Eskişehir Osmangazi Üniversitesi Müh.Mim.Fak.Dergisi C.XXI, S.1, 2008
Eng&Arch.Fac. Eskişehir Osmangazi University, Vol. .XXI, No:1, 2008

Makalenin Geliş Tarihi : 07.09.2007
Makalenin Kabul Tarihi : 13.03.2008

TAM ZAMANLI MONTAJ HATLARINDA ÇOK AMAÇLI KARIŞIK MODEL SIRALAMA İÇİN ROTA BİRLEŞTİRMELİ AÇGÖZLÜ RASSALLAŞTIRILMIŞ UYARLAMALI ARAMA YORDAMI

Şerafettin ALPAY[1]

ÖZET: *Bu çalışma, bir Tam Zamanında Üretim (TZÜ) sisteminde hazırlık sayısının ve malzeme kullanım oranı dengesinin eş zamanlı eniyilemesini amaçlayan karışık-model montaj hattı sıralama problemi için Açgözlü Rassallaştırılmış Uyarlamalı Arama Yordamı (GRASP) sezgiselinin yeni bir uygulamasını sunmaktadır. Birçok test problemi GRASP ile çözülmüş ve sonuçlar, tam sayımlama ve yasaklı arama, genetik algoritmalar, Kohonen self-organizing map sezgisellerinin literatürden alınan sonuçlarıyla kıyaslanmıştır. Test sonuçları, Rota birleştirmeli (Path relinking) GRASP'ın her iki amaç cinsinden en iyiye yakın değerler ürettiğini ve ortalama % başarısızlık ve ortalama yüzdelik performansının da diğer sezgisellerden daha üstün olduğunu göstermektedir. Bununla birlikte GRASP'ın işlemci (CPU) süresi performansı göreceli olarak zayıf çıkmıştır.*

ANAHTAR KELİMELER: *Çok amaçlı karar verme; Karışık-model sıralama; Sezgiseller; Açgözlü Rassallaştırılmış Uyarlamalı Arama Yordamı; Optimizasyon*

A GRASP WITH PATH-RELINKING FOR MIXED-MODEL SEQUENCING WITH MULTIPLE OBJECTIVES ON JIT ASSEMBLY LINES

ABSTRACT: *This research presents a new application of Greedy Randomized Adaptive Search Procedure (GRASP) to address the production sequencing problem for mixed-model assembly line in a just-in-time (JIT) production system when two objectives are present: minimization of setups and optimization of stability of material usage rates. Several test problems are solved via GRASP and the results are compared to the solutions, taken from the literature, obtained via complete enumeration, tabu search, genetic algorithms and Kohonen self-organizing map approaches. Experimental results reveal that the GRASP with Path Relinking provides near-optimal solutions in terms of the two objectives and its "average inferiority %" and "average percentile" performances are superior to that of other heuristics. Results also show that the GRASP performs a little poorly with regard to CPU time.*

KEYWORDS: *Multiple objective decision making; Mixed-model sequencing; Heuristics; Greedy randomized adaptive search procedure; Optimization*

---

[1] *Eskişehir Osmangazi Üniversitesi, Mühendislik Mimarlık Fakültesi*
Endüstri Mühendisliği Bölümü, Bademlik Kampüsü, 26030, ESKİŞEHİR

I. GİRİŞ

Karışık model montaj hatları, ürün karakteristikleri birbirine benzer olan pek çok farklı ürün modelinin montajlandığı üretim hatları tipidir. Bu tip üretim hatları pek çok farklı model için doğabilecek anlık talep değişimlerine yüksek miktarlarda stok bulundurmadan hızlı bir şekilde cevap verebilecek yetenekte olup tam zamanında üretim (TZÜ) sistemlerinin başarılı bir şekilde uygulanması için gereklidir. Bir TZÜ üretim çevresinde karışık model montaj hatlarındaki üretim sırasının belirlenmesi hayati derecede önem taşımaktadır [1]. İyi bir ürün üretim sırası, kabul edilebilir ürün karışımı seviyesini ve aynı zamanda, hazırlık sürelerinin önemli olduğu durumlarda, ihtiyaç duyulan hazırlık sayısının kabul edilebilir miktarını içermelidir. Ürün karışımı seviyesini ölçmek amacıyla Miltenburg [2] malzeme kullanımı dengesi cinsinden "*kullanım oranı*" olarak isimlendirdiği bir ölçüt geliştirmiştir. TZÜ üretim çevresinde, yöneticiler sıklıkla hem hazırlık sayısını hem de kullanım oranını en küçükleyecek üretim sıralarının bulunmasını arzu ederler ki bu da sıralama kararını çok amaçlı bir problem haline getirmektedir.

Geçmiş çalışmalar [3-9] bu iki amacın: hazırlık sayısının ve kullanım oranının enküçüklenmesi, sıklıkla birbiri ile çelişmekte olduğunu göstermektedir. Bu çalışmada, karışık-model üretim sistemlerinde, iki amaçlı bir kombinatoryal sıralama probleminin çözümlerinin araştırılmasıyla ilgilenilmektedir. Çalışmada ayrıca, diğer çalışmaların aksine, farklı modellerin değişimleri arasındaki hazırlık sürelerinin küçük ama önemli olduğu varsayılmaktadır. Bu makalede sunulan araştırmada bu tür çok amaçlı problemler için kullanılan bir yaklaşım olan Etkin Sınır (*efficient frontier)* yaklaşımı kullanılmıştır. Bu yaklaşım, ağırlıklandırılmış birleşik fonksiyonlara ilişkin zorluklarla uğraşmadan, her iki amacı da aynı anda eniyilemek amacıyla kullanılabilmektedir [6]. Çalışmada, etkin sınır, popüler bir sezgisel olan ve daha önce karışık model sıralama problemleri için uygulanmamış, *Açgözlü Rassallaştırılmış Uyarlamalı Arama Yordamı* (GRASP — Greedy Randomized Adaptive Search Procedure) ile oluşturulmuştur ve literatürden alınan değişik test problemlerini çözmek amacıyla kullanılmıştır. Sezgiselin performansı tam sayımlama ile elde edilen en iyi sonuçlarla ve aynı zamanda diğer popüler sezgisellerin; Yasaklı Arama (*Tabu Search)* (YA), Genetik Algoritmalar (*Genetic Algorithms)* (GA), Kohonen Self-Organizing Map (SOM) sonuçlarıyla kıyaslanmıştır.

II. KARIŞIK-MODEL ÇİZELGELEME PROBLEMİ

II.1 Amaç Fonksiyonları

İki amacın matematiksel ifadesinin sunumundan önce aşağıdaki gösterimler tanımlanmıştır.

n: üretilecek farklı tip ürünlerin sayısı

D: tüm ürünlerden üretilecek miktarların toplamı ya da toplam talep

d_i: i. tip ürün için talep, $i=1,2,...,n$

x_{ik}: 1'den k'ya kadar ki aşamalarda i. üründen üretilen toplam miktar. Burada k bir sıradaki pozisyon indeksini ifade etmekte olup, $k=1,2 \ldots, D$'dir.

s_k: eğer hazırlık gerekiyorsa (model dönüşümü varsa) 1, aksi halde 0'dır.

İlk amaç, gereksinim duyulan hazırlık sayısının enküçüklenmesidir. Toplam hazırlık süresinin toplam hazırlık sayısına orantılı olduğu varsayımı altında, bir sıradaki gereksinim duyulan hazırlık sayısı (S) [4]:

$$S = 1 + \sum_{k=2}^{D} s_k \tag{1}$$

eşitliği ile bulunur.

İkinci amaç malzeme kullanım oranlarının eniyilenmesidir. Malzeme kullanım oranı (U) [4]:

$$U = \sum_{k=1}^{D} \sum_{i=1}^{n} (x_{ik} - k\frac{d_i}{D})^2 \tag{2}$$

eşitliği ile hesaplanır.

İkinci amaç için, daha düşük kullanım oranları arzu edilmektedir.

II.2 Etkin Sınır (Efficient Frontier)

Bileşik fonksiyonlar kullanarak amaçların ağırlıklandırılmasının zorluklarından dolayı, bu çalışmada, her bir ihtiyaç duyulan hazırlık sayısı için kullanım oranlarını enküçükleyecek sıraların belirlenmesi amacıyla "*etkin sınır*" yaklaşımı kullanılmaktadır. Bir etkin sınır, sınır üzerinde olmayan diğer tüm noktalara baskın

öznitelikler taşıyan bir eğri üzerindeki noktalar kümesidir. Bu araştırmada, öznitelikler, ilgilenilen bir sıra için, gereksinim duyulan hazırlık sayısı ve kullanım oranıdır. Her bir öznitelik için bir eksen mevcut olup, gereksinim duyulan hazırlık sayısı *x*-ekseninde ve kullanım oranı da *y*-ekseninde gösterilmektedir [5]. Grafikte, her bir gereksinim duyulan hazırlık sayısı için en küçük kullanım oranına sahip olan nokta (ya da sıra) etkin kabul edilmekte ya da tersi bir ifadeyle, ilgilenilen bir hazırlık sayısı için etkin sıra dışındaki diğer tüm noktalar etkin olmayan olarak kabul edilmektedir [6].

Bir örnek olarak, bir üretim sırasında bulunacak 3 farklı ürünün, *A*, *B* ve *C,* verildiği varsayılsın (n=3). Bu ürünlerin talep değerleri de sırasıyla d_1=5, d_2=4, ve d_3=3 olsun. Böylece, toplam talep miktarı D=5+4+3=12 olur. Bir olası üretim sırası, *CCCAAAAABBBB* olup, (1) no'lu denklemden bu sıra için gereksinim duyulan hazırlık sayısı 3 olarak bulunur (S=3). Bu sıranın kullanım oranı da, (2) no'lu denklemden, 59,03 olarak bulunur (U=59,03). Bir başka olası üretim sırası ise *ABCABACBACBA*'dır. Bu sıranın gereksinim duyulan hazırlık sayısı ve kullanım oranı, aynı denklemler yardımıyla, sırasıyla 12 (S=12) ve 3,36 (U=3,36) olarak bulunur. Bu sıralardan, ilki daha az hazırlık sayısı gerektirmekte ancak daha fazla kullanım oranı sağlamakta iken, ikinci sıra daha fazla hazırlık sayısı ile daha az kullanım oranı sağlamaktadır. Bu örnekten de görüldüğü üzere, gereksinim duyulan hazırlık sayısı ile kullanım oranı arasında açık bir ödünleşme (trade-off) mevcuttur.

Sonuç olarak, Etkin Sınır yaklaşımı her bir gereksinim duyulan hazırlık sayısı için ilişkili kullanım oranını en küçükleyecek sıraların bulunması amacıyla karar vericiye bir imkân sağlamaktadır [10]. Şekil 1'de, örnek problem için, farklı hazırlık sayıları için tam sayımlama ile bulunan en küçük kullanım oranlarına ilişkin etkin sınır gösterilmektedir.

Şekil 1. Örnek problem için Etkin Sınır.

II.3 Problem Karmaşıklığı

Her iki amacı arzu edilebilir seviyede sağlayan üretim sıralarının bulunması problemi NP-Zor'dur [11]. *n* Adet ürünün olduğu karışık-model montaj sıralama problemi için toplam olası sıra sayısı [6];

$$Toplam\ Sıra = \frac{(\sum_{i=1}^{n} d_i)!}{\prod_{i=1}^{n} (d_i!)} \tag{3}$$

ifadesiyle bulunmaktadır.

Problem boyutu büyüdüğünde uygun çözümlerin sayısı üstel olarak artmaktadır ─ problem boyutundaki küçük bir büyüme en iyi çözümleri bulmak için gerekli olan hesaplama ihtiyaçlarını büyük miktarda arttırmaktadır ─ Bu nedenle, olası çözüm sayısı geniş olan problemler için doğrusal veya tamsayılı programlama gibi geleneksel çözüm yaklaşımları ile en iyinin bulunması, kabul edilebilir bir süre içinde, genellikle mümkün olamamaktadır. Sonuç olarak, arama sezgiselleri, kabul edilebilir bir süre ve çabayla arzu edilir ya da en iyiye yakın çözümler bulmaya imkân sağlamaktadırlar.

III. ROTA BİRLEŞTİRMELİ AÇGÖZLÜ RASSALLAŞTIRILMIŞ UYARLAMALI ARAMA YORDAMI

GRASP kombinatoryal eniyileme problemleri için kullanılan bir meta-sezgisel olup, her bir tekrarda probleme ilişkin bir uygun çözümün hesaplandığı, arama uzayının tekrarlı rassal örneklenmesine dayalıdır. Her bir GRASP tekrarlaması (iterasyonu) 2 ana aşamadan oluşmaktadır: Rassallaştırılmış bir sezgisel temelindeki *Oluşturma Aşaması* ve ilk aşamada oluşturulan başlangıç çözümü iyileştiren *Yerel Arama Aşaması*'dır. Tekrarlama süreci belirtilen bir tekrar sayısına ulaşıncaya kadar sürdürülür ve o zamana kadar elde edilen en iyi çözüm son çözüm olarak değerlendirilir. GRASP ilk kez Feo ve Bard [12] tarafından 1989 yılında tanıtılmıştır. GARSP ve uygulamaları için daha detaylı bilgi Resende ve Riberio [13] tarafından sunulmaktadır.

Karışık-model sıralama problemi için bu çalışmada önerilen Rota Birleştirmeli (Path Relinking) GRASP algoritmasının temel adımları aşağıda gösterilmektedir.

Rota Birleştirmeli GRASP Yordamının Adımları

0: Girdi veriyi oku.

1: $j \leftarrow 1$, $i \leftarrow 1$

2: i ≤ *EnbTekrar* olduğu sürece, adım 3'e git, aksi durumda adım 9'a git

3: Açgözlü Rassallaştırılmış Oluşturma (*Greedy Randomized Constructor*)

4: Yerel Arama (*Local Search*)

5: Eğer $i=j*TekrarSayısı$ ise Rota Yenileme gerçekleştir ve *j*'yi 1 arttır

6: En iyi çözümü güncelle

7: *i*'yi 1 arttır

8: Adım 2'ye git

9: Mevcut en iyi çözümü göster ve Dur.

GRASP, verilen bir en büyük tekrar sayısına (*EnbTekrar*) erişilinceye kadar tekrar döngüsünü sürdürmektedir. GRASP'ın her bir tekrarında, açgözlü (greedy) rassallaştırılmış (randomized) oluşturma ile (adım 3) bir uygun çözüm oluşturulmakta, geçerli çözümün önceden belirlenmiş komşuluğu dikkate alınarak yerel en iyiye erişilinceye kadar yerel arama gerçekleştirilmekte (adım 4), önceden belirlenmiş sayıdaki tekrar aralıklarında (*TekrarSayısı*) rota yenileme gerçekleştirilmekte (adım 5), ve adım 6 da olası bir çözüm güncellemesi gerçekleştirilmektedir.

III.1 Oluşturma Aşaması

Çözümler, uygunluğu daima sağlayacak şekilde, açgözlü rassallaştırılmış yordam yardımıyla oluşturulmaktadır. Aday işler kümesi, henüz sıralanmamış ve olası bir sırada bulunabilecek tüm işleri kapsamaktadır – burada, başlangıçta, sıralanacak olası işlerin toplam sayısının toplam talebe eşit olduğunu not etmek gerekmektedir. Bir çözüm oluşturmak için ilk iş aday işler içinden rassal olarak seçilir ve bu iş sıraya konur. Bu tek işlik sıra "*geçerli sıra*" olarak kabul edilir. Sıranın sonraki işleri açgözlü yordam yardımıyla

seçilmektedir. Bu yordama göre, aday işlerin her biri için geçerli sıra ile oluşturulabilecek farklı sıralar, uygunluğu da dikkate alarak, Bölüm 3.2'de açıklanan "*ileriye kaydırma*" metodu yardımıyla türetilmektedir. Bu metod rassal başlangıç sıra oluşturmaya kıyasla daha fazla süre gerektirmekte ancak daha iyi başlangıç çözümler oluşturmaya imkân sağlamaktadır. Kısaca metod şu şekilde çalışmaktadır: Geçerli sıranın '*ABC*' olduğunu ve aday iş'in de *D* olduğunu varsayalım. *D* işi ile, *ileriye kaydırma* metodu yardımıyla, oluşturulabilecek alternatif sıralar şunlardır; "*DABC*", *ADBC*", "*ABDC*" ve "*ABCD*" ─ Burada *D* işinin, geçerli sıra dikkate alındığında, soldan sağa (ileriye) doğru sırasıyla kaydırıldığını not etmek gerekmektedir. Bu sıralardan uygun olmayan ─ gerekli hazırlık sayısından daha fazla sayıda hazırlık içeren (henüz tüm aday işler sıralanmamışken gereğinden daha az sayıda hazırlık içeren sıralar uygun olarak kabul edilmektedir) ─ sıralar elenmekte ve her bir uygun sıra için denklem (2) yardımıyla kullanım oranları hesaplanmaktadır. Tüm aday işler için uygun alternatif sıralar türetildikten sonra, bu aday işler ve ilgili uygun sıralarından "*Sınırlandırılmış Aday Listesi (SAL)*" (Restricted Candidate List-RCL) oluşturulmaktadır. Amacımız kullanım oranını enküçüklemek olduğundan, tüm alternatif sıralardan en küçük kullanım oranına sahip sıra ile ilişkili iş aslında sıralanacak bir sonraki en iyi iştir, ancak bu şekilde bir seçim yapısı tam bir açgözlü tip sezgiseli doğurmaktadır. Bunun yerine, çalışmada, en iyi kullanım oranlarına sahip sıralarla ilişkili işlerden bir *SAL* oluşturulmaktadır.

U_{enk} ve U_{enb}'ün açgözlü yordam yardımıyla hesaplanan en küçük ve en büyük kullanım oranları olduğu varsayılsın. *SAL* içinde yer alacak iş sayısı, $\alpha \in [0,1]$ olmak üzere bir α parametresi yardımıyla sınırlandırılabilmektedir: α değerine bağlı olarak, kullanım oranları, $U_{enk} + \alpha (U_{enb} - U_{enk})$ değerine eşit ya da küçük olan sıralara ilişkin tüm aday işler *SAL* içinde yer almaktadır. α Değeri 1'e eşit olduğunda tüm aday işler *SAL*'a dahil edilir ki bu rassallaştırılmış oluşumla sonuçlanmaktadır. Diğer yandan, α değerinin 0 olması halinde ise, *SAL* içinde en küçük kullanım oranlı sıra ile ilişkili iş bulunur. Bu durum da tam açgözlü oluşumla sonuçlanmaktadır. *SAL* oluşturulduktan sonra, sıralanacak sonraki iş, kullanım oranlarının dikkate alınması ile *SAL* içinden rassal olarak

seçilmektedir. Seçim süreci, aynı zamanda, seçilen işle ilişkili sırayı da belirlediğinden, bu sıra geçerli iş sırası olarak kabul edilir ve aday işler kümesi güncellenir.

Bu yordam aday işler kümesinde hiçbir iş kalmayıncaya yani bir uygun çözümün oluşturulmasına kadar devam ettirilmektedir.

Oluşturma Aşamasının Adımları

0: İlk işi rassal olarak seç ve sıraya koy. Bu sırayı geçerli sıra olarak kabul et.

1: Aday işler kümesini güncelle.

2: Aday işler (henüz sıralanmamış olanlar) için olası uygun sıraları *ileriye kaydırma metodu* ile türet.

3: Her bir aday iş için türetilen uygun sıraların kullanım oranlarını hesapla

4: Sınırlandırılmış Aday Listesini (*SAL*) türet

5: *SAL*'den rassal olarak bir iş ve ilişkili sırasını seç. Bu sırayı geçerli sıra olarak kabul et.

6: Eğer bir çözüm tamamlandı ise, Adım 7'ye git, aksi halde (henüz tüm işler sıralanmadı ise) Adım 1'e git

7. Oluşturulan çözümü geri döndür ve Dur.

Oluşum aşaması için α parametresi büyük önem taşımaktadır. Resende ve Riberio [13] α değerinin 0-1 arasında sabit bir değer olarak belirlenmesinin yerine dinamik olarak değiştirilmesinin daha iyi olacağını raporlamaktadırlar. Bu nedenle, α değerleri, bu çalışmada, bir kesikli düzgün olasılık dağılımı yardımıyla seçilmektedir. Olasılık dağılımına göre α'nın alabileceği olası değerler; 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0'dır.

III.2 Yerel Arama Aşaması

Açgözlü rassallaştırılmış oluşturmayla, küçük komşuluklar için bile, sıklıkla en iyi çözümler türetilememektedir. Yerel arama aşaması, oluşturulan başlangıç çözümde genellikle iyileşme sağlamaktadır. Bir yerel arama algoritması komşulukta bulunan daha iyi bir çözümü geçerli çözümle yer değiştirecek şekilde tekrarlamalı olarak çalışmaktadır [13]. Bu çalışmada uygulanan yerel arama algoritması, *Değişken Komşuluk Arama (DKA)* (Variable Neighborhood Search) ve *Değişken Komşuluk*

aZalması (DKZ) (Variable Neighborhood Descent) sezgisellerinin bir varyantı olup Gupta ve Smith [14] tarafından önerilmektedir. Gupta ve Smith, ilgili çalışmalarında, üç farklı arama komşuluğu tanımlayarak bunları *DKA* ve *DKZ* sezgiselleriyle birleştirmişlerdir. Bu arama komşulukları aşağıda tariflenmektedir.

İkili Değişim Komşuluğu (pairwise interchange neighborhood). Bir geçerli sıradaki tüm olası ikili değişimler gerçekleştirilmektedir. Örneğin, geçerli sıranın "*ABC*" olduğu varsayımıyla, sırasıyla A ve B, A ve C,…,C ve A, C ve B ikili değişimleri gerçekleştirilir. Bu değişimler sonucu türetilen sıralardan uygun olmayanlar elimine edilir. Herhangi bir ikili değişim sonucu bir iyileşme sağlanırsa, geçerli sıra güncellenir ve işlemlere baştan tekrar başlanır. Süreç hiçbir ikili değişim sonucu iyileşme sağlanamadığında durdurulur ve geçerli sıra yerel en iyi olarak kabul edilir.

İleriye Kaydırma Komşuluğu (Forward Insertion Neighborhood). Bir geçerli sıradaki her bir iş için olası tüm ileriye doğru kaydırmalar yerine getirilmektedir. Örnek olarak, geçerli sıranın "*ABC*" olduğu varsayılsın. Kaydırmalara en soldaki iş, *A*'dan başlanır ve sırasıyla diğer tüm işler için ileriye doğru tekrarlanır. Böylece, *A*'nın kaydırılması ile, "*BAC*" ve "*BCA*" sıraları, *B*'nin kaydırılması ile de, "*ACB*" sırası elde edilir. Herhangi bir kaydırma sonucu bir iyileşme sağlanırsa, geçerli sıra güncellenir ve işlemlere baştan tekrar başlanır. Süreç hiçbir kaydırma sonucu iyileşme sağlanamadığında durdurulur ve geçerli sıra yerel en iyi olarak kabul edilir.

Geriye Kaydırma Komşuluğu (Backward Insertion Neighborhood). Bir geçerli sıradaki her bir iş için olası tüm geriye doğru kaydırmalar yerine getirilmektedir. Örnek olarak, geçerli sıranın "*ABC*" olduğu varsayılsın. Kaydırmalara en sağdaki iş, *C*'den başlanır ve sırasıyla diğer tüm işler için geriye doğru tekrarlanır. Böylece, *C*'nin kaydırılması ile, "*ACB*" ve "*CAB*" sıraları, *B*'nin kaydırılması ile de, "*BAC*" sırası elde edilir. Herhangi bir kaydırma sonucu bir iyileşme sağlanırsa, geçerli sıra güncellenir ve işlemlere baştan tekrar başlanır. Süreç hiçbir kaydırma sonucu iyileşme sağlanamadığında durdurulur ve geçerli sıra yerel en iyi olarak kabul edilir.

Bu komşuluklar temelinde yerel arama aşağıda tanımlanmaktadır.

Yerel Arama Aşamasının Adımları

0: Geçerli çözümün kullanım oranı değerini *KulORN'a* ata. Bir yerel en iyi elde edilene kadar bu çözüm üzerinde *İkili Değişim Komşuluğu* araması uygula. Yerel en iyiyi geçerli çözüm olarak kabul et.

1: Bir yerel en iyi elde edilene kadar geçerli çözüm üzerinde *Geriye Kaydırma Komşuluğu* araması uygula. Yerel en iyiyi geçerli çözüm olarak kabul et.
2: Bir yerel en iyi elde edilene kadar geçerli çözüm üzerinde *İleriye Kaydırma Komşuluğu* araması uygula. Yerel en iyiyi geçerli çözüm olarak kabul et.
3: Geçerli çözümün kullanım oranı değeri *KulORN*'dan daha iyi ise Adım 0'a git, aksi halde Dur.

III.3 Rota Birleştirme

Rota Birleştirme (PR — Path Relinking), temel GRASP yordamının iyileştirilmesi için geliştirilen bir başka yaklaşım olup genellikle çözüm kalitesini arttırmaktadır [13]. PR, bir *başlangıç çözüm* ve bir *klavuz çözüm*'ü birbirine bağlayan yörüngelerin araştırılmasını içermektedir. Klavuz çözüm, genellikle, kalitesi yüksek bir çözümdür [15]. Hareketler başlangıç çözümden başlayarak klavuz çözüme erişilecek şekilde, herhangi bir komşuluk araması (ikili değişim, ileriye veya geriye kaydırma komşulukları) temelinde gerçekleştirilmektedir. Bu hareketler sırasında uygunluk daima korunmak zorundadır. Süreç, başlangıç çözümden klavuz çözüm elde edilmesine kadar sürdürülmektedir [14].

Bu araştırmada, 50 klavuz çözümden oluşan bir kümeyle çalışılmaktadır. Başlangıçta bu küme boş olduğundan, ilk 50 tekrarlama sonucunda elde edilen 50 çözüm ile küme oluşturulmaktadır. Bir tekrarlamadan sonra elde edilen bir yerel en iyi, kümedeki en kötü çözümden daha iyi ise, yerel en iyi, en kötü çözümle yer değiştirmektedir. Böylece, klavuz çözümler kümesi, daima elde edilen en iyi çözümleri içermektedir. PR, karar verici tarafından belirlenen bir tekrarlama aralığında (bu çalışmada her bir 50 tekrarlamadan sonra) yerine getirilmektedir. Klavuz çözüm küme içinden rassal olarak seçilmektedir. Başlangıç çözüm ise, PR'den hemen önce elde edilen geçerli yerel en iyi çözüm olarak alınmaktadır. Hareketler, ileriye ve geriye doğru kaydırma komşulukları temelinde başlangıç çözümden klavuz çözüm elde edilinceye kadar yerine getirilmektedir. Bu esnada, eldeki en iyi çözümden daha iyi bir çözüm elde edildiğinde eldeki en iyi çözüm güncellenmektedir. İzleyen paragrafta 3 farklı ürün için rota birleştirme örneklenmiştir.

Klavuz çözüm kümesinden rassal olarak seçilen klavuz çözüm "*ABCC"* ve başlangıç çözüm de "*CBAC"* olsun. İleriye kaydırma komşuluğu temelinde başlangıç çözümden hareketle elde edilen sıralar şunlardır.

Başlangıç çözüm "*CBAC"*

C ileriye kaydırılır	*"BCAC"*	(Eldeki en iyiden daha iyi mi kontrol et)
C ileriye kaydırılır	*"BACC"*	(Eldeki en iyiden daha iyi mi kontrol et)
B ileriye kaydırılır	*"ABCC"*	(Klavuz çözüm. Dur.)

Burada uygulanan PR süreci Gupta ve Smith [14] tarafından sunulan PR sürecine oldukça benzerdir.

IV. ÖNERİLEN YAKLAŞIMIN UYGULAMASI

Önerilen metodolojiyi değerlendirmek için, GRASP yordamı McMullen [6] tarafından sağlanan ve EK'te verilen birçok test problemini çözmek için kullanılmıştır. GRASP yordamının performansını ölçmek amacıyla da üç performans ölçütü kullanılmıştır. Bunlar, her bir hazırlık seviyesi için kullanım oranı cinsinden en iyiye kıyasla *% başarısızlık*, en iyiye kıyasla GRASP'ın *yüzdelik performans'*ı ve GRASP çözümü için gerekli işlemci (CPU) süresinin en iyi çözüm için gerekli işlemci süresine oranı (*CPU oranı*).

İlk ve üçüncü ölçüt için küçük değerler arzu edilirken, ikinci ölçüt için ilgili değerlerin mümkün olduğunca 1'e yakın olması arzu edilmektedir. Burada, üç ölçütün de tek bir değerden ziyade bir oran içerdiğini not etmek gerekmektedir. Aşağıda, önceki kesimde verilen örnek problem için ilgili performans ölçütlerinin nasıl hesaplandığı örneklenmiştir. Bu amaçla Çizelge 1'deki değerler verilmiştir.

Çizelge 1. Örnek problem için Rota Birleştirmeli GRASP yordamının en iyiye kıyaslanması

Hazırlıklar	En İyi Kullanım Oranı	Rota Birleştirmeli GRASP Kullanım Oranı	% Başarısızlık	Başarısızların sayısı
3	59,028	59,028	0,00	0
4	29,028	29,028	0,00	0
5	15,028	15,028	0,00	0
6	11,361	11,361	0,00	0
7	7,361	7,361	0,00	0
8	6,361	6,528	2,62	2
9	5,194	5,194	0,00	0
10	4,361	4,361	0,00	0
11	4,028	4,194	4,12	2
12	3,361	3,361	0,00	0

Çizelgedeki ilk üç sütun, sırasıyla, gereksinim duyulan hazırlık sayılarını, ilgili hazırlık seviyelerinde tam sayımlama ile bulunan en iyi kullanım oranlarını ve GRASP kullanım oranlarını göstermektedir. Dördüncü sütun, ilgili hazırlık seviyesinde, GRASP kullanım oranı ile en iyi kullanım oranı arasındaki farkın en iyi kullanım oranına bölünmesi sonucu bulunan yüzdesel başarısızlığı ifade etmektedir. Son sütun ise, ilgili hazırlık seviyesi için tam sayımlama ile bulunan en iyi kullanım oranı ile GRASP kullanım oranı aralığında ve GRASP kullanım oranından daha iyi kaç tane değer bulunduğunu göstermektedir. Örnek problem için, ortalama başarısızlık, *% başarısızlık* değerlerinin ortalaması olup, (4,12+2,62)/10=%*0,674* bulunur. Toplam başarısızların sayısı 4 ve toplam olası sıra sayısı, Denklem 3'ten, 27.720 olduğundan *yüzdelik performans* (27.720-4)/27.720=*%99,9856* bulunur. Bu sonuç, Rota Birleştirmeli GRASP'ı *99,9856.* yüzdelik seviyeye yerleştirmektedir. GRASP ve tam sayımlama için gerekli işlemci süreleri, örnek problem için, sırasıyla 111 ve 363 milisaniye olarak bulunmuştur. Böylece *CPU oranı* da 111/363=*0,306* olarak bulunur.

Tüm etkin sınırların (gerek sezgisel ve gerek sayımlama için) türetilmesi Pentium M 1.6Ghz 988 MHz işlemci ile gerçekleştirilmektedir. EK'te verilen problemler için, GRASP'ın yeterliliğini test etmek amacıyla performans ilişkili sonuçları, McMullen [6] tarafından raporlanan, YA, GA ve Kohonen SOM arama sezgisellerinin sonuçlarıyla kıyaslanmıştır.

V. UYGULAMA SONUÇLARI

GRASP yordamı Delphi 7 programlama dili ile kodlanmış ve her bir problem için farklı bir çekirdekle (seed) 25 tekrar yapılmıştır. Çizelge 2'de ilgilenilen üç performans ölçütünün ortalama sonuçları gösterilmektedir. Çizelge kullanılan arama sezgisellerine göre organize edilmiştir.

Çizelge 2. Çözüm yaklaşımları için performans ölçütleri

	Problem Seti 1	Problem Seti 2	Problem Seti 3	Genel Ortalama
Yasaklı Arama(McMullen [6])				
Ortalama % başarısızlık	0,7813	0,7856	2,2344	1,306
Ortalama yüzdelik perf.	99,9917%	99,9968%	99,9986%	99,9960
CPU oranı	0,4330	0,1276	0,0029	0,0079
Genetik Algoritmalar (McMullen [6])				
Ortalama % başarısızlık	4,4811	1,1333	4,1011	3,139
Ortalama yüzdelik perf.	99,9778%	99,9946%	99,9919%	99,9890
CPU oranı	0,4330	0,1276	0,0029	0,0079
Kohonen SOM (McMullen [6])				
Ortalama % başarısızlık	0,7271	1,7978	10,1000	4,49
Ortalama yüzdelik perf.	99,9976%	99,9980%	99,9987%	99,9980
CPU oranı	2,1669	3,129	0,084	0,2208
Rota Birleştirmeli GRASP				
Ortalama % başarısızlık	0,0584	0,1299	0,1942	0,1330
Ortalama yüzdelik perf.	99,9965%	99,9975%	99,9999%	99,9981
CPU oranı	0,3153	0,1339	0,0035	0,0095

Sonuçlar incelendiğinde Rota Birleştirmeli GRASP'ın, tüm ölçütler cinsinden, diğer sezgisellerle yarışır performans sağladığı görülmektedir. Özellikle *Ortalama % başarısızlık* ölçütü yönünden GRASP diğer sezgisellere büyük üstünlük sağlamaktadır. Problem boyutu büyüdüğünde GRASP'ın ortalama yüzdelik performans'ı daha da iyileşmekte ve ortalama % başarısızlığındaki artış, diğer sezgisellere kıyasla, çok daha az miktarda olmaktadır.

Sonuçlar, en kötü CPU oranını Kohonen SOM' yaklaşımının verdiğini göstermektedir. Her ne kadar, problem seti 1 için, GRASP'ın CPU oranı diğer tüm sezgisellerden daha iyi ise de, problem boyutu büyüdüğünde YA ve GA sezgisellerinin CPU oranları GRASP ve Kohonen SOM'a kıyasla daha üstün hale

gelmektedir. Bu durum, genel CPU oranı ortalamalarına da yansımaktadır. Rota Birleştirmeli GRASP'ın CPU oranı performansının, problem boyutu büyüdükçe YA ve GA sezgisellerininkinden sonra geleceği açıktır, ancak aradaki fark çok büyük gibi görünmemektedir.

Bu çalışmada, deney sonuçları göstermiştir ki, GRASP için göreceli başarısızlıkların çoğu en küçük ve en büyük hazırlık sayıları arasında ortaya çıkmaktadır. Bir başka ifadeyle, Rota Birleştirmeli GRASP, uç hazırlık seviyelerinde, kullanım oranı cinsinden daha iyi performans sağlamakta iken, orta seviyeli hazırlıklarda kötü performans sergilemektedir.

VI. SONUÇLAR VE ÖNERİLER

Bu çalışmada, sıra bağımlı hazırlık sürelerinin küçük ama önemli olduğu karışık-model montaj hattı sıralama problemi ele alınarak, popüler bir sezgisel olan, GRASP (açgözlü rassallaştırılmış uyarlamalı arama yordamı) ile çözümler araştırılmıştır. Çalışmada, yeni bir oluşturma algoritması geliştirilmiş ve yerel arama amacıyla bir melez (hybrid) Değişken Komşuluk Arama algoritması sunulmuştur. Ayrıca eldeki çözümlerin iyileştirilmesi stratejisi olarak da rota yenileme uygulanmıştır.

Gereksinim duyulan hazırlık sayısıyla ilişkili kullanım oranı yönünden arzu edilebilir üretim sıralarının bulunması amacıyla Etkin Sınır yaklaşımı kullanılmıştır. Literatürden alınan birçok test problemi GRASP yordamı kullanılarak çözülmüş ve elde edilen sonuçlar diğer popüler sezgisellerin ―Yasaklı Arama, Genetik Algoritmalar, Kohonen Self-Organizing Map― sonuçlarıyla kıyaslanmıştır.

Sonuçlar, GRASP'ın tam sayımlamaya kıyasla çok daha kısa sürede en iyiye yakın sonuçlar ürettiğini ve diğer sezgisellerle yarışır performans sergilediğini göstermektedir. GRASP'ın ortalama % başarısızlık performansı diğer tüm sezgisellerden açıkça daha iyidir. Problem boyutunun büyümesiyle birlikte, tüm sezgiseller için ortalama yüzdelik performanslarda bir iyileşme gözlemlenmekte, göreceli olarak GRASP yordamındaki iyileşme miktarı diğerlerine göre daha fazla olmaktadır. Ayrıca, her iki performans ölçütü cinsinden Rota Birleştirmeli GRASP'ın, diğer sezgisellere kıyasla, daha iyi genel ortalama performansı sağladığı görülmektedir.

CPU oranı ölçütü cinsinden sonuçlar değerlendirildiğinde, en kötü performansı Kohonen SOM sezgiselinin verdiği gözlemlenmektedir. Bununla birlikte, GRASP

sezgiselinin problem boyutu büyüdükçe performansının kötüleşmekte olduğu ancak Yasaklı Arama ve Genetik Algoritmalara kıyasla aranın fazla açılmadığı görülmektedir. CPU oranı genel ortalama sonuçları en iyi performansın Yasaklı Arama ve Genetik Algoritmalara ait olduğunu göstermektedir.

Büyük boyutlu problemlerde GRASP'ın CPU oranını kötüleştiren en önemli etkenlerden birisi, ilgili komşulukların çok fazla sayıda uygun çözümler içerebilmesidir. Böylece komşuluk boyutu da büyümektedir. Bir başka etken ise, GRASP'ın kullandığı başlangıç çözüm türetme ve yerel arama algoritmalarının yapılarıdır. Rassal olarak bir başlangıç çözüm türetmek, GRASP'ın türetme mekanizmasına kıyasla, oldukça kısa süre gerektirmektedir. Ayrıca, yerel aramada kullanılan komşuluklar (ikili değişim, ileriye ve geriye kaydırma) uygun arama uzayının taranmasının bitirilmesi için sürecin birçok kez baştan başlatılmasını gerektirmekte bu da gerekli olan süreyi arttırmaktadır. Bu nedenle, GRASP'ın süre yönlü performansının iyileştirilmesi için daha etkin komşulukların geliştirilmesi yazar tarafından gelecek için potansiyel bir araştırma alanı olarak görülmektedir.

EK

Çizelge 3, 4, 5

Çizelge 3. Problem Seti 1 (ürün karmasındaki her bir ürün tipinden miktarlar (br)-toplam talep 10 br)

Proble	Ürün 1	Ürün 2	Ürün 3	Ürün 4	Ürün 5	Çözümler
B	6	1	1	1	1	5.040
C	5	2	1	1	1	15.120
D	4	2	2	1	1	37.800
E	4	3	1	1	1	25.200
F	3	3	2	1	1	50.400
G	3	2	2	2	1	75.600
H	2	2	2	2	2	113.400

Çizelge 4. Problem Seti 2 (ürün karmasındaki her bir ürün tipinden miktarlar (br)-toplam talep 12 br)

Problem	Ürün 1	Ürün 2	Ürün 3	Ürün 4	Ürün 5	Çözümler
B	8	1	1	1	1	11.880
C	7	2	1	1	1	47.520
D	6	3	1	1	1	110.880
E	6	2	2	1	1	166.320
F	5	3	2	1	1	332.640
G	5	2	2	2	1	498.960
H	4	3	2	2	1	831.600
I	4	4	2	1	1	415.800
J	3	3	2	2	2	1.663.200

Çizelge 5. Problem Seti 3 (ürün karmasındaki her bir ürün tipinden miktarlar (br)-toplam talep 15 br)

Problem	Ürün 1	Ürün 2	Ürün 3	Ürün 4	Ürün 5	Çözümler
B	11	1	1	1	1	32.760
C	10	2	1	1	1	180.180
D	9	3	1	1	1	600.600
E	7	5	1	1	1	2.162.160
F	7	3	2	2	1	10.810.800
G	6	3	3	2	1	25.225.200
H	5	3	3	3	1	50.450.400
I	4	3	3	3	2	126.126.000
J	3	3	3	3	3	168.168.000

KAYNAKLAR

[1] Ş. Alpay, "*Birden fazla montaj bandı ve ürün çeşidi olan işletmelerde bilgisayar destekli üretim programı ve çizelgeleme sistemi tasarım"ı*, Thesis (Master) (in Turkish), Eskisehir Osmangazi University, Turkey, 1997.

[2] J. Miltenburg, "Level schedules for mixed-model assembly lines in just-in-time production systems". *Management Science*, 35 (2), 192-207, 1989.

[3] P.R. McMullen, "JIT sequencing for mixed-model assembly lines with setups using tabu search", *Production Planning and Controlling*, 9 (5), 504-510, 1998.

[4] P.R. McMullen, G.V. Fraizer, "A simulated annealing approach to mixed-model sequencing with multiple objectives on a JIT line", *IIE Transactions*, 32 (8), 679-686, 2000.

[5] P.R. McMullen, "An efficient frontier approach to addressing JIT sequencing problems with setups via search heuristics", *Computers & Industrial Engineering*, 41, 335-353, 2001.

[6] P.R. McMullen, "A Kohonen self-organizing map approach to addressing a multiple objective , mixed-model JIT sequencing problem", *Int. J. Production Economics*, 72, 59-71, 2001.

[7] P.R. McMullen, "An ant colony optimization approach to addressing a JIT sequencing problem with multiple objectives", *Artificial Intelligence in Engineering*, 15, 309-317, 2001.

[8] P.R. McMullen, "A beam search heuristic method for mixed-model scheduling with setups", *Int. J. Production Economics*, 96, 273-283, 2005.

[9] H.S. Cho, C.H. Paik, H.M. Yoon, H.G, Kim, "A robust design of simulated annealing approach for mixed model sequencing", *Computers & Industrial Engineering*, 48, 753-764, 2005.

[10] S.A. Mansouri, "A multi-objective genetic algorithm for mixed-model sequencing on JIT assembly lines", *European Journal of Operational Research*, 167, 696-716, 2005.

[11] R. Tavakkoli-moghadam, A.R. Rahimi-Vahed, "Multi-criteria sequencing problem for a mixed-model assembly line in a JIT production system", *Applied Mathematics and Computation*, *Article in Press,* 2006.

[12] T.A. Feo J. Bard, “Flight scheduling and maintenance base planning”, *Management Science*, 35(12), 1415–1432, 1989.

[13] M.G.C. Resende, C.C. Riberio, Chapter 8: Greedy randomized adaptive search procedures, *In*: Glower F.W., Kochenberger G.A., (Eds.), *Handbook of Metaheuristics*, *International Series in Operations Research and Management Science*, Vol. 57, Kluwer Academic Publishers, 2002

[14] S.R. Gupta, J. Smith, “Algorithms for single machine total tardiness scheduling with sequence dependent setups”, *European Journal of Operational Research*, 175, 722-739, 2006

[15] R.M. Aiex, S. Binato, M.G.C. Resende, “Parallel GRASP with path-relinking for job shop scheduling”, *Parallel Computing*, 29, 393-430, 2003.