d_{ij}^{R} şöyle tanımlanır.

$$d_{ij}^{R} = \sum_{k=1}^{M-1} d_{ij}^{k} . \delta_{ij}^{k} \qquad i, j = 1,2,...,n.$$

Düzeltilmiş boşluklara (aralara) bağlı olarak HC algoritası şöyle çalışır.

1. Herhangi bir sezgisel yöntemle ara çözüm olarak adlandırılan bir başlangıç iş sırasını bul.
2. d_{ij}^{R} lerin düzeltilmiş değerlerini hesapla.
3. h(h=1,...,n) ara çözümde işin pozisyonunu ve P_h de bu pozisyondaki işi temsil etsin. Başlangıç olarak baştan sona bir sayaç olan a'ya 1ve sondan başa bir sayaç olan b'ye iş sayısı olan n değerini ata.
4. d_{PaPl}^{R} lerin arasından en büyük değerli olanı bul ve bu değeri X ile tanımla. Burada h, a ve b arasında olmalıdır. X'in bulunduğu h değerini u temsil etsin.
5. d_{PlPb}^{R} lerin arasından en küçük değerli olanı bul ve bu değeri Y ile tanımla. Burada h, a ve b arasında olmalıdır. Y'nin bulunduğu h değerini de v temsil etsin.
6. Eğer (X<0),(Y>0) ve ($|X| <= |Y|$) ise 9. adıma geç.
7. Eğer (X<0),(Y>0) ve ($|X| > |Y|$) ise 10. adıma geç.
8. Eğer($|X| > |Y|$) ise 9. adıma git değilse 10. adıma git.
9. Burada a'yı bir artırarak (a=a+1) a ve u pozisyonlarındaki işlerin yerlerini değiştir ve 11. adıma geç.
10. Burada b'yi bir azalt (b=b-1), b' ve v pozisyonlarındaki işlerin yerlerini değiştir.
11. Performans ölçüsüne göre eğer yeni çözüm problemin önceki çözümden iyi ise çözüm olarak bu yeni çözümü al. Aksi taktirde değiştirilen işleri orjinal çözümdeki yerlerine geri değiştirilen işleri orjinal çözümdeki yerlerine geri getir.
12. Eğer b=a+2 ise aramayı durdur değilse 4. adımdan tekrar aramaya devam et.

Burada bir 5-iş 4-makina problemi üzerinde HC Algoritmasının işleyişini gösterelim. Her bir işin makinalarda harcayacağı işlem zamanları Tablo 2.15 de verilmiştir. İlk olarak denklem 2'den faydalanılarak boşluklar tespit edilir. Bunlar Tablo 2.16-18'de verilmiştir. Faktör(k)'nın hesaplanmasından faydalanılarak indirgeme faktörleri 1.0, 0.55 ve 0.1 olarak bulunur. Değerlendirilmiş d_{ij}^{R} değerleri de ilgili denklem sayesinde bulunur. Örneğin d_{31}^{R} = (11).(1.0)+(-14).(0.55)+ (-19).(0.1)=1.40 Tablo 2.19 bu değerleri göstermektedir.

Birinci adımda CDS sezgisel metodu kullanılarak başlangıç çözümü, iş sırası (4-2-1-5-3) ve maksimum akış zamanı 246 olarak bulunmuştur. Başlangıçta a=1 ve b=5 olmaktadır. 4. adımda, Tablo 2.19'dan en büyük değer (X), d_{4pi}^{R} = -6.40 bulunmuştur, burada h, a(=1) ve b(=5) pozisyonları arasında yer almak üzere u da en büyük değerin bulunduğu pozisyon olan dördüncü değerini almaktadır.

5. Adımda, d_{pl3}^{R} en küçük değerli olarak bulunmuş, h yine a ve b arasında olmak üzere bu en küçük değer -3.80 ve v=4 olmaktadır. X ve Y'nin her ikisi de negatif olduklarında adım 8'e geçilir. Bu adımda ($|X|>|Y|$) olduğu için a bir artırılır a=a+1=2, iş 2(Pa) ve 5 (Pu), yer değiştirilir ve böylelikle yeni bir sıra (4-5-1-2-3) elde edilir. Ancak bu yeni iş sırası daha yüksek bir maksimum akış zamanına (Fmax=248>246) sahip olduğundan bu sıra kabul göörmeyecek işler eski yerlerine geri dönecektir.

Algoritma a=2, b=5 ve b≠a+2 olduğunda devam ettirilir ve başa 4. adıma tekrar dönülür. 4. Adımda, Tablo 2.18 den d_{2pl}^{R} nin en büyük değeri (X), 18.60 ve u=4 olarak bulunur. Burada h, a(=2) ve b(=5) pozisyonları arasında yer alır. Adım 5'te d_{2pl}^{R} nin en küçük değeri -3.80 ve v=4 olarak bulunur. Aynı şekilde h yine a ve b arasında değer almaktadır. X pozitif Y ise negatif olduğundan 8. adıma gideriz. ($|X|>|Y|$) olduğundan a=a+1=3 olarak yeniden belirlenir, iş 1 (Pa) ve 5(Pu) yer değiştirilir ve yeni bir sıra elde edilir. Bu sıranın maksimum akış zamanı 213 (<246) Fmax=213<246 olduğundan elde edilen bu yeni sıra ara çözüm olarak korunur. Daha sonra a=3, b=5 ve b=a+2 olduğundan algoritmanın işleyişi durdurulur ve çözüm iş sırası (4-2-5-1-3) maksimum akış zamanı 213 olarak problem çözülmüş olur.

Tablo 2.15. : 5-iş 4-makina örnek problemi işlem zamanları

İş	M_1	M_2	M_3	M_4
1	31	41	25	30
2	19	55	3	34
3	23	42	27	6
4	13	22	14	13
5	33	5	57	19

Tablo 2.16. : 5-iş 4-makina örnek problemi d_{ij}^1 değerleri

d_{ij}^1 d_{ij}^1	1	2	3	4	5
1	--	22	18	28	8
2	24	--	32	42	22
3	11	23	--	29	9
4	-9	3	-1	--	-11
5	-26	-14	-18	-8	--

Tablo 2.17. : 5-iş 4-makina örnek problemi d_{ij}^2 değerleri

d_{ij}^2	1	2	3	4	5
1	--	-30	-17	3	20
2	-38	--	-39	-19	-2
3	-14	-28	--	5	22
4	-27	-41	-28	--	9
5	16	2	15	35	--

Tablo 2.18. : 5-iş 4-makina örnek problemi d_{ij}^3 değerleri

d_{ij}^3	1	2	3	4	5
1	--	27	3	16	-27
2	9	--	7	20	-23
3	-19	3	--	-8	-51
4	-12	10	-14	--	-44
5	6	16	-8	5	--

Tablo 2.19. : 5-iş 4-makina örnek problemi d_{ij}^R değerleri

d_{ij}^R	1	2	3	4	5
1	--	32.50	11.65	47.00	25.30
2	12.10	--	17.55	51.55	18.60
3	1.40	10.60	--	33.20	25.90
4	-25.05	-9.55	-17.80	--	-6.40
5	-10.60	4.00	-3.80	32.00	--

BÖLÜM 3. GENETİK ALGORİTMALAR (GA)

3.1. Giriş

Bu bölümde ilk olarak genetik algoritmaların bir tanım ve tarihçesi verilmektedir. Daha sonra genetik algoritmaların temelini oluşturan birimler hakkında genel bilgiler ve açıklayıcı örnekler verilmektedir. Genetik algoritmaların genel işleyiş prosedürü, kullanılabileceği alanlar hakkında fikir verici bilgiler ve basit bir uygulama örneği de yine bu bölüm içerisinde bulunabilecek konulardır.

3.2. Genetik Algoritmaların Tanım ve Tarihcesi

Genetik algoritmalar, doğadaki canlıların geçirdiği evrim sürecini örnek alarak matematiksel modeli kurulamayan veya çözüm alanı çok geniş olan problemlerin çözümünde kullanılan tekniklerdir. Genetik algoritmaların evrimden faydalanma ilkesi anne ve babadan meydana gelen bir çocuğun hem annesinin hem de babasının bazı özelliklerini taşıdığı gibi ebeveynlerinden daha üstün özellikleri de taşıyabilmesi varsayımına dayanmaktadır. Makina öğrenmesi üzerinde çalışan John Holland (1975) evrimden ve canlılardaki bu süreçten etkilenerek, canlılardaki bu genetik süreci bilgisayar ortamına aktarmak istemiş ve tek bir mekanik yapının öğrenme yeteneğini geliştirmek yerine, böyle yapılardan oluşan topluluğun çiftleşme, çoğalma ve değişim vb. gibi genetik süreçlerden geçerek başarılı yeni bireyler oluşturabildiğini göstermiştir. Çalışmalarının sonuçlarının 1975'te yayınlanmasından sonra geliştirdiği yöntemin adı Genetik Algoritmalar olarak yerleşmiştir. Bu teorik aşamadan sonra 1985 yılında Holland'ın doktora öğrencisi olan David E. Goldberg adlı inşaat mühendisi tezini, "Gaz boru hatlarının genetik algoritmalar kullanarak denetlenmesi" konusunda yaparak GA'nın uygulanması yönünde ilk adımı atmıştır. Bu ilk uygulamadan sonra 1989'da içerisinde 83 endüstriyel uygulaması bulunan ve GA sahasının klasiği sayılacak bir kitap yayınlayarak (Goldberg, 1989) genetik algoritmaların dünyanın her yerinde farklı sahalarda kullanılabileceğini göstermiştir.

Genetik algoritmalarda her bir çözüm, birey veya kromozom adı verilen dizinler ile gösterilir. Biyolojiyi andırır şekilde, bir kromozom genel olarak 0 ve 1 lerden oluşan dizinler halinde gösterilir. Biyolojik kromozom üzerinde, belirli genlerin belirli karakteristik özellikleri taşıması gibi genetik algoritmadaki kromozomların belirli kısımlarının problemin belirli özellikleri ile çözümünü içerdiği kabul edilir.

Son yıllarda üretim planlama, tasarım, elektronik ve finansman gibi farklı ve çok geniş sahaları kapsayan konuların alt birimlerinde yapılan gerek teorik gerekse uygulamalı genetik algoritma çalışmaları hızla artmıştır. GA'ya olan bu ilgi her geçen gün artmaktadır.

3.3. Temel Kavramlar

Bu bölümde genetik algoritmaların daha iyi anlamak için bazı temel kavramlar tartışılacaktır.

3.3.1. Gen;

Kendi başına anlamlı bir genetik bilgi taşıyan en küçük genetik birimdir. Kısmi bilgi taşıyan bu küçük yapıların bir araya gelmesiyle, konuyla ilgili tüm bilgileri içeren kromozomlar meydana gelir.

Programlama açısından genlerin tanımlanması programcının olayı tanımlamasına bağlıdır. Bir gen A,B gibi bir karekter olabileceği gibi 0 veya 1 ile ifade edilen bir bit veyahut bit dizisi olabilir. Örneğin bir cismin yalnızca x koordinatındaki yerini gösteren bir gen, 101 bit dizisi şeklinde olabilir. Bu cisme ait hız, ağırlık vb. gibi üzerinde durulan diğer noktalarda benzer şekilde ifade edilebilir.

3.3.2. Kromozom;

Bir yada daha fazla genin biraraya gelmesiyle oluşan ve üzerinde durulan probleme ait tüm bilgileri içeren genetik yapılardır. Bir grup kromozom bir araya gelerek bir kütle (popülasyon) oluştururlar. Yani kromozomlar popülasyondaki birey veya üyelere karşılık gelirler. Üzerinde durulan problem açısından ise kromozomlar, geçerli alternatif aday çözümler anlamına gelir.

Örneğin bir kromozom ele alınan bir tasarım probleminde koordinat, açı, boyut gibi değişkenlerden meydana gelen bir bütün olabilir. Aynı kromozom bir üretim planlama probleminde miktar, işlem rotası, zaman gibi değişkenleri içerebir. Basit olarak 100 011 101 bit dizisi 4x3x5 brim boyutlarında tasarlanan bir dikdörtgen kutunun boyutları olabilmektedir.

Kromozomlar, genetik algoritma yaklaşımının üzerinde uygulandığı en temel birim olduğundan olayın bilgisayar ortamında çok iyi ifade edilmesi gereklidir. Kromozomun hangi kısmına ne anlam yükleneceği ve ne tür bir bilgi gösterimi kullanılacağı kullanıcının olaya bakışına ve probleme göre değişik olacaktır.

3.3.3. Popülasyon (Kütle);

Popülasyon, kromozomlar veya bireyler topluluğu olarak tanımlanabilir. Popülasyon aynı zamanda üzerinde durulan problem için geçerli alternatif çözümler kümesidir. Aynı anda bir popülasyonda bulunan birey sayısı sabit olup probleme bağlı olup kullanıcı tarafından belirlenir. Gerçek hayatta da olduğu gibi GA'nın çalışması esnasında popülasyonun bir kısım bireyleri yok olmakta ve yerlerini yenilere bırakmaktadır. İleride anlatılacak genetik operatörler sayesinde sağlanan bu sürekli yenilenme sayesinde yeni çözümlere ulaşmakta ve probleme daha uygun çözümler aranmaktadır.

3.3.4. Uygunluk fonksiyonu;

Kromozomların problemin çözümünde gösterdiği performansı belirleyen ve problemden probleme değişen bir değerlendirme kriteridir. Kromozomun problemin çözümüne uygunluğunu gösteren başarı ölçüsü olarak da düşünülebilir. Hangi kromozomların bir sonraki nesilde de hayat sürdürebileceğini belirlemede ve yeni kromozomları oluşturacak eşlerin oluşturulmasında kromozomların uygunluk fonksiyonu değerleri ağırlıklı olarak gözönünde bulundurulur. Aynı şekilde popülasyonda yeni bireylere yer açmak amacı ile popülasyondan eski bireyleri çıkarma işleminde de uygunluk değeri etkin rol oynar.

Uygunluk fonksiyonu, problem için en uygun çözümü belirleme kriteri olduğundan üzerinde durulan konuyla ilgili kâr veya verimliliği maksimum yapacak, maliyeti veya kaybı minimum yapacak değişkenlerin ölçülmesini sağlayacak bir fonksiyon olmalıdır. Bu fonksiyonun belirlenmesi için problemin iyi tahlil edilerek objektif bir değerlendirme kriteri seçilmelidir.

3.3.5. Genetik operatörler

Genetik operatörler genetik algoritma yaklaşımının problemler üzerinde uygulanmasını sağlayan temel işlemcilerdir. Problemde çözüme ulaşmayı sağlayan bu operatörlerdir. Genetik algoritmaların en temel operatörleri kromozomlar arasında bilgi alışverişini sağlayan "çaprazlama" ve küçük değişimi sağlayan "mutasyon" dur. Genetik algoritmaya bu genetik özelliği veren de bu operatörlerdir. Kromzomlardan bazılarının bir sonraki evrime de aynen kalmalarını sağlayan "kopyalama" operatörü, çaprazlamaya tabi tutulacak kromozomların seçilmesinde "eşleme" operatörü, popülasyondan çıkacak veya yeni popülasyona girecek kromozomları belirleyen "seçme" operatörü ve ilk olarak başlangıç popülasyonunu oluşturan "üretme" operatörü gibi yardımcı operatörlerden faydalanılarak problemin çözümü kolaylaştırılabilir. Genetik operatörlerin en büyük özelliği problemden probleme farklı şekilde tanımlanabilmeleridir.

3.3.5.1. Çaprazlama operatörü;

İki kromozomun bir araya gelerek karşılıklı bilgi değişimi ile yeni kromozomların oluşmasını sağlayan işlemcidir. Bilgi değişimini sağlayacak kromozomların seçiminde uygunluk değerleri gözönünde bulundurulur. Çaprazlanarak bilgi alışverişinde bulunacak kromozomların seçiminden önce, başlangıçta kromozomların çaprazlamaya tutulma olasılığını belirten sabit bir çaprazlama oranı tanımlanır. Bu çaprazlama oranı kromozomların çaprazlamaya tabi tutulma olasılığını verir. Çaprazlamada diğer önemli bir husus ta çaprazlama stratejisidir. Bu strateji ile eş kromozom seçiminde hangi kriterler gözönünde bulundurulacağı belirlenir. Mesela çaprazlanacak ilk kromozom en yüksek uygunluk değerli kromozomdan başlanarak seçilirken ikincisi de diğerleri arasından rastgele seçilebilir. Kromozomlarda bilgi değişimi yapılarındaki seçilen bazı kısımların değiştirilmesiyle olur. Çaprazlama işlemi çeşitli şekillerde olabilir.

Pozisyona dayalı çaprazlamada, çaprazlanacak iki eş kromozom üzerinde bir grup çaprazlama noktası (bir veya daha fazla nokta) rastgele seçilir. İkinci kromozomun bu pozisyonlarındaki işler birinci kromozomun ilgili pozisyonlarına konur ve daha sonra boş kalan pozisyonlar yeni kromozomda olmayan birinci kromozom elemanları sırası ile alınarak doldurulur, böylelikle bir yeni kromozom meydana gelir. Aynı işlem diğer kromozom için de yapılarak yeni bir kromozom daha elde edilir . *Sıraya dayalı çaprazlamada* ise bir grup nokta rastgele seçilir. Birinci kromozomun seçilen noktalara karşılık gelen karekterleri aynen yerlerini korur. İkinci kromozomun seçilen noktalara ait karakterleri birinci kromozomun aynı noktalarındaki karakterlerinin önüne getirilir. Geriye kalan boş pozisyonlara; ikinci kromozomdan aktarılan yeni karakterler de gözönünde bulundurularak ilk kromozomun kullanılmayan karekterleri tarafından sıra ile (soldan sağa) yerleştirilerek yeni bir kromozom elde edilir. İkinci yeni kromozom da aynı şekilde elde edilir. Bu tür çaprazlama kromozomu oluşturan karekterlerin sayı ve sıralarının önem taşıdığı durumlarda kullanılır. Bu çaprazlama işlemine ait birer çaprazlama örneği aşağıda verilmektedir;-

Çaprazlamadan

	Önce	Sonra
Pozisyona göre çaprazlama	A B C D E F G | | G F E D C B A	====G A C D E B F =====A G E D C F B
Sıraya göre çaprazlama	A B C D E F G | | G F E D C B A	==== A G C D E F B ==== G A E D C B F

Çaprazlamada önemli olan bir diğer faktör de çaprazlama noktası sayısıdır. Çaprazlama bir veya daha fazla noktadan olabilir. Basit ve pratik olması bakımından tek noktalı çaprazlama yaygın olarak kullanılmaktadır. Çaprazlamada bilgi değişimi esas olduğundan problemin yapısına göre iki ve daha çok noktalı çaprazlama uygulamaları tercih edilebilmektedir. Diğer geliştirilmiş çaprazlama operatörleri hakkında geniş bilgi Goldberg (1989) da bulunabilir. En klasik tek noktalı çaprazlama birinci ve ikinci kromozom üzerinde ortak belirlenen rastgele nokta temel alınarak birinci kromozomun bu noktadan önceki kısmı ile ikinci kromozomun bu noktadan sonraki kısmı birleştirilerek yeni bir kromozom elde edilir. İkinci komozom için de kromozomların diğer kısımları birleştirilir. Çift noktalı çaprazlama ise kromozomlar üzerinde rastgele belirlenen iki nokta esas alınarak kromozomların bu noktalar arsında kalan kısımlarının karşılıklı değiştirildiği çaprazlamadır. Tek noktalı ve çift noktalı çaprazlamaya ait birer örnek aşağıda verilmektedir;

Tek noktalı çaprazlama

Kromozom1	***1 0 1 1 0 1 0 0***	Yeni Kromozom1	***1 0 1 1*** 0 1 1 0
	|		|
Kromozom2	1 1 0 0 0 1 1 0	Yeni Kromozom2	1 1 0 0 ***0 1 0 0***

Çift noktalı çaprazlama

Kromozom1	*1 0 1 1 0 1 0 0*	Yeni Kromozom1	*1 0 1* 0 0 *1 0 0*
Kromozom2	1 1 0 0 0 1 1 0	Yeni Kromozom2	1 1 0 *1 0* 1 1 0

3.3.5.2. Mutasyon operatörü

Kromozomların genleri veya genleri oluşturan küçük birimleri üzerinde değişiklik yapılmasını sağlayan işlemcidir. Mutasyona uğratılacak kromozomun seçiminde, kromozomun mutasyona uğrama ihtimalini gösteren ve başlangıçta sabit olarak tanımlanan bir mutasyon oranı söz konusudur. Genetik algoritmalarda mutasyona tabi tutulacak kromozomların belirlenmesinde bazılarının istisna tutulması veya özellikle mutasyona uğratılması gibi özel stratejiler tanımlanabilir. Mutasyon işlemi kromozom üzerinde seçilen alel (gen'i oluşturan alt birim) üzerinde yapılacak küçük bir değişiklikle (0'ın 1 yapılması veya tersi gibi) gerçekleşir. Gösterim olarak buna uymayan bir yapıda da yine rastgele seçilecek iki genin yerleri vaya sırası değiştirilmek suretiyle mutasyon gerçekleştirilir. Genetik algoritmalarda mutasyonun sağladığı avantaj problemin çözüm alanını araştırmada yön değişikliklerini sağlayarak araştırmanın kısır döngüye girmesini önlemektir. Çeşitli mutasyon operatörleri vardır. *Pozisyona bağlı mutasyonda* kromozomların rastgele seçilen karakterlerinin yer değiştirilerek gerçekleştirilir. *Sıraya göre mutasyon* ise kromozomun rastgele seçilen karakterlerinden ikincisinin, birincinin önüne getirilmesiyle olur. Kromozomun gösterimine göre sıranın ve karakter sayısının sınırlı olmadığı bir ikili sistem kromozom gösteriminde de rastgele seçilen bir karekterin karşıt (0'ın 1 gibi) değeriyle değiştirilmesiyle olur. Yukarıda bahsedilen çaprazlama türlerine ait birer örnek aşağıda verilmektedir;-

	Mutasyondan Önce	Sonra
Pozisyona göre mutasyon	*A* B C D E **F**	**F** B C D E *A*
Sıraya göre mutasyon	*A* B C D E **F**	**F** *A* B C D E
Kromozom	1 1 0 1 0 1 ***1*** 0	1 1 0 1 0 1 ***0*** 0

3.3.6. Evrim;

Evrim, genetik bilgi taşıyan kromozomlar üzerinde genetik işlemlerin uygulanması sürecidir. Diğer bir deyişle oluşturulan bir topluluğun (popülasyon) kopyalama, çaprazlama ve mutasyon vb gibi işlemlere tabi tutularak yeni topluluğun oluşumuna kadar geçen evredir. Popülasyon üzerinde tüm genetik operatörlerin uygulandığı bir iterasyondur. Bu evrim sürecine, başlangıçta belirlenen bir durdurma kriteri gerçekleşinceye veya uygun çözüm bulununcaya kadar devam edilir.

3.4. Genetik Algoritmaların İşleyişi

Genetik algoritmalarda, öncelikle rastgele veya bilinen "*olurlu*" çözümleri içeren ve uygunluk değerleri hesaplanmış **başlangıç kümesi** (popülasyon) oluşturulur. Popülasyondaki kromozom sayısı sabit olarak alınır. Bu sayının çok küçük olması işlem kolaylığı sağlamakla beraber, alternatif çözüm çeşitliliğini azaltacağından tercih edilmez. Kromozomların fazla olması işlem hacmini arttırarak iterasyonların uzamasına neden olacağından tercih edilmez. Genel olarak popülasyonun büyüklüğü 100 olarak alınmaktadır. Bunun üzerine çıkıldığında iterasyonlar yavaşlamaktadır. Bu başlangıç (başlangıç çözümleri) daha sonra, probleme bağlı olarak belirlenen

temel genetik operatörler yardımıyla evrimden geçirilerek yeni çözümler oluşturulur. Yeni bireyleri oluşturma işleminde kullanılacak kromozomların seçiminde öncelik genellikle uygunluk değeri yüksek olanlara verilir. Bir veya birden fazla rastgele çaprazlama noktası seçilip, bu noktalara göre çaprazlama operatörü uygulanarak bilgi değişimi sağlanır ve yeni kromozomlar oluşturulur. Çaprazlama operatöründen sonra veya çaprazlama esnasında mutasyon operatörü uygulanır. Mutasyon operatörü tek bir kromozom üzerinde işlem yapar. Kromozomun herhangi bir elemanı, rastgele seçilir ve onun yerine o elemanın alabileceği başka bir değer ile değiştirilir. Bu operatörler uygulandıktan sonra mevcut popülasyon bireylerinden bazıları yeni bireylere yer açmak amacı ile popülasyondan çıkartılır. Uygunluk değeri daha yüksek bireyler ise popülasyona eklenir, düşük olanlar ise çıkartılır. Optimal çözüme yaklaşmak amacı ile bu evrimden geçirme operasyonu daha uygun çözüm bulunamayıncaya kadar devam eder. Bu işlemler Şekil 3.1 de gösterilmiştir.

Oluşturulan popülasyonun uygunluk fonksiyonuna göre değerlendirilip uygunluk değerleri hesaplanır. Aradığımız sonucu veren kromozom olup olmadığı kontrol edilir. Kabul edilebilir sonuca ulaşılmamışsa uygunluk değeri gözönünde bulundurularak yeni kromozomlar oluşturmak üzere bazı kromozomlar eşlenir. Eşleştirilen bireyler bilgi alışverişinde bulunmak üzere daha önceden belirlenen çaprazlama stratejisi ve çaprazlama oranına göre çaprazlamaya tabi tutulur. Çaprazlama esnasında veya sonrasında bazı bireyler yine önceden belirlenen mutasyon oranı ve stratejisi çerçevesinde mutasyona uğratılır. Bir önceki popülasyondan direkt aktarılan kromozomlar ve çaprazlama sonucu üretilen yeni kromozomlara yer açmak üzere eski kromozomlardan en düşük uygunluk değerli olanlar veya rastgele seçilenler silinir. Daha sonra oluşan yeni popülasyon uygunluk fonksiyonuna göre değerlendirilir. Belirlenen hedefe ulaşılıncaya kadar bu evrim devam ettirilir.

Şekil 3.1: Genetik algoritmaların genel işleyişi

BÖLÜM 4. GENETİK OPERATÖRLERE DAYALI İŞ SIRALAMA (PERMUTASYON TİPİ ÇIZELGELEME)

4.1.Giriş

Genetik Algoritmalar, alternatif çözümlerin çok fazla olduğu ve bu alternatiflerin birer birer denenerek en iyi çözümün bulunmasının oldukça fazla zaman aldığı (NP complete) problemlerde, klasik çözüm algoritmalarına göre daha iyi sonuçlar verebilmektedir. Parça tasarımı, iş çizelgeleme, gezgin satıcı vb gibi 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 bunlardan birisidir. O halde genetik algoritmalar iş sıralamasında kullanılabilirler. Aşağıda bu problemin çözümü için bir yaklaşım verilmektedir.

4.2. Problemin Tanımlanması (Gösterimi)

Genetik algoritmaların iş sıralama problemine uygulanmasında oluşturulan her bir kromozom, işlerin oluşturduğu alternatif bir iş sırasını, her bir gen de bir işi göstermektedir. Populasyon ise, alternatif iş sıralarını temsil eden kromozomlar topluluğudur. Populasyonun hacmi, içerdiği kromozom sayısı ile tanımlıdır. Örnek kromozomlar şu şekilde olabilir:

Kromozom 1 1 4 5 3 2

Kromozom 2 2 5 1 4 3

Çizelgeleme problemlerinin genetik algoritmalar ile çözümü için problemin gösteriminde, yani kromozomların kodlanmasında, nümerik gösterim kullanılmıştır. Mesela bir kromozomdaki 3 rakamı 3. işi göstermektedir. Bir kromozomda her iş yalnızca 1 defa bulunmalıdır. İşlerin çaprazlama sonunda bu koşula uymalarını sağlamak için özel çaprazlama operatörleri kullanılmıştır.

4.3. Uygunluk Fonksiyonu

Bulunan kromozomların yani iş sıralarının değerlendirilmesinde literatürde farklı kriterler (uygunluk fonksiyonları) kullanılmaktadır. Çizelgeleme problemleri; çözümü sonucunda ne olması gerektiği bilinmeyen problemlerdendir. Bu nedenle bulunan sonuçlar, ya daha önceki bulunan sonuçlara göre gelişme ile ya da atölye verimlilikleriyle kontrol edilerek değerlendirilir. Genel olarak iş sıralama problemlerinde; alternatif iş sırasını veren çözümler içerisinden toplam tamamlanma zamanı en küçük olan veya atölye kullanım oranı en yüksek olan çözüm en iyi olarak kabul edilir. Bu çalışmada da uygunlık fonksiyonunu belirlemede işlerin hazırlık zamanlarının olmayışı, permutasyon tipi çizelgeleme için geliştirilen algoritmaların üzerinde durdukları performans kriterleri gözönünde bulundurulmuştur. Uygunluk fonksiyonu olarak; maksimum tamamlanma zamanı (C_{max}), ortalama iş akış zamanı ($\overline{C}$) ve ortalama pozitif gecikme ($\overline{T}$) alınmış ve enküçüklenmeye çalışılmıştır.

4.4. Çaprazlama Operatörü

Çaprazlama operatörünün uygulanmasından önce çaprazlamaya girecek işlerin seçilmesi gerekir. Çaprazlamaya tabi tutulacak kromozomlar, bir kromozomun çaprazlamaya girebilme ihtimalini belirten ve önceden belirlenen çaprazlama oranı dahilinde seçilir. Çaprazlanacak kromozomların belirlenmesinde önemli olan diğer bir faktör de çaprazlama oranına bağlı olarak tanımlanan popülasyondan kromozom seçme stratejisidir. Çaprazlanacak eşler, populasyon içerisindeki uygunluk değerlerine göre seçilmiştir. Kromozomlar performans kriterine (amaç) göre uygunluk değerleri hesaplanmış ve en iyi sonucu veren kromozomdan başlayarak en kötü çözümü verene doğru sıralanmıştır. Bu kromozomlar arasından çaprazlanacak ilk eş en iyi çözümden başlayarak seçilir ikinci eş ise sıralamada ilk eş'ten sonraki kromozomlar arasından rastgele seçilir. Bunun nedeni en iyi sonucu veren kromozomların bir sonraki çözümlerde kullanılmasını sağlamaktır. Eşleştirilecek toplam kromozom sayısı (η) şöyle belirlenmiştir:-

η = populasyon büyüküğü * çaprazlama oranı

Kullanılan populasyon büyüklüğü 30, çaprazlama oranı ise 0.6 olduğundan her seferinde 18 kromozom çaprazlama işlemine sokulmuştur. Eşlenecek kromozomlar bulunduktan sonra çaprazlama operatörü belirlenir. Literatürde, aynı işin bir kromozomda birden fazla tekrarlanmasını önlemek amacıyla sıralama problemleri için geliştirilmiş özel çaprazlama operatörleri tanımlanmıştır. Bunlardan bazıları; OX (order crossover)(Davis 1985), CX (cycle crossover) (Fox ve arkadaşları 1991), ER (edge recombination) (Lawrence 1991, Fox 1991), PMX (partially mapped crossover) (Uckun ve arkadaşları 1993) ve LOX (Linear Order Crossover) (Lee ve Choi 1995)çaprazlama operatörleridir. Bu çalışmada diğer operatörlere göre daha iyi sonuç veren LOX operatörü (Colin, 1995) kullanılmıştır. Bu operatörde rastgele seçilen iki çaprazlama noktası arasında kalan işlerin numaraları diğer eş kromozomda hangi pozisyona denk düşüyorsa o pozisyon **H** ile gösterilir. Çaprazlama ile bilgi değişiminden sonra iş tekrarının olmaması için kromozomlardaki **H** ler çaprazlama noktaları arasına kaydırılır. İş tekrarı ihtimali bu şekilde ortadan kaldırılınca, kromozomların çaprazlama noktaları arasındaki başlangıç bilgileri karşılıklı değiştirilir. Bu çaprazlama işlemi şu örnek ile açıklanabilir:

Kromozom 1 1 2 3 4 5 6 7 → 1 **H** 3 4 5 **H H** → 1 3 **H H H** 4 5

Kromozom 2 3 4 6 2 7 1 5 → **H H** 6 2 7 1 **H** → 6 2 **H H H** 7 1

Yeni kromozom 1 1 3 6 2 7 4 5

Yeni kromozom 2 6 2 3 4 5 7 1

4.5. Mutasyon Operatörü

Mutasyon operatörü; kromozomların mutasyona uğratılma olasılığını gösteren belirli bir mutasyon oranı dahilinde seçilen kromozomların genleri üzerinde değişiklik yapmaktadır. Sıralama problemlerinde bu değişiklik, rastgele seçilen iki iş birbirleri ile yer değiştirilerek veya seçilen ikinci iş birinci işin önüne getirilerek yapılabilir. Bu çalışmada mutasyon oranı 0.3 alınmış ve bu olasılıkla, kromozomun temsil ettiği iş sırasından rastgele iki iş seçilerek yerleri değiştirilmiştir. Mutasyon oranının bu şekilde yüksek tutulmasının nedeni problemin çözümüne yaklaşmadaki tıkanmaları önlemektir. Mutasyon operatörü çaprazlama işlemi tamamlandıktan sonra uygulanmıştır. Mutasyon oranı yüksek tutulduğu

için bulunan en iyi kromozomun mutasyon ile değişime uğrayarak kaybedilmemesi için sıralamadaki ilk iki kromozom mutasyona uğratılmamıştır. Diğer her bir kromozomun mutasyona uğrama olasılığı ise 0.3 olmaktadır. Bu operatörün çalışması aşağıdaki örnek ile gösterilmiştir.

Mutasyondan önce 1 ***3*** 2 4 ***6*** 5

Mutasyondan sonra 1 ***6*** 2 4 ***3*** 5

4.6. Yeni Popülasyon Oluşturma Operatörü

Yeni popülasyonun kromozomlarını belirlerken; üretilen yeni kromozomlar önceki popülasyon kromozomları ile birlikte uygunluk değerlerine göre sıralanırlar. Daha sonra sıranın ilk 20 kromozomu ve 30 dan sonraki 10 kromozomu alınarak yeni popülasyon oluşturulmuştur. Buradan amaçlanan ilk başlangıç popülasyonondan son popülasyona kadar her popülasyonda iyi ve kötü kromozomların bulunmasını sağlamaktır. Bunun sağlanmadığı durumda kısa süre sonunda popülasyonun tüm kromozomlarının aynı uygunluk değerine sahip oldukları görülmektedir.

BÖLÜM 5. TARTIŞMA ve SONUÇLAR

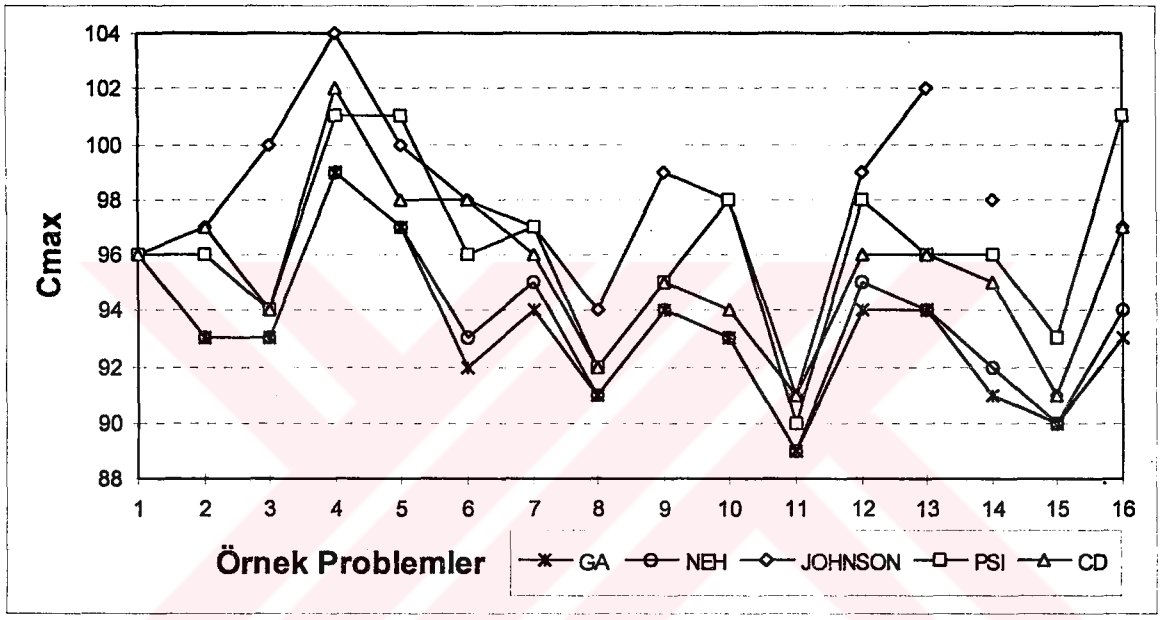
Bu çalışmada, genetik algoritmaların permütasyon tipi atölye iş sıralama problemlerinin çözümünde etkin bir şekilde kullanıldığı gösterilmiş ve klasik sezgisel yöntemlerle karşılaştırması yapılmıştır. Literatürde permütasyon tipi iş sıralamada en iyi sonucu verdiği belirtilen NEH algoritması (Nawaz ve diğerleri 1983, Kondakcı 1990, Taillard 1990) ve diğer sezgisel yöntemlerin önerdiği iş sıralaması sonuçları ile GA'nın ürettiği sonuçlar karşılaştırılmıştır. Bu karşılaştırmalar üç ayrı performans kriteri için de yapılmıştır.

Genetik algoritmalar açısından problemin çözümünü gerek süre gerekse etkinlik açısından etkileyen popülasyon büyüklüğü, mutasyon oranı ve çaprazlama oranı gibi parametreler ile eşleştirilecek kromozomların belirlenmesi, popülasyondan çıkartılacak eski kromozomların ve popülasyona dahil edilecek yeni kromozomların seçilmesi gibi stratejiler için farklı seçenekler denenmiştir. Bu denemelerden populasyonun büyük olduğunda iterasyonların çok zaman aldığı, küçük olduğunda ise çözüme yaklaşmada yetersiz kaldığı görülmüş ve popülasyon büyüklüğü 30 olarak alınmıştır. Çaprazlama oranı olarakta en iyi sonuçların 0.6 da verdiği tesbit edilmiştir. Popülasyonda sürekli değişmelerin sağlanması için mutasyon oranının yüksek tutulup en iyi çözümün sürekli korunması için mutasyona uğratılmadığı durumlarda daha iyi sonuçlar elde edilmiştir.

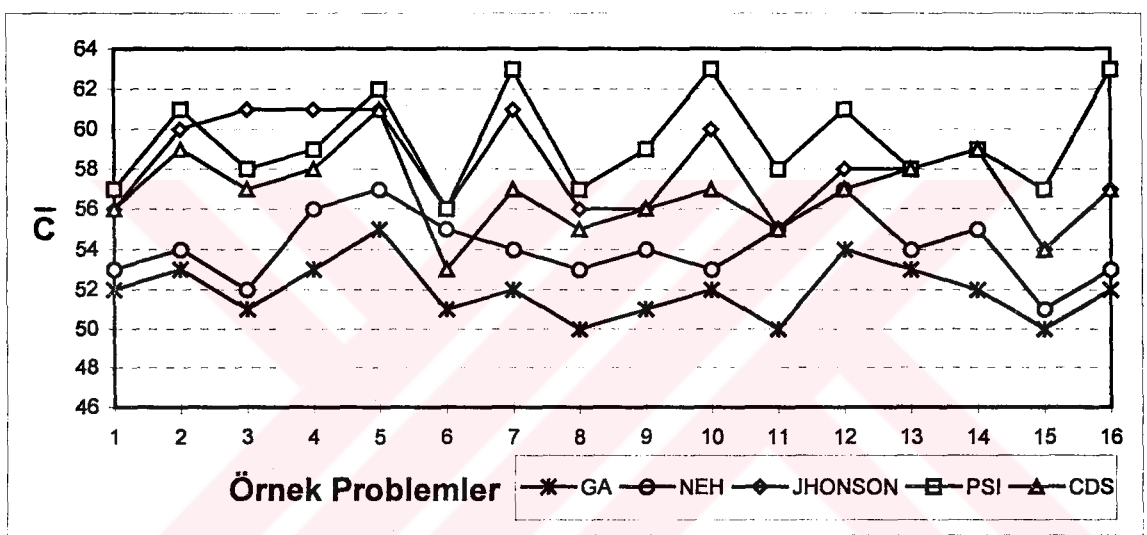
Bu çalışmada; önce örnek olarak 16 farklı 20-iş 4-makina problemi oluşturulmuştur. Aynı problemler, maksimum tamamlanma zamanı (C_{max}), ortalama iş akış zamanı ($\overline{C}$) ve ortalama pozitif gecikme ($\overline{T}$) performans kriterlerine göre sezgisel metodlar ile ayrı ayrı çözülmüş ve sonuçlar GA ile karşılaştırılarak Tablo 5.1'de özet olarak verilmiştir. Karşılaştırılan algoritmalar arasında Dannenbring algoritmasının nedeni gerek PSI ve CDS yöntemi ile ortak özellikleri taşıması ve daha önceden yapılan çalışmalarda NEH algoritmasının daha iyi sonuç vermesidir (Taillard, 1990). Şekil 5.1, Şekil 5.2 ve Şekil 5.3 te ise sonuçlar her bir performans kriteri için ayrı ayrı grafiklerle gösterilmiştir.

Tablo 5.1. : C_{max}, $\overline{C}$, $\overline{T}$ performans kriterlerine göre 20-iş 4-makina problemleri çözümü

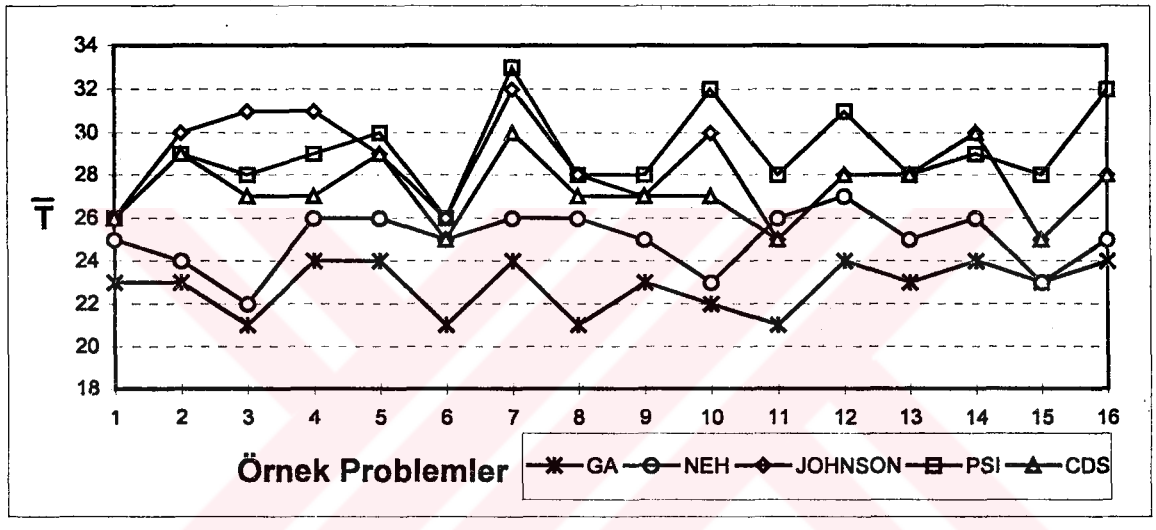
Performans Kriteterleri															
	C_{max}					$\overline{C}$					$\overline{T}$				
Ör.	GA	NEH	JNS	PSI	CDS	GA	NEH	JNS	PSI	CDS	GA	NEH	JNS	PSI	CDS
1	96	96	96	96	96	52	53	56	57	56	23	25	26	26	26
2	93	93	97	96	97	53	54	60	61	59	23	24	30	29	29
3	93	93	100	94	94	51	52	61	58	57	21	22	31	28	27
4	99	99	104	101	102	53	56	61	59	58	24	26	31	29	27
5	97	97	100	101	98	55	57	61	62	61	24	26	29	30	29
6	92	93	98	96	98	51	55	56	56	53	21	25	26	26	25
7	94	95	97	97	96	52	54	61	63	57	24	26	32	33	30
8	91	91	94	92	92	50	53	56	57	55	21	26	28	28	27
9	94	94	99	95	95	51	54	56	59	56	23	25	27	28	27
10	93	93	98	98	94	52	53	60	63	57	22	23	30	32	27
11	89	89	91	90	91	50	55	55	58	55	21	26	25	28	25
12	94	95	99	98	96	54	57	58	61	57	24	27	28	31	28
13	94	94	102	96	96	53	54	58	58	58	23	25	28	28	28
14	91	92	98	96	95	52	55	59	59	59	24	26	30	29	30
15	90	94	91	93	91	50	51	54	57	54	23	23	25	28	25
16	93	94	97	101	97	52	53	57	63	57	24	25	28	32	28



Şekil 5.1: 20/4/P/C_{max} Problemleri Sonuçlarının Grafiksel Gösterimi



Şekil 5.2.: 20/4/P/$\overline{C}$ Problemleri Sonuçlarının Grafiksel Göterimi



Şekil 5.3. : 20/4/P/$\overline{T}$ Problemleri Sonuçlarının Grafiksel Gösterimi

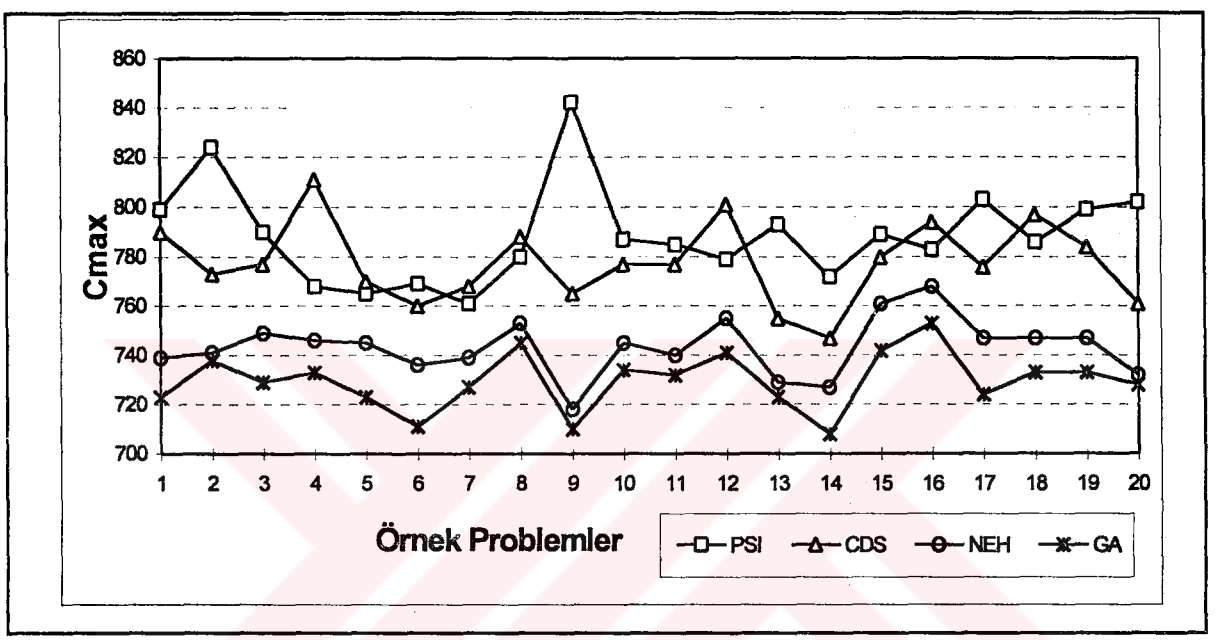
20/4/P/C_{max} problemi çözümünde Şekil 5.1'de görüldüğü gibi iş sayısının ve makina sayısının az olması, işlem sürelerinin düşük (2-6) olması yöntemler arasındaki sayısal farkın düşük olmasına yol açmaktadır. Buna rağmen GA sürekli en düşük sonucu vermekte NEH algoritmasının ise beklendiği gibi GA ya en yakın sonuçları hatta bazen aynı sonuçları verdiği görülmektedir. Diğer performans kriterlerine ($\overline{C}$ ve $\overline{T}$) göre elde edilen sonuçlarda da GA nın diğer sezgisel metodlara göre üstünlüğü Şekil 5.2 ve Şekil 5.3'te açıkça görülmektedir. Benzer bir sonuç Kondakcı'nın (1990) verdiği 9-iş 4-makina problemi için de elde edilmiştir. Kondakcı'nın "veri seti 9" olarak verdiği örnek problemin tamamlanma zamanı, kendi geliştirdiği etkileşimli yöntemle 73, PSI ile 69, CDS ile 69 olarak bulmuştur. Aynı problemin çözümünde geliştirilmiş NEH algoritması ve GA ile 66 değeri elde edilmiştir. Bu ise diğerlerine göre oldukca iyi bir sonuçtur.

Genetik algoritmaların daha iyi sonuçlar verebileceğini gözlemlemek için iş ve makina sayıları artırılarak alternatif iş sırası sayısı 50! olan 50-iş 10-makina problemi ele alınmıştır. 50-iş 10-makina problemi için 20 örnek seti oluşturulmuş ve farklı algoritmaların uygulanması ile elde edilen sonuçlar aynı performans kriterlerine göre maksimum tamamlanma zamanı (C_{max}), ortalama iş akış zamanı ($\overline{C}$) ve ortalama pozitif gecikme ($\overline{T}$) Tablo 5.2'de verilmiştir. Şekil 5.4, Şekil 5.5 ve Şekil 5.6'da ise sonuçlar her bir performans kriteri için ayrı ayrı grafiksel olarak gösterilmiştir. Johnson algoritmasının performansı makina sayısının arttığında düştüğünden ve bu algoritmanın CDS algoritması içerisinde çalıştırıldığından 50-iş 10-makina probleminde değerlendirmeye alınmamıştır.

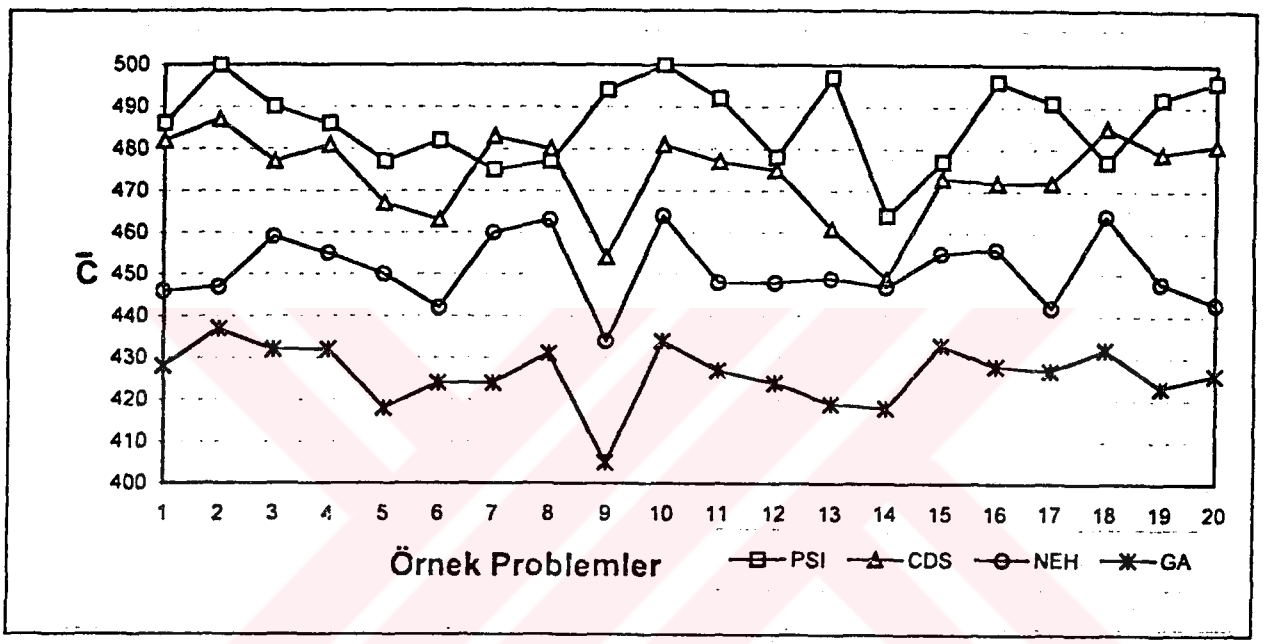
Şekil 5.4, Şekil 5.5 ve Şekil 5.6 dan görüldüğü gibi iş sayısı ve alternatif iş sırasının artmasıyla GA diğer klasik sezgisel yöntemlerin hepsinden daha iyi sonuçlar vermektedir. Her üç performans kriterine göre de en iyi sonuçları vermesi genetik algoritmaların iş sırasını iyileştirmede de çok amaçlı olarak kullanılabileceğini göstermektedir. Problemin zorlaşmasına etki eden makina sayısının artması diğer sezgisel yöntemlerin etkinliğini azaltırken makina sayısından bağımsız çözüm yapan GA, bu üstünlüğünü de kullanarak daha iyi sonuçlar vermektedir. Burada dikkati çeken nokta GA'nın tüm örnek setlerinde NEH den daha iyi sonuç vermesidir. Bu da başarılı sonuçların rastgele olmadığını, büyük ve karmaşık problemlerde GA'nın NEH ten daha iyi sonuç verdiğini göstermektedir.

Tablo 5.2: C_{max}, $\overline{C}$, $\overline{T}$ performans kriterlerine göre 50-iş 10-makina problemleri çözümü

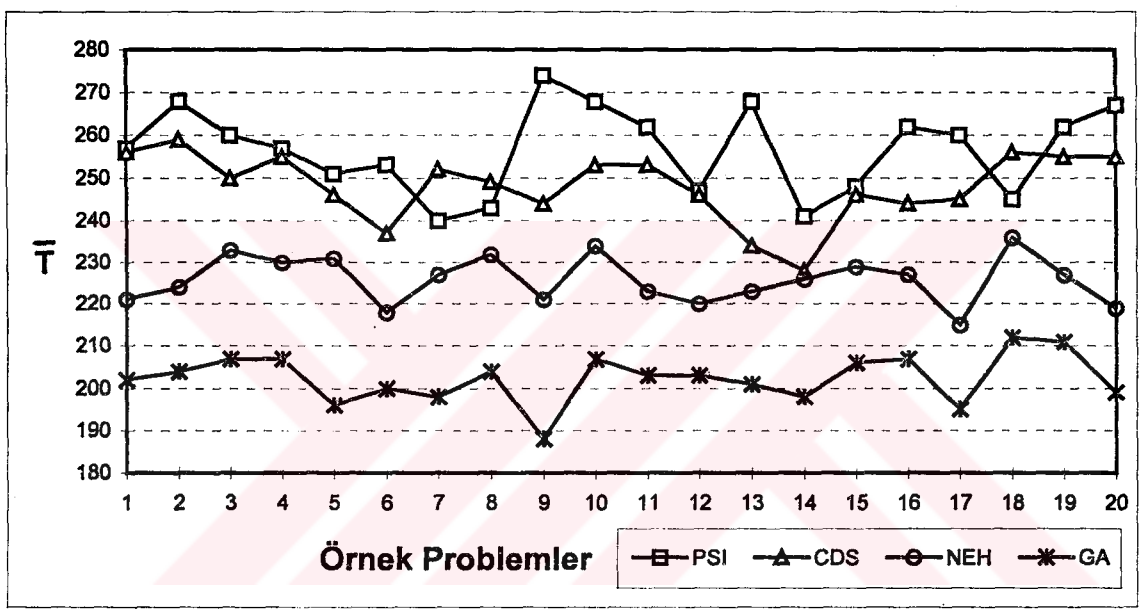
PERFORMANS KRİTERLERİ												
	C_{max}				$\overline{C}$				$\overline{T}$			
ÖR	*PSI*	*CDS*	*NEH*	*GA*	*PSI*	*CDS*	*NEH*	*GA*	*PSI*	*CDS*	*NEH*	*GA*
1	799	790	739	723	486	482	446	428	257	256	221	202
2	824	773	741	737	500	487	447	437	268	259	224	204
3	790	777	749	729	490	477	459	432	260	250	233	207
4	768	811	746	733	484	481	455	432	257	255	230	207
5	765	770	745	723	477	467	450	418	251	246	231	196
6	769	760	736	711	482	463	442	424	253	237	218	200
7	761	768	739	727	475	483	460	424	240	252	227	198
8	780	788	753	745	477	480	463	431	243	249	232	204
9	842	765	718	710	494	454	434	405	274	244	221	188
10	787	777	745	734	500	481	464	434	268	253	234	207
11	785	777	740	732	492	477	448	427	262	253	223	203
12	779	801	755	741	478	475	448	424	247	246	220	203
13	793	755	729	723	497	461	449	419	268	234	223	201
14	772	747	727	708	464	449	447	418	241	228	226	198
15	789	780	761	742	477	473	455	433	248	246	229	206
16	783	794	768	753	496	472	456	428	262	244	227	207
17	803	776	747	724	491	472	442	427	260	245	215	195
18	786	797	747	733	477	485	464	432	245	256	236	212
19	799	784	747	733	492	479	448	423	262	255	227	211
20	802	761	732	728	496	481	443	426	267	255	219	199



Şekil 5.4: 50/10/P/C_{max} Problemleri Sonuçlarının Grafiksel Gösterimi



Şekil 5.5. : 50/10/P/$\overline{C}$ Problemleri Sonuçlarının Grafiksel Gösterimi



Şekil 5.6: 50/10/P/$\overline{T}$ Problemleri Sonuçlarının Grafiksel Gösterimi

GA'nın diğer yöntemlere göre dezavantajı ise, genetik algoritmaların rastgele çözümlerden yola çıkması ve araştırmaya rastgele yönlerde devam etmesidir. Bunun sonucunda, problemin çözümü için daha fazla iterasyon yapmakta ve dolayısı ile daha çok bilgisayar zamanına ihtiyaç duyulmaktadır.

GA'nın başarılı sonuçlar verdiği sadece yukarıda gösterilen algoritmalar ile karşılaştırma sonuçları gösterilmekle birlikte bilinen diğer sıralama algoritmaları ile karşılaştırma yapılması genel bir sonuç çıkartmak için gerekli olacaktır. Yalnız burada NEH algoritmasının çok başarılı bir algoritma olduğu unutulmamalıdır. GA'nın ondan daha iyi sonuçlar verdiği bilindiğine göre, NEH algoritmasının daha başarılı diğer algoritmalardan da GA'nın daha başarılı olduğu söylenebilir. Bu Ho ve Chang (1991) tarafından geliştirilen HC algoritmasının üzerinde yapılan analizler ile açıkça görülmektedir. HC algoritması önce bilinen bir sezgisel algoritma ile çözülen problemin sonucunu iyileştirmektedir. CDS gibi bir algoritmanın sonucunu iyileştirmekle birlikte NEH tarafından üretilen sonucu daha iyi yapamamaktadır. Yani NEH ile aynı sonucu vermektedir. Bu nedenle NEH algoritması ile elde edilen sonuçları HC ile iyileştirmek yerine CDS ile elde edilen sonuçlar iyileştirilmeye çalışılmıştır. GA'nın NEH algoritmasından daha iyi sonuç verdiği gibi HC algoritmasından da iyi sonuçlar verdiği Tablo 5.3 de gösterilmiştir.

Tablo 5.3: C_{max} performans kriterine göre 50-iş 10-makina problemleri çözümü

Yöntem	1	2	3	4	5	6	7	8	9	10
CDS	777	777	801	755	747	780	794	776	797	784
HC	777	770	789	755	747	780	794	776	786	777
GA	740	735	741	723	722	742	753	734	733	733

KAYNAKLAR

Campbell H.G., Dudek R.A., Smith B.L. (1970), "A heuristic algorithm for the n-job, m-machine sequencing problem", **Management Science**, 16, ss:16

Colin, R.R. (1982), "A genetic algorithm for flow shop sequencing", **Computers Ops. Res.,** Vol:22, No:1, ss:5-13

Dannenbring D.G. (1977), "An evaluation of flow-shop sequencing heuristic", **Management Science** 23, ss:11

Davis L., (1985), "Job-Shop Scheduling with Genetic Algorithms", Proc. Intl. Conf. on Genetic Algorithm and their Aplications, Lawrence Erlbaum, Hillsdale, N. J., ss. 136-140

Fox, B.R., McMahon M.B. (1991), "Genetic operatörs for sequencing": In foundations of genetic algorithms, G. J. E. Rawlins (Ed), Morgan Koufmann Publishers, San Mateo, California

French S. (1982), **Sequencing and Scheduling**, John Wiley and Sons, New York

Goldberg, D. E. (1989), **Genetic Algorithms in Search, Optimization & Machine Learning**, Mass.:Addison Wesley, Reading

Ho J. C. ve Chang Y. (1991), "A new heuristicfor the n-job, m-machine flow-shop problem", **European J. of Operations Research** 52, ss: 194-202

Holland, J.H. (1975), **Adaptation in natural and artificial systems** Ann Arbor: The University of Michigan Press

Holland, J.H. (1992), "Genetic algorithms", **Scientific American** , July, ss: 44-50

Kondakcý S., (1990), "Seri iþ akýþlý bir sistemde etkileþimli çizelgeleme ile sezgisel yöntemlerin karþýlaþtýrýlmasý", **Endüstri Müh**, 2/ 9, ss:18-22

Lam S. S., Tang K. W. C. ve Cai X. (1996), "Genetic algorithm with pigeon-hole coding scheme for solving sequencing problems", **Applied Artifical Inteligence,** 10, ss. 239-256

Lawrence, D. (1991), " Hendbook of Genetic Algorithm", Van Nostrand Reinhold, Newyork

Lee C. Y. ve Choi J.Y. (1995), "A genetic algorithm for job sequencing problems with distinct due dates and general early-tardy penalty weights", **Computers Ops. Res.,** 22/8, ss:857-869

Nawaz M., Enscore Jr., E., ve Ham, I. (1983), " A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem", OMEGA, **The Internetional Journal of Management Science** 11/1, ss: 91-95

Öztemel E. (1995), "Endüstri mühendisliðinde yapay zeka uygulamalarý ve grup teknolojisinde bir örnek", **2. otomasyon sempozyumu, 30-31 Mart,** Istanbul, ss:87-98

Palmer D.S. (1965), "Sequencing jobs through a multi stage process in minimum total time a quick method of obtaining a near optimum", **Operations Research 16, ss:1**

Taillard E. (1990), " Some efficent heuristic methods for the flow shop sequencing problem", **European J. of Operations Research,** 47, ss: 65-74

Taþgetiren, M.F. (1988), "Bilgisayar Destekli Atelye Tipi Üretim Çizelgeleme", Yüksek lisans tezi, ÝTÜ Fen Bilimleri Enstitüsü, Ýstanbul

Uckun S., Sugato B., Kawamura K., ve Miyabe Y. (1993), " Managing Genetic Search in Job Shop Scheduling", IEEE Expert October, ss: 15-23

ÖZGEÇMİŞ

Muharrem Düğenci, 1971 yılında Çankırı'nın Ovacık ilçesinde dünyaya geldi. İlk öğrenimini Ovacık'ta, orta öğrenimini Zonguldak'da tamamlayan Muharrem Düğenci, 1989 yılında girdiği İ.T.Ü. Sakarya Müh. Fak. Endüstri Mühendisliği bölümünden 1993 te mezun oldu. 1994 de aynı bölümde Araştırma Görevlisi olarak göreve başladı. Evli ve bir çocuk babası.