Aralık Çizelgelemede Kapasite Planlama ve Kapasite Artırımı Problemleri

Proje No: 109M576

Yrd.Doç.Dr. Deniz TÜRSEL ELİİYİ

MAYIS 2012

İZMİR

ÖNSÖZ

Bu rapor, TÜBİTAK tarafından desteklenen 109M576 nolu “Aralık Çizelgelemede Kapasite Planlama ve Kapasite Artırımı Problemleri” başlıklı proje kapsamında yapılan çalışmaların sonuç raporu olarak hazırlanmıştır. Projeye desteğinden dolayı TÜBİTAK’a teşekkür ederiz.

Gerek teorik altyapı gerekse kodlama ve analiz aşamalarında projeye önemli katkılarda bulunan doktora öğrencisi Uğur Eliiyi’ye katkılarından dolayı içtenlikle teşekkür ederiz.

İÇİNDEKİLER

TABLO LİSTESİ

ŞEKİL LİSTESİ

ÖZET

Bu projede yöneylem araştırmasının çizelgeleme alanında önemli konulardan olan aralık çizelgeleme ele alınmıştır. Rezervasyon sistemleri dahil olmak üzere birçok uygulama alanında kullanılan pratik değeri yüksek bu problem kapsamında taktik ve operasyonel kararların birleştirilmesi hedeflenmiştir.

Rezervasyon sistemlerinde kullanılacak makine veya operatör sayısının ve maliyetinin belirlenmesi projede ele alınan taktik kararları, seçilen operatörler ile hangi iş veya rezervasyon kümesinin işleneceğinin, dolayısıyla karın belirlenmesi ise operasyonel kararları oluşturmaktadır. Literatürde daha önce ayrı ayrı ele alınan ve hiyerarşik sırada incelenen bu problemlerin aynı anda çözümünü sağlayan matematiksel modeller ve çözüm yaklaşımları projenin ana kapsamını oluşturmaktadır. Özellikle mevsimsellik gösteren rezervasyon sistemlerinde yararlı olacak bu yeni problemlerin literatürde aralık çizelgelemede kapasite planlama ve kapasite artırımı alanında yer alan önemli bir boşluğu dolduracağı ve bilimsel açıdan önemli katkı sağlayacağı düşünülmektedir.

Bu kapsamda öncelikle temel sabit iş çizelgeleme problemi ele alınmış, bu problem özelinde operasyonel ve taktik kararları birleştiren yeni bir bütünleşik problem tanımlanarak bu problem için bir matematiksel model geliştirilmiştir. Bu problem için polinom zamanlı bir kesin çözüm yaklaşımı geliştirilmiş ve sayısal deneylerle performans analizi yapılmıştır. Projenin daha sonraki kısımlarında daha zor problemler olan ve pratik önemi yüksek kar kısıtlı sabit iş çizelgeleme ve çalışma zamanı belirleme problemleri bütünleşik şekilde tanımlanarak analiz edilmiştir. Bu problemler için problemlerin yapısal özelliklerden yararlanan etkin ve verimli çözüm yöntemleri geliştirilmiş ve sayısal deneylerle performans ölçümü gerçekleştirilmiştir. Son olarak NP-zor bir problem olan bütünleşik değişken iş çizelgeleme ele alınmış ve problem için çok etkin bir sezgisel çözüm algoritması geliştirilmiştir. Projede geliştirilen model ve yöntemlerin ülkemizin teorik birikimine ve uygulamaya katkı sağlaması hedeflenmiştir.

Anahtar Kelimeler: Aralık çizelgeleme, kapasite planlama ve kapasite artırımı, Matematiksel programlama ve optimizasyon, sezgisel yöntemler, sabit iş çizelgeleme, değişken iş çizelgeleme.

ABSTRACT

In this project we consider interval scheduling, which is an important topic in the scheduling area of operations research. The aim of the project is to merge and integrate the tactical and operational decisions in this problem of high practical value with many real-life applications including reservation systems.

The determination of the number/cost of machines/operators to be used shapes the tactical decisions, whereas selecting the subset of jobs/reservations to be processed by the decided capacity level while maximizing the obtained profit identifies the operational decisions in the problem. The main scope of the project includes novel mathematical models and respective solution procedures to combine these decisions, which have been previously dealt separately (in a hierarchical manner) in literature. The simultaneous solution of these two problems is useful especially in systems showing seasonality, and the developed models and the solution approaches are expected to fill and important gap in literature on capacity planning and extension.

In this respect, initially the basic fixed job scheduling is considered in the project, and a new combined mathematical model is developed for integrating tactical and operational decisions in this context. A polynomial time exact solution algorithm is proposed for the problem and the performance of the algorithm is tested via extensive computational experimentation. On later stages, harder combined problems of high practical value are analyzed, namely the profit-constrained fixed job scheduling problem and the working time determination problem. We develop efficient and effective solution approaches exploiting the structural properties of these problems and test the performances experimentally. In the last stage, the NP-hard problem of combined variable job scheduling problem is considered, and a very efficient heuristic algorithm is proposed. The products of this project, which are the unique models and solution methods, are intended to contribute to our country's theoretical and practical knowledge base.

Keywords: Interval scheduling, capacity planning and capacity expansion, mathematical programming and optimization, heuristic methods, fixed job scheduling, variable job scheduling.

1. GİRİŞ

Günümüzde birçok sektörde varolan sıkı rekabet ortamı firmalar için avantajlarını korumayı çok zor hale getirmektedir. Yabancı rakip firmalar ve teknolojinin hızlı bir şekilde gelişmesi firmaları daha yenilikçi politikalara itmekte veya fason üretim (outsourcing) yoluyla fiyat avantajı sağlamaya yönlendirmektedir. Tüketici elektroniği, telekomünikasyon ve tekstil sektörleri bahsedilen rekabete örnek olarak verilebilir. Bu gibi sektörlerde yaşanan rekabet firmaların değişen talep veya yeni üretim teknolojilerine uyum sağlayabilecek esnek imalat sistemlerine ve esnek kapasitelere sahip olmalarını gerektirmektedir.

Büyük firmalara çalışan fason üreticiler için de aynı durum söz konusudur. Zira firmanın talebindeki herhangi bir oynama fason üreticiyi de doğrudan etkileyebilmektedir. Fason üreticiler genelde atölye tipi ve sipariş üzerine üretim yapan küçük şirketlerden oluşmaktadır ve çok sayıda olduklarından kontratlarını devam ettirebilmeleri üretim ve operasyonlarının dikkatli yönetimiyle mümkün olabilmektedir.

Üretim yönetiminde en önemli konulardan biri de gelen işlerin makineler üzerinde çizelgelenmesidir. Bu karar temin süreleri, makinelerin kullanım oranları ve üretim hızı gibi performans ölçütlerini doğrudan etkilemektedir. Jain ve Meeran (1999) atölye tipi üretimde çizelgelemeyle ilgili kapsamlı bir literatür taraması çalışması yapmışlardır. Bu gibi sistemlerde kapasite planlaması da makine ve işçi kullanım oranlarını ve sistemin genel performansını etkileyecek çok önemli bir başka karardır. Özellikle küçük işletmelerde bu kararların sistem üzerindeki sonuçları daha belirleyici olmaktadır. Bu iki karar hizmet sektöründe de imalat sektöründe olduğu kadar önemlidir.

Aralık Çizelgeleme (Interval Scheduling) problemi hizmet ve imalat sektöründe yukarıda bahsedilen sistemlerde dahil olmak üzere oldukça sık rastlanan bir çizelgeleme problemidir. Problem, geliş ve termin zamanları önceden belirlenmiş olan işlerin (görevlerin) paralel ve özdeş makineler (kaynaklar) tarafından işlenmesini içermektedir. Problem özellikle kaynak kullanım sürelerinin önceden belirlenerek ayırtıldığı rezervasyon sistemlerinde (otel rezervasyon, araç kiralama/tamir, kaynak çizelgeleme, derslik çizelgeleme vb.) çok önemli bir analiz aracıdır. Aralık çizelgeleme iki ana başlık altında incelenir: Sabit İş Çizelgeleme (Fixed Job Scheduling) ve Değişken İş Çizelgeleme (Variable Job Scheduling). Sabit iş çizelgelemede (SİÇ) işlerin işlem süreleri geliş ve termin zamanları arasındaki süreye eşittir. Dolayısıyla bir iş sisteme girdiği an (geliş zamanında) işleme alınmalıdır, aksi halde işlenemez ve o işten elde edilecek getiri kaybolmuş sayılır. Değişken iş çizelgelemede (DİÇ) ise işlerin işlenme süreleri geliş ve termin zamanları arasındaki süreden daha kısadır. Bu durumda herhangi bir işin geliş zamanından sonra işleme başlamasına kadar sistemde bir süre kalabilmesi mümkündür. Her iki problem için taktik ve operasyonel olmak üzere iki versiyon bulunmaktadır. Taktik problem tüm işleri işleyebilecek makinelerin sayı veya kullanım maliyetlerinin enküçüklenmesini hedeflerken operasyonel problemde amaç mevcut makinelerle işlenebilecek iş kümesinin toplam sayısını veya karını enbüyüklemektir.

Pratik önemine rağmen literatürde aralık çizelgeleme üzerine az sayıda çalışma mevcuttur, ancak tüm bu çalışmalarda problemin gerçek hayat uygulamalarından bahsedilmiştir. Önceki çalışmalar operasyonel ve taktik problemlerinin polinom zamanda çözülebileceğini, ancak çalışma zamanı (working time), yaygınlık zamanı (spread time), ve makinelerin işleyebilirlikleri (eligibility) gibi kısıtlar eklendiğinde problemlerin zorlaştığını işaret etmektedir. Literatürde yer alan çalışmalar bir sonraki bölümde incelenecektir.

Aralık çizelgeleme problemi, hem SİÇ hem de DİÇ formlarında çok geniş bir uygulama alanına sahip önemli bir problemdir. Her iki problemin taktik ve operasyonel versiyonları çeşitli uygulamalarda kullanılmıştır. Ancak taktik ve operasyonel problemler tüm yazarlar

tarafından ayrı ayrı çalışılmıştır, literatürde bu iki versiyonun özelliklerini birleştiren bir modele rastlanmamıştır.

Rezervasyon sistemlerinde, sistemde bulundurulacak makine sayısı en önemli faktördür ve işlenebilecek işleri doğrudan belirlemektedir. Literatürde bu konudaki çalışmaların hepsi planlamada taktik modeli kullanmaktadır. Bu çalışmalarda gözlemlenen önemli bir eksiklik şöyle ifade edilebilir: Herhangi bir rezervasyon sisteminde kapasite planı yapabilmek için planlama periyodunun oldukça uzun olması gerekmektedir. Ancak sistemlerin dinamik yapısı uzun dönemde rezervasyonlarda olabilecek bazı değişiklikleri de beraberinde getirmektedir. Şu ana kadar yapılmış tüm çalışmalar bu gerçeği göz ardı ederek planlama dönemi boyunca olabilecek değişiklikleri yok saymaktadır. Bunun yanında taktik problem kapasiteyi planlayabilmek için uzun dönemde iş gelişlerinin tam olarak kestirilebileceği varsayımına dayanmaktadır ki bu çoğu zaman geçerli bir varsayım değildir ve geliştirilen çözümlerin doğruluğunu ve geçerliliğini tartışılabilir kılmaktadır. Operasyonel problem daha kısa dönemli bir planlama içerdiğinden iş gelişlerinin daha isabetli tahmin edilebilmesi (rezervasyonların teyid edilmiş olmasıyla) mümkündür, dolayısıyla problem verisi daha gerçekçi olmaktadır. Ancak literatürde operasyonel problemin özelliklerinin kapasite planlamada hiç kullanılmadığı görülmektedir; bu önemli bir boşluğu oluşturmaktadır. Üstelik şu ana kadar yapılmış çalışmalar probleme yalnızca belli bir anda bakarak kaç makine gerektiği üzerinde durmuştur. Literatürde bir rezervasyon sisteminin kapasite artırımına dair herhangi bir çalışmaya rastlanmamıştır, oysa bu konu mevsimsel talep değişkenliği gösteren (tekstil atölyeleri, araba kiralama, otel rezervasyonları, rıhtım atama vb.) sistemlerde oldukça değerli bir çalışma alanını oluşturmaktadır. Burada bahsedilen mevsim kavramının bazen gün bazen haftalara, hatta bazı sistemler için saatlere karşılık gelebileceği unutulmamalıdır.

Bu projede taktik ve operasyonel problemlerin özellikleri birleştirilerek aralık çizelgelemede kapasite planlaması ve kapasite artırımı için yeni ve özgün matematiksel modeller geliştirilmiştir. Problemler için yapısal özelliklerden yararlanacak etkin ve verimli çözüm yöntemlerinin geliştirilmesi projenin ana amacını oluşturmaktadır. Bu bağlamda projenin literatürde kapasite planlama ve kapasite artırımı alanında yer alan önemli bir boşluğu dolduracağı ve bilimsel açıdan önemli bir katkı sağlayacağı düşünülmektedir. Çalışılan problemlerden bazıları polinom zamanlı, bazıları ise polinom zamanlı olmayan NP-zor problemlerdir. Ortaya çıkarılacak model ve yöntemlerin ülkemizin teorik birikimine ve uygulamaya katkı sağlaması hedeflenmiştir.

Bir sonraki bölümde operasyonel ve taktik SİÇ problemleri hakkında problem tanımları ve literatür incelemesi ve genel bilgiler sunulacaktır. Üçüncü bölümde izlenen yöntem üzerinde durulacak ve bilinen çözüm yöntemleri özetlenecektir. Dördüncü bölümde ise çalışılan tüm problemler için elde edilen bulgular ve geliştirilen karar destek sisteminin ayrıntıları ayrı alt başlıklar altında sunulacaktır. Beşinci ve son bölüm sonuç ve önerilerden oluşmaktadır.

2. GENEL BİLGİLER: SABİT İŞ ÇİZELGELEME PROBLEMİ

Bir önceki bölümde tanımlanan aralık çizelgeleme problemi matematiksel olarak ifade edildiğinde, sistemde n adet iş ve m adet paralel makine bulunmaktadır. Her j işi için belirlenmiş bir geliş zamanı (r_j), termin zamanı (d_j), işlem süresi (p_j) mevcuttur. Eğer $(p_j = d_j - r_j)$ ifadesi tüm işler için geçerli ise bahsi geçen aralık çizelgeleme problemi bir SİÇ problemidir, aksi halde en az bir iş için $(p_j \leq d_j - r_j)$ ifadesi geçerlidir ve problem bir DİÇ problemidir. Bu durumda yukarıda anlatılanlardan da anlaşılacağı üzere, SİÇ problemi aslen DİÇ probleminin özel bir durumudur.

Her iki problem için taktik ve operasyonel olmak üzere iki versiyon bulunmaktadır. Taktik problem için k makinesinin kullanım maliyeti (c_k) parametresiyle ifade edilir. Eğer tüm makinelerin maliyetleri birbirine eşit ise amaç minimum makine sayısının belirlenmesine dönüşmektedir. Operasyonel problem için ise j işinin getireceği kar (yani görece önemi veya ağırlığı) (w_j) parametresiyle gösterilmektedir ve aynı şekilde eğer tüm işlerin ağırlıkları aynı ise amaç en fazla sayıda işi işleyebilmeye dönüşmektedir.

Problemin varsayımları aşağıdaki şekilde sıralanabilir:

1. Her makine aynı anda tek bir iş işleyebilir ve her iş aynı anda yalnızca tek bir makine tarafından işlenebilir.
2. Makinelerin her an iş yapabileceği (uygun oldukları) varsayılmaktadır ve herhangi bir işleyebilirlik kısıtı bulunmamaktadır; dolayısıyla her makine her işi işleyebilir.
3. Bir iş işlenmeye hangi makine üzerinde başlamışsa o makine üzerinde kesintisiz olarak işlemini tamamlamalıdır.
4. Tüm parametreler tam sayılardan oluşmaktadır ve önceden kesin olarak bilinmektedir.

Operasyonel ve taktik SİÇ problemlerinin matematiksel modelleri için ortak bir karar değişkeni gerekmektedir.

$$x_{jk} = \begin{cases} 1, & \text{eğer } j \text{ işi } k \text{ makinesinde işleniyorsa,} \\ 0, & \text{diğer türlü.} \end{cases} \qquad \forall j,k.$$

Bu ikili karar değişkeniyle operasyonel SİÇ probleminin matematiksel modeli aşağıdaki gibi ifade edilmektedir:

Operasyonel SİÇ Modeli:

$$\text{Enb} \sum_{k=1}^{m}\sum_{j=1}^{n} w_j x_{jk} \qquad (1)$$

$$\sum_{k=1}^{m} x_{jk} \le 1 \qquad j = 1,...,n \qquad (2)$$

$$\sum_{j\in P_a} x_{jk} \le 1 \qquad k = 1,...,m \qquad \forall a \qquad (3)$$

$$x_{jk} \in \{0,1\} \qquad k = 1,...,m \qquad j = 1,...,n \qquad (4)$$

Yukarıda verilen amaç fonksiyonu (1) yapılabilecek tüm işlerin kar toplamlarının enbüyüklenmesini ifade etmekte, (2) numaralı kısıt her işin en fazla bir makine tarafından işlenmesini sağlamakta, (3) numaralı kısıt her makinenin en fazla bir işi işlemesini temin etmekte, (4) numaralı kısıt da ikili değişkenleri tanımlamaktadır. Eğer amaç fonksiyonunda yer alan tüm w_j parametre değerleri birbirine eşit ise $(w_j = sabit, \forall j)$ problemin amacı işlenen toplam iş sayısını enbüyüklemeye dönüşmektedir.

Burada P_a daha önceden işlerin geliş ve termin zamanlarına göre belirlenen zaman dilimlerindeki iş kümelerini temsil etmektedir. Yani eğer $\{t_1, t_2, \dots, t_z\}$ dizisi işlerin r_j ve d_j parametrelerinin kronolojik sırada dizilerek tekrarların atılmasıyla oluşturulmuş bir küme ise, P_a kümesi $[t_a, t_{a+1})$ aralığında işlenmesi gereken işleri içermektedir $(a = 1,2,\dots,z-1$ için).

Buna göre (3) numaralı kısıt planlama periyodundaki her zaman dilimi için herhangi bir makinenin en fazla bir iş işleyebileceğini gösterir.

Taktik SİÇ problemini modellemek için ikinci bir ikili karar değişkenine ihtiyaç duyulmaktadır.

$$y_k = \begin{cases} 0, & k \text{ makinesine hiçbir iş atanmadıysa,} \\ 1, & \text{diğer türlü.} \end{cases} \qquad \forall k.$$

Buna göre taktik model iki karar değişkeniyle aşağıdaki gibi ifade edilir.

Taktik SİÇ Modeli:

$$\text{Enk} \sum_{k=1}^{m} c_k y_k \qquad (5)$$

$$\sum_{k=1}^{m} x_{jk} = 1 \qquad j = 1,...,n \qquad (2')$$

$$\sum_{j \in P_a} x_{jk} \le 1 \qquad k = 1,...,m \qquad \forall a \qquad (3)$$

$$x_{jk} \in \{0,1\} \qquad k = 1,...,m \qquad j = 1,...,n \qquad (4)$$

$$x_{jk} \le y_k \qquad k = 1,...,m \qquad j = 1,...,n \qquad (6)$$

$$y_k \in \{0,1\} \qquad k = 1,...,m \qquad (7)$$

Problemin amaç fonksiyonu en genel haliyle kullanılan makinelerin toplam maliyetini enküçüklemeye çalışır. Eğer tüm makine maliyetleri birbirine eşit ise problemin özel bir durumu oluşmakta, bu özel durum kullanılan toplam makine sayısını enküçükleme amacını taşımaktadır.

Bu modelde (2') numaralı kısıtın eşitlik formunda kullanılması her işin işlenmesini sağlamakta ve modelin en iyi çözümünün "0" olmasını engellemektedir. Ayrıca ek olarak gelen (6) numaralı kısıt iki karar değişkenini birbiriyle ilintilendirir ve kullanılmayan bir makinenin herhangi bir işi işlenmesini engeller. (7) numaralı kısıt ise ikili y değişkenini tanımlar. Bu modeldeki m sayısının makine sayısı için bir üst sınır olduğu (örneğin $m = n$) unutulmamalıdır.

Literatürde yer alan çalışmalarda problemin gerçek hayat uygulamalarından bahsedilmiştir. Örneğin Kroon (1990) ve Kroon vd. (1997) taktik SİÇ problemini (TSİÇ) bir havayolu şirketinin havaalanı bakım ekibinin kapasite planlamasında kullanmıştır. Problemde havaalanına iniş yapan uçakların daha önceden belirlenmiş zaman aralıklarında mecburi bakımlarının yapılması söz konusudur. Bu işi plana uygun olarak yürütecek personel sayısının enazlanması bir TSİÇ olarak modellenerek çözülmüştür. Daha sonraki bir çalışmada Kroon vd. (1995) aynı probleme operasyonel açıdan (OSİÇ) yaklaşmış ve mevcut personel sayısıyla bakımı yapabilecek uçakların sayısını enbüyüklemeye çalışmışlardır. Bu çalışmalarda herhangi bir bakım işçisinin bakımını yapabileceği uçak kümeleri önceden tanımlanmıştır. Dolayısıyla her işçi her uçağa atanamamakta, işleyebilirlik kısıtları devreye girmektedir. Başka bir gerçek hayat problemi de otobüs şoförlerinin çizelgelenmesinde TSİÇ problemi kullanılarak Fischetti ve diğerleri (1987, 1989, 1992) tarafından çalışılmıştır. Bu çalışmalarda şoförlerin sendika tarafından sınırlanan çalışma saatleri de probleme kısıt olarak eklenmiş ve çalışma zamanı (working time) ve yaygınlık zamanı (spread time) kısıtları

altında taktik problem ayrı ayrı incelenmiştir. TSİÇ modeli proje yürütücüsü ve diğerleri (Eliiyi vd., 2009a) tarafından bir gerçek hayat probleminden hareketle üniversite oyunlarında kullanılacak araç filosunun kapasite planlamasında ve çizelgelemesinde kullanılmıştır. Diğer gerçek hayat uygulamalarına örnek olarak operasyonel SİÇ probleminin gözlem uydularının çizelgelemesinde kullanılması (Wolfe ve Sorensen, 2000), veri aktarımı (Faigle vd., 1999), derslik çizelgeleme (Kolen ve Kroon, 1991) ve baskılı devre kartı üretiminde çizelgeleme (Spieksma, 1999) verilebilir. Operasyonel DİÇ problemi (ODİÇ) uydudan veri aktarımı probleminde (Rojanasoonthon vd., 2003) ve limanlarda gemilerin rıhtımlara atanması probleminde proje yürütücüsü tarafından (Eliiyi vd., 2009b) kullanılmıştır. Zaman ve işleyebilirlik kısıtları altında operasyonel problem için çalışmalara örnek olarak Eliiyi ve Azizoğlu (2006, 2009c, 2010b, 2011) ve Rossi vd. (2010) verilebilir.

Önceki çalışmalar OSİÇ ve TSİÇ problemlerinin polinom zamanda çözülebileceğini, ancak çalışma zamanı (working time), yaygınlık zamanı (spread time), ve makinelerin işleyebilirlikleri (eligibility) gibi kısıtlar eklendiğinde problemlerin zorlaştığını işaret etmektedir (Kolen ve Kroon, 1992). Bouzina ve Emmons (1996) temel OSİÇ probleminin bir ağ akış problemine dönüştürülerek polinom zamanda çözülebileceğini göstermiştir. Arkin ve Silverberg (1987) ve Hashimoto ve Stevens (1971) taktik model üzerine çalışmış, amaç fonksiyonu makine sayısını enküçüklemek olan taktik SİÇ problemi için polinom zamanlı bir çözüm algoritması geliştirmiştir. Proje yürütücüsü ise (Eliiyi, 2004) ise amaç fonksiyonu toplam makine maliyetini enazlamak olan taktik SİÇ problemi için bir çözüm algoritması geliştirmiştir.

Aralık çizelgeleme hakkında oldukça yeni ve geniş literatür tarama çalışmaları Kovalyov vd. (2007) ve Kolen vd. (2007) tarafından yapılmıştır.

Fischetti ve diğerleri (1987, 1989) taktik SİÇ problemini çalışma ve yaygınlık zaman kısıtları altında çalışmıştır. Yazarlar tarafından otobüs şoförü çizelgelemesinde kullanılan zaman kısıtlı problemlerin uygulama alanları oldukça geniştir. Herhangi bir makine için çalışma zamanı o makinenin toplam işlem yapma süresini kısıtlar. Yaygınlık zamanı ise makinenin işlediği ilk işin başlangıç zamanından itibaren geçen süreyi kısıtlar, bu sürenin içine makinenin işlem yapmadığı zamanlar da dahildir. Yaygınlık zamanı bir bakımdan başlangıç zamanı makinedeki ilk iş tarafından belirlenen dinamik bir vardiyayı ifade etmektedir. Yazarlar ayrı ayrı bu kısıtlar altında taktik SİÇ probleminin NP-zor olduğunu göstermişler, problemler için dal ve sınır algoritmaları geliştirmişlerdir. Daha sonraki bir çalışmada ise (Fischetti ve diğerleri, 1992) bu problemler için sezgisel yöntemler geliştirilmiştir.

Zaman kısıtları altında operasyonel problem için literatürde önde gelen çalışmalar proje yürütücüsü tarafından gerçekleştirilmiştir. Proje yürütücüsü daha önceki çalışmalarında (Eliiyi, 2004) çalışma zamanı kısıtları altında operasyonel SİÇ problemini incelemiş ve problemin NP-zor olduğunu göstermiştir. Eliiyi ve Azizoğlu (2006) yaygınlık kısıtları altında OSİÇ'nin NP-zor olduğunu göstermiştir. Her iki çalışmada da problemler için etkin üst ve alt sınır yöntemleri içeren dal-sınır algoritmaları geliştirilmiştir. Bu problemler için çok iyi sonuçlar veren demet biçimli arama ve ağ akış temelli sezgisel yöntemlerin geliştirildiği bir çalışma da bulunmaktadır (Eliiyi ve Azizoğlu, 2011).

Makine işleyebilirlik kısıtları altındaki aralık çizelgeleme problemlerinde tüm makineler tüm işleri işleyebilmek için uygun değildir. Bu durumda her makine sınıfı için işleyebileceği bir iş kümesi tanımlanmakta veya iş-makine uyumluluk matrisleri oluşturulmaktadır. Daha genel bir yaklaşım olarak makine-bağımlı iş ağırlıkları (w_{jk}) tanımlanmakta ve eğer k makinesi j işini işleyemiyorsa ilgili w_{jk} negatif bir değer almaktadır. Problem hastanelerde ameliyathanelerin operasyonlara atanması veya otel odalarının konaklayıcılara ayrılması gibi birçok hizmet sektörü probleminde, bunun yanı sıra değişken kalitede imalat yapan makinelerin bulunduğu üretim sistemlerinde uygulama alanı bulmaktadır. Makine işleyebilirlik kısıtları altındaki taktik

ve operasyonel SİÇ problemi için literatürdeki başlıca çalışmalar Arkin ve Silverberg (1987), Kolen ve Kroon (1992) ve Kroon vd. (1995, 1997) tarafından yapılmış, çeşitii makine ve iş sınıflarına sahip problemlerin temel probleme kıyasla çok daha zor olduğu gösterilmiş, bu problemler için sezgisel yöntemler ve Lagrange gevşetme yöntemleri geliştirilmiştir. Yaygınlık zamanı ve işleyebilirlik kısıtları altında TSİÇ modeli proje yürütücüsü ve diğerleri (Eliiyi vd., 2009a) tarafından üniversite oyunlarında kullanılacak araç filosunun kapasite planlamasında ve çizelgelemesinde kullanılmıştır. Bu çalışmada sporcuları taşıyacak tüm seferleri gerçekleştirecek bir araç filosunda maliyeti enküçükleyecek şekilde çeşitli araç tipleri gözetilerek kaçar adet bulunması gerektiği düşünülmüştür. Varolan araç yaygınlık zamanı ve işleyebilirlik kısıtları problemi NP-zor hale getirmektedir. Problem için yapısal özelliklerinden faydalanan özel sezgisel yöntemler geliştirilmiş ve deneysel olarak performanslarının çok iyi olduğu gözlemlenmiştir.

Yukarıda kısaca geniş uygulama alanlarına değinilen işleyebilirlik kısıtlarının daha genel bir ifadesi olan makine-bağımlı iş ağırlıkları problemi literatürde sadece proje yürütücüsü vd. (Eliiyi ve Azizoğlu, 2009c; Eliiyi vd. 2009b) tarafından çalışılmıştır. İlk çalışmada yazarlar bu ağırlıkların varlığında operasyonel SİÇ problemini ilk kez ele almış, problemin NP-zor olduğunu kanıtlamış, polinom zamanda çözülebilen bazı özel durumları belirlemiş ve çözüm algoritmaları belirlemişlerdir. Ayrıca genel problem için işler arasında birtakım baskınlık kuralları geliştirilmiş, etkin alt ve üst sınırlardan yararlanarak optimal çözüm üreten bir dal-sınır algoritması tasarlanarak sayısal deneylerle performans ölçümü yapılmıştır. Eliiyi vd. (2009b) ise daha genel olan DİÇ problemini bir gerçek hayat probleminden hareketle ele almıştır. Operasyonel DİÇ problemi için literatürde yalnızca iki çalışma yer almaktadır (Rojanasoonthon vd., 2003; Rojanasoonthon ve Bard, 2005). Makine-bağımlı iş ağırlıkları altındaki operasyonel DİÇ problemi ise ilk kez proje yürütücüsü tarafından ele alınmıştır ve konteyner limanlarında gemi-rıhtım atamaları probleminde kullanılabilecek şekilde modellenmiştir (Eliiyi vd. 2009b). Bu çalışmada öncelikle İzmir Alsancak konteyner limanı gözlenmiş, limana yanaşacak gemilerin körfezde bekleme süreleri izlenmiş ve problemin değişken iş çizelgeleme yapısına sahip olduğu anlaşılmıştır. Daha sonra problemin NP-zor olduğu göz önüne alınarak sezgisel çözüm yöntemleri geliştirilmiştir. Performans ölçümü gerçek verilere uygun olarak türetilen çok sayıda problem üzerinde optimal değerlerle karşılaştırmak suretiyle yapılmış ve geliştirilen sezgisel yöntemlerin çok kısa süreler içinde eniyi çözümlere çok yakın sonuçlar verdiği gözlemlenmiştir.

Problemle ilgili yukarıda verilen literatür analizinden de anlaşılacağı üzere Aralık Çizelgeleme problemi, hem SİÇ hem de DİÇ formlarında çok geniş bir uygulama alanına sahip önemli bir problemdir. Her iki problemin taktik ve operasyonel versiyonları çeşitli uygulamalarda kullanılmıştır. Ancak taktik ve operasyonel problemler tüm yazarlar tarafından ayrı ayrı çalışılmıştır, literatürde bu iki versiyonun özelliklerini birleştiren bir modele rastlanmamıştır. Bu iki modelin özelliklerinin birlikte kullanılması kapasite artırımı yoluyla kapasite planlamaya ve çizelgelemeye bütünleşik bir bakış açısı sağlaması açısından önemlidir. Bütünleşik problem şu soruya cevap arayan çok önemli bir problemdir: “Varolan bir sistemdeki makine sayısı karı istenilen seviyeye çıkarabilmek için ne kadar artırılmalı ve gelen işler bu ek makinelere nasıl atanmalıdır?”

Rezervasyon sistemlerinde, sistemde bulundurulacak makine sayısı en önemli faktördür ve işlenebilecek işleri doğrudan belirlemektedir. Bu gerçekten hareketle yapılan ve yukarıda bir analizi sunulan aralık çizelgelemede kapasite planlamasına dair çalışmalar literatürde belirli bir yere sahiptir. Ne var ki bu çalışmaların hepsi planlamada taktik modeli kullanmaktadır ve dolayısıyla gözlemlenen önemli bir eksiklik şöyle ifade edilebilir: Herhangi bir rezervasyon sisteminde kapasite planı yapabilmek için planlama periyodunun oldukça uzun olması gerekmektedir. Ancak sistemlerin dinamik yapısı uzun dönemde rezervasyonlarda olabilecek bazı değişiklikleri de beraberinde getirmektedir. Şu ana kadar yapılmış tüm çalışmalar bu

gerçeği göz ardı ederek planlama dönemi boyunca olabilecek değişiklikleri yok saymaktadır. Bunun yanında kapasiteyi planlayabilmek için uzun dönemde işlerin gelişlerinin kesin olarak kestirilebileceği varsayımına dayanmaktadır ki bu çoğu zaman geçerli bir varsayım değildir ve geliştirilen çözümlerin doğruluğunu ve geçerliliğini tartışılabilir kılmaktadır.

Operasyonel problem daha kısa dönemli bir planlama içerdiğinden iş gelişlerinin daha isabetli tahmin edilebilmesi (rezervasyonların teyid edilmiş olmasıyla) mümkündür, dolayısıyla problem verisi daha gerçekçi olmaktadır. Ancak literatürde operasyonel problemin özelliklerinin kapasite planlamada hiç kullanılmadığı görülmektedir; bu önemli bir boşluğu oluşturmaktadır. Üstelik şu ana kadar yapılmış çalışmalar probleme yalnızca belli bir anda bakarak kaç makine gerektiği üzerinde durmuştur. Literatürde bir rezervasyon sisteminde kapasite artırımına dair herhangi bir çalışmaya rastlanmamıştır, oysa bu konu mevsimsel talep değişkenliği gösteren (araba kiralama, otel rezervasyonları, rıhtım atama vb.) sistemlerde oldukça değerli bir çalışma alanını oluşturmaktadır.

3. PROJEDE KULLANILAN YÖNTEM VE GEREÇLER

Bu bölümde projede incelenen problemlere baz oluşturan iki model incelenecektir. Operasyonel SİÇ ve makine sayısını enazlayan taktik SİÇ problemleri için varolan çözüm yaklaşımları sunulacaktır. Bu çözüm yaklaşımları çalışılan problemler için geliştirilen çözüm yöntemlerine baz oluşturması açısından önemlidir. Ayrıca projede kullanılan gereçler de bu bölümde açıklanacaktır.

Bir önceki bölümde modeli sunulan operasyonel SİÇ problemi Bouzina ve Emmons (1996) tarafından çalışılmıştır. Yazarlar probleminin $n+1$ düğüme (node) ve $2n$ çizgiye (arc) sahip bir Minimum Maliyetli Ağ Akış Problemi (MMAA) olarak modellenebileceğini ve $O(mn\,logn)$ sürede çözülebileceğini göstermişlerdir.

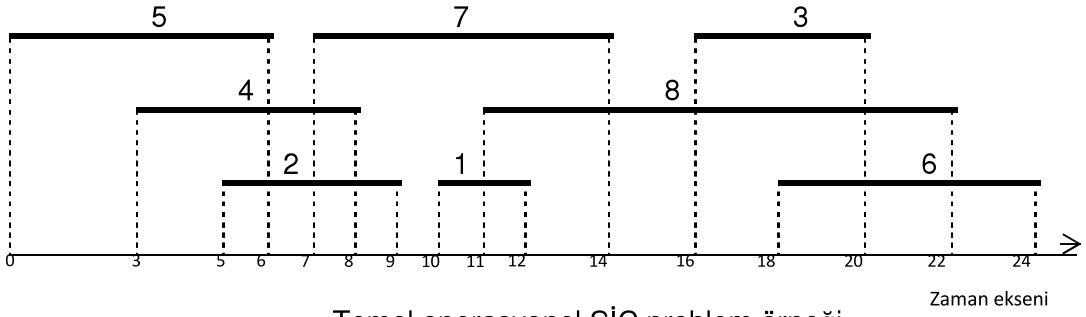
Modelleme şu adımları izleyerek yapılır:

1. İşler hazır olma zamanlarına göre kronolojik sırada dizilir.
2. İçindeki her düğüm bir işe karşılık gelen bir düğüm kümesi, $s = V_1, V_2, ..., V_n$ yaratılır.
3. Ek olarak bir kukla düğüm $t = V_{n+1}$ yaratılır.
4. Her düğüm kendinden bir sonraki düğüme sıfır maliyet ve m birim kapasite ile bağlanır.
5. Her j işine ait V_j düğümü kendinden sonra gelen ve j işiyle zaman ekseninde çakışmayan (üstüste binmeyen) ilk işe ait düğüme bağlanır. Eğer çakışmayan bir iş bulunamıyorsa (V_j, t) çizgisi yaratılır. Yaratılan tüm bu çizgilerin maliyeti $-w_j$ ve kapasitesi bir birimdir.

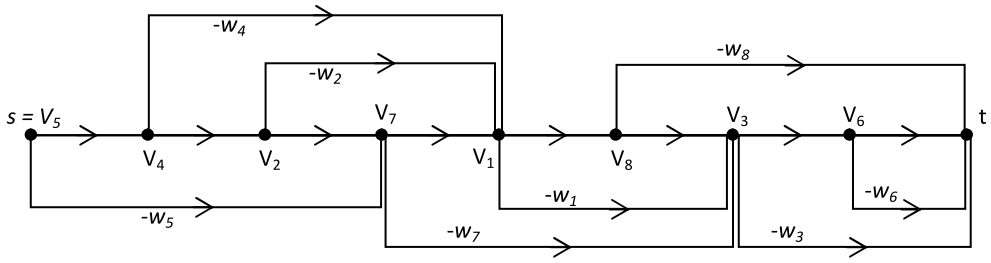
Ortaya çıkan ağda s düğümünden t düğümüne m birim akış gönderecek şekilde MMAA problemi çözülür. Çözüm polinom zamanlıdır, herhangi bir ticari optimizasyon yazılımı (örneğin IBM ILOG CPLEX) yardımıyla kolayca bulunabilir ve bu çözüm operasyonel SİÇ problemi için kesin çözüme karşılık gelmektedir.

Bir modelleme örneği Şekil 1'de gösterilmiştir. Bu örnekte Şekil 1(a)'da zaman ekseni üzerinde gösterilen işler iki paralel özdeş makinede çizelgelenecektir. Koyu renk çizgiler işlere, üzerlerindeki sayılar iş numaralarına karşılık gelmektedir. İşlerin ağırlıkları işlem sürelerine eşit olarak alınmıştır, yani $w_j = p_j, j = 1, ..., 8$.

Şekil 1(b)'de örnek probleme karşılık gelen ağ akış modeli verilmiştir. Modelde maliyeti verilen çizgiler için kapasite bir birim, maliyeti verilmeyen çizgiler için maliyetler sıfır, kapasiteler ise $m = 2$ birimdir. Bu ağda MMAA problemi çözüldüğünde optimal kar seviyesi 35 birim ve çizelgelenen iş kümesi {4,5,6,7,8} olarak bulunmaktadır.



Temel operasyonel SİÇ problem örneği.



Karşılık gelen ağ yapısı.

Şekil 1. Temel operasyonel SİÇ probleminin MMAA problemine dönüştürülmesi. (Eliiyi, 2004)).

Yine bir önceki bölümde modellenen temel taktik SİÇ problemi için makine sayısını enazlayacak optimal çözüm zaman ekseninde işlerin maksimum çakışma sayısının bulunmasıyla $O(n)$ zamanda hesaplanabilir (Hashimoto ve Stevens, 1971).

Matematiksel olarak maksimum çakışma sayısı;

$Maks_a\{|P_a|\}$,
$|P_a|$: P_a kümesinin eleman sayısı,

olarak tanımlanır. Örneğin Şekil 1'de tüm işleri işleyebilmek için gerekli minimum makine sayısı (dolayısıyla taktik problemin çözümü) herhangi bir andaki maksimum iş çakışma sayısı 3 olduğundan 3 olarak hesaplanmaktadır.

Eliiyi (2004) ise $c_k \neq c$ durumundaki optimal çözümün, yani temel taktik SİÇ'de toplam makine maliyetini enazlayacak optimal çözümün $O(n\,logn)$ zamanda bulunabileceğini göstermiştir. Bu çözüm yönteminde öncelikle Hashimoto ve Stevens (1971) çözümü ile tüm işleri işleyebilecek minimum makine sayısı $Maks_a\{|P_a|\}$ olarak hesaplanmakta, daha sonra tüm makineler arasından en düşük maliyete sahip $Maks_a\{|P_a|\}$ adedi seçilmektedir. En düşük toplam maliyetli çözüm bu makinelerden oluşur.

Önceki kısımlarda da belirtildiği gibi literatürdeki çalışmalarda kapasite planlama problemi için taktik versiyon kullanılagelmiştir. Kapasite artırımı problemi ise daha önce çalışılmamıştır. Bu projede bu iki problemi çözebilmek için taktik ve operasyonel problemlerin özelliklerini içeren çözüm yöntemleri kullanılmış ve iki problem birlikte değerlendirilmiştir.

Bu amaçla projede herhangi bir makine sayısı için operasyonel problemin çözümü bilinen m makineli bir sistemde, makine sayısı $m+1$ yapıldığında oluşacak yeni problemin çözümünün eldeki çözüm kullanılarak nasıl bulunacağı araştırılmıştır. Bir sonraki bölümde problemle ilgili ayrıntılar sunulacaktır, ancak genel anlamda Tablo 1'de sağ sütunda bulunan amaç fonksiyonu değerlerinin hesaplanması, bunun için eniyi çözüm veya yaklaşık çözümler üretecek sezgisel yöntemlerin geliştirilmesi amaçlanmıştır.

Tablo 1. Bütünleşik SİÇ ve kapasite artırımı yapısı.

Makine sayısı	***Operasyonel Problemin Amaç Fonksiyonu Değeri***
0	0
1	En kısa yol çözümü (polinom zamanda bulunabilir)
. .	
k	MMAA çözümü (polinom zamanda bulunabilir)
. .	
Üst sınır = $Maks_a\{\|P_a\|\}$	$\sum_{j=1}^{n} w_j$

İncelenen problemler bir sonraki bölümde ayrıntılarıyla sunulacaktır. Tüm problemler için öncelikli hedef kesin çözüm üretmek hedeflenmiş, ancak kesin çözümün mümkün olmaması durumunda yaklaşık çözüm üretmek amacıyla sezgisel yöntemlerden yararlanılmıştır. Analizlerde kısıtlı optimizasyon problemlerinin çözümleri de kullanılmıştır (ör. verilen bir hedef kar seviyesi kısıtıyla makine sayısının enazlanmasını hedefleyen taktik problem). Ayrıca modern bir modelleme ve çözüm yöntemi olarak kısıt programlamadan faydalanılmıştır.

Çalışma yöntemi olarak incelenen tüm problemler için geliştirilen algoritmaların en kötü durum irdelemesi teorik olarak yapılmış, ortalama performansları ise kapsamlı sayısal deneylerle ölçülmüştür. Bu bağlamda elde edilen çözümlerin çeşitli veri setleri üzerinde eniyi çözümlerle karşılaştırılması ve çözüm zamanı açısından değerlendirilmesi yapılmıştır.

Projenin son aşamasında potansiyel karar verici/kullanıcıların probleme yönelik analizleri (değişik problem versiyonlarına ait çeşitli yöntemlerin kullanılması, duyarlılık analizi, verimli çözümler arasında seçim yapma vb.) daha kolay yapabilmeleri için kullanıcı dostu basit bir arayüze sahip bir karar destek sistemi geliştirilmiştir. Söz konusu karar destek sistemi girdi olarak problem tipine göre kullanıcıdan verileri almakta, çıktı olarak ise bir veya daha fazla çözüm yöntemi için karşılaştırmalı sonuçları bir arayüzle görsel olarak sunmaktadır. Kullanıcıya sunulan sonuçlar üzerinde değişiklik yapma, duyarlılık analizi gibi özellikler de sisteme entegre edilmiştir.

Projede kullanılan gereçler olarak, geliştirilen matematiksel modellerin çözümünde yazılım olarak IBM ILOG CPLEX 11.2 optimizasyon paketi sayılabilir. Algoritmalar ve karar destek sistemi MS Visual Studio 2008 .NET ortamında C# ve C++ dilleriyle uygulanmış ve matematiksel modelleme olarak IBM ILOG OPL 6.3 dili kullanılmıştır.

4. BULGULAR

Bu bölümde proje kapsamında incelenen problemlerin tanımları, ilgili kısa literatür özeti, geliştirilen algoritmalar ve deneylerden elde edilen bulgular ayrı başlıklar altında sunulacaktır.

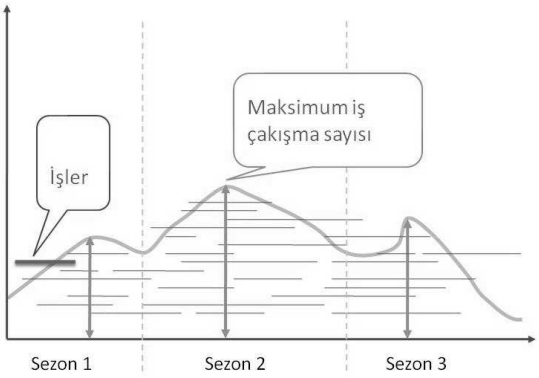
4.1. BÜTÜNLEŞİK SABİT İŞ ÇİZELGELEME PROBLEMİ

Projenin ilk dönemlerinde incelenen bu problemde varolan ve çözümü bilinen bir rezervasyon sistemi için kapasite artırımı ve planlamasıyla birlikte SİÇ probleminin bütünleşik çözümü araştırılmaktadır. Bu sebeple makine sayısının belirli bir α sayısı kadar artırılmasıyla oluşacak operasyonel çözüme ihtiyaç vardır. Genel anlamda bir önceki bölümde yer alan Tablo 1'de sağ sütunda bulunan amaç fonksiyonu değerlerinin üretilmesi söz konusudur.

Tablo 1'deki değerlerin kolaylıkla hesaplanabilir olması bilinen bir çözüm için o çözümü kullanarak polinom zamanda değişik kapasite alternatifleriyle tüm verimli çözümleri üretmeyi getirmektedir. Bu durumda hem kapasite planlama ve kapasite artırım analizi hem de duyarlılık analizi geliştirilen yöntemler kullanılarak kolaylıkla yapılabilir.

Tablo 1'de görüldüğü gibi makine sayısı 1 olduğunda operasyonel problemin özel bir durumuyla karşılaşılmaktadır, ve çözüm en kısa yol çözümüne karşılık gelmektedir. Bu durum şu şekilde açıklanır: Önceki bölümde belirtildiği gibi m makineli temel operasyonel SİÇ problemi MMAA şeklinde modellenerek polinom zamanda çözülebilmektedir. Tek makineli problem m makineli problemin özel bir durumu olduğundan aynı yöntemle çözülebilir. Ancak bir MMAA probleminde m yerine 1 birim akış sağlayacak şekilde bir çözüm istenirse karşılık gelen problem artık bir MMAA problemi değil, daha kolay bir problem olan en kısa yol problemi olmaktadır. Dolayısıyla aynı dönüşüm ve araçlar kullanılarak operasyonel problemin kesin çözümü mümkündür. Solyalı ve Özpeynirci (2009) bu problemin $O(n)$ zamanda çözülebileceğini göstermişlerdir.

Talebin mevsimsel değişkenlik gösterdiği sistemlerde Tablo 1'deki değerlerin her sezonda ayrı ayrı hesaplanması gerekebilecektir. Şekil 2'de bu durum gösterilmiştir. Şekilde işler zaman eksininde yatay çizgilerle ifade edilmiştir. Maksimum iş çakışma sayısı sezonlar arasında farklılık göstermektedir.



Şekil 2. Sezonsallık gösteren bir sistemde SİÇ yapısı.

4.1.1. Bütünleşik Çözüm için Kesin Çözüm Algoritması

Yukarıdaki problem tanımı ışığında geliştirilen algoritmayı daha iyi anlayabilmek için sistemde m makinenin bulunduğunu, planlama periyodu içerisinde -her planlama periyodu bir sezona karşılık gelecektir- n rezervasyon yapıldığını ve işlerin hepsinin eldeki m makine ile işlenemediğini varsayalım $(m < Maks_a\{|P_a|\})$. Karar verici kapasite artırımı kararını verebilmek ve dolayısıyla optimal kapasite seviyesini belirleyebilmek için her bir ek özdeş makinenin kar katkısına ihtiyaç duyacaktır. Bunun için geliştirilen kesin çözüm algoritması yukarıda bahsedilen çözüm yöntemlerinden yararlanmaktadır.

Eğer sistem halihazırda işleyen bir sistem ise işlerin bir çizelgesi mevcuttur. Eğer çözüm mevcut değilse temel operasyonel problemi bir kez çözerek (MMAA çözümüyle) optimal çizelge elde edilebilir. Eldeki makinelerle işlenebilen (çizelgelenmiş) işlerin kümesi S ile ifade edilirse $n - |S|$ iş işlenemiyor (çizelgelenememiş) demektir. Bu durumda çözüm algoritmasının adımları aşağıdaki gibidir.

Bütünleşik Çözüm İçin Algoritma:

1. $n - |S|$ adet çizelgelenememiş iş ile taktik problem çözümünü bul. $A = n - |S|$ işi işleyebilmek için gereken minimum makine sayısı olsun.
2. $k = 1,2, \dots, A$ için;
   k makine ve $n - |S|$ iş ile MMAA çözümü bul. Z_k^* = Optimal amaç fonksiyonu değeri
3. Z_k^* değerlerine göre marjinal katkı hesapla.

Algoritmanın birinci adımındaki A değeri Şekil 2'de "Maksimum iş çakışma sayısı" olarak gösterilen değere karşılık gelmekte ve kapasite için bir üst sınır oluşturmaktadır. Dolayısıyla herhangi bir makinenin kullanılmadan kaldığı durumların oluşmaması için sistemde en fazla $m + A$ makine olabilir. İkinci adımda her ek makine yani her kapasite seviyesi adayı için bir operasyonel problem çözümü bulunmakta ve kar getirileri hesaplanmaktadır. Eğer elde ilk çözüm yoksa bir operasyonel problem de ilk çözüm için çözüleceğinden algoritmada toplamda (taktik problem dahil) $A + 2$ polinom zamanlı problem çözülmüş olacaktır. Dolayısıyla algoritma polinom zamanlıdır.

Z_k^* değerlerine göre marjinal kar getirisinin hesaplanması algoritmanın son adımını oluşturmaktadır. Bu kar getirisi seviyelerine göre karar verici istenilen kapasite artırımını ve

seviyesini seçebilecektir. Oluşacak kar seviyesi serisinin eğilimini daha iyi görebilmek için grafik gösterimden yararlanılabilir.

4.1.2. Sayısal Deney

Bu bölümde bir önceki bölümde verilen Bütünleşik Çözüm İçin Algoritma'nın bir gerçek hayat probleminden hareketle türetilen veri üzerinde deneysel olarak uygulanması açıklanacaktır.

Giriş bölümünde bahsedildiği gibi projede ilgilenilen problem pratikte birçok alanda karşılaşılabilecek bir problemdir. Yapılan araştırmalar sonucunda tekstil sektöründe yer alan dikim atölyelerinde de bu bütünleşik kararların çok önemli olduğu anlaşılmıştır. Dikim atölyeleri genelde fason üretim yapmakta ve üretimde kullanılan dikiş makineleri çoğunlukla kiralama yoluyla sağlanmaktadır. Bu sebeple bu tip atölyelerde kapasite seviyesini değiştirmek kolaylıkla yapılabilmektedir. Ayrıca bu tip atölyelerde işlerin geliş ve termin zamanları oldukça belirli olmaktadır, dolayısıyla sistem SİÇ problemi olarak modellenebilmektedir. Projenin ilk aşamasındaki deneysel çalışmada talep değişkenliğinin sıklıkla görüldüğü tekstil sektöründe bir dikim atölyesinin kapasite artırımı kararı ve çizelgelemesi üzerinde sayısal bir örnek üzerinde durulmuştur. Veri türetme amacıyla bir dikim atölyesi incelenmiş ve görece geliş, termin ve işlem zamanlarına ilişkin ölçümler yapılmıştır.

Çalışma için iki makineye sahip bir dikim atölyesinde bir sonraki sezonda 200 iş rezervasyonu yapılan orta boy bir problem seçilmiştir. Bu 200 iş dikilecek gerçek parça sayısına veya birleştilmiş parçalara yani lotlara karşılık gelecek şekilde düşünülebilir. Planlama zamanında 200 zaman birimi olduğu kabul edilmiştir. Bu zaman dilimleri de karar vericinin belirleyeceği gerçek zaman dilimlerine karşılık gelecek şekilde seçilebilir. Seçilen zaman aralığı birimlerinin işlerin işlem sürelerinin bir katı şeklinde belirlenmesi hesaplama kolaylığı sağlayacağı için tercih edilebilir. İşlerin geliş zamanları bu 200 birimlik planlama periyoduna düzgğn dağılmış olarak kabul edilmiştir. İş başına işlem süresi ve görece ağırlıklar ise sırasıyla (1,10) ve (1,40) aralıklarında düzgün dağılımdan türetilmiştir.

Algoritma MS Visual Studio ortamında Visual C++ diliyle kodlanmış, IBM ILOG CPLEX optimizasyon motorunun entegrasyonu ile taktik problemin, MMAA probleminin ve en kısa yol probleminin çözümü sağlanmıştır. Hesaplamalar 1 GB Ram ve 2.20 GHz Core2Duo işlemcili bir PC üzerinde yapılmıştır.

Sistemde elde bir çözüm olmadığı varsayılarak ilk önce 200 iş ve 2 makine için operasyonel problem bir kez çözdürülmüştür ve optimal çizelge elde edilmiştir. Bu ilk problem P0 olarak adlandırılmıştır. İlk problem için toplam kar seviyesi 2086 olarak bulunmuştur. Optimal çözüme göre gelen 200 işten 117 adeti çizelgelenememiş olarak görünmektedir, yani bir sonraki sezonda gelmesi beklenen bu 200 işten 117'si eldeki kapasite seviyesiyle işlenememektedir.

Algoritmanın ilk adımında 200 iş için taktik problem çözümü elde edilmiştir. Bu çözüm tüm işleri işleyebilmek için kullanılması gereken minimum makine sayısının 12 olması gerektiğini göstermektedir. Ancak burada unutulmaması gereken bir nokta bu sayının bir üst sınır oluşturduğudur. 12 makinelik bir kapasite ile işletilecek bir atölye tüm işleri işleyebilecek olsa da makinelerin kullanım oranlarının çok düşük olması beklenen bir durumdur. Dolayısıyla bu kapasite seviyesi etkin bir şekilde kullanılamayacaktır.

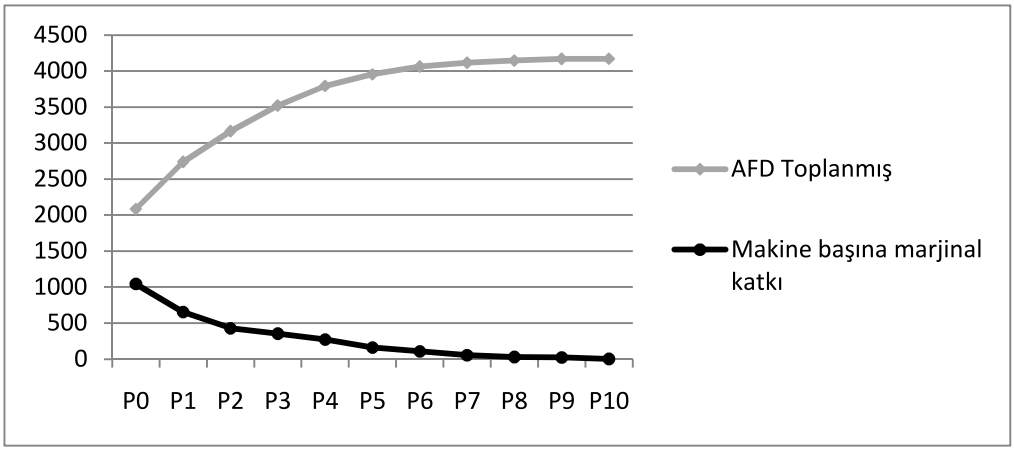
Algoritmanın sonraki adımlarında şu anki kapasite seviyesi olan 2 makine ile üst sınır olarak bulunan 12 makine arasındaki tüm seviyeler için operasyonel problemlerin çözümleri elde edilmiş ve değerlendirilmiştir. Başka bir deyişle 117 iş ve 1, 2, ..., 10 makinelik 10 adet operasyonel alt problem çözülmüş, bu alt problemler P1, P2, ..., P10 olarak adlandırılmıştır. Çözülen tüm operasyonel SİÇ problemlerine ait sonuçlar Tablo 2'de sunulmaktadır.

Tabloda P0 iki makineli alt probleme, P1 bir makineli alt probleme, P10 ise on makineli alt probleme karşılık gelmektedir. "Amaç fonksiyonu değeri (AFD)" sütunu alt problemlere ait toplam kar değerlerini, "AFD Toplanmış" sütunu ise ilgili alt probleme ait toplam kar değerinin P0 alt probleminin toplam kar değeri ile toplanmış halini göstermektedir. Makine başına marjinal katkı her ek makinenin (yani her ek kapasite seviyesinin) getirdiği ekstra kar değeri olarak hesaplanmıştır. "Çizelgelenememiş İş Sayısı" sütununda ise ilgili alt problemdeki kapasite seviyesi ile işlenemeden kalan işlerin sayısı verilmektedir.

Tablo 2. Operasyonel problem çözümleri.

Problem No	***Amaç Fonksiyonu Değeri (AFD)***	***AFD Toplanmış***	***Makine başına Marjinal Katkı***	***Çizelgenenememiş İş Sayısı***
P0	2086	2086	1043	117
P1	653	2739	653	91
P2	1081	3167	428	69
P3	1435	3521	354	50
P4	1707	3793	272	33
P5	1868	3954	161	22
P6	1976	4062	108	12
P7	2030	4116	54	7
P8	2059	4145	29	3
P9	2082	4168	23	1
P10	2084	4170	2	0

Tablo 2'de açıkça görülebileceği üzere makine başına marjinal katkı kapasitenin artmasıyla birlikte beklenen bir düşüş göstermektedir. Çizelgelenememiş işlerin sayısı ilk eklenen makinelerde hızla azalırken sonraki kapasite artırımlarının ekonomik açıdan ciddi bir fayda sağlanmadığı gözlenmektedir. Aynı durum grafik olarak Şekil 3'de gösterilmiştir.



Şekil 3. Kapasite seviyeleri ve marjinal katkılar.

Bir önceki bölümde anlatılan ve bu bölümde deneysel uygulaması sunulan çözüm yaklaşımı karar verici açısından çok değerli bir araç oluşturmaktadır. Her bir kapasite seviyesi için oluşabilecek maksimum kar seviyeleri optimal olarak elde edilmiştir. Çözüm sürelerinin ihmal edilebilir derecede kısa olduğu gözlenmiştir; tüm algoritmanın çalışma süresi bir saniyenin altında kaldığından tek tek alt problemlerin çözüm süreleri sunulmamıştır. Karar verici kapasite artırım kararını Tablo 2 ve Şekil 3 yardımıyla bakarak rahatça verebilecektir. Bu karar alındığı anda eklenecek yeni makinelerin çizelgeleri de otomatik olarak belirlenmiş olacaktır. İlgili toplam kar seviyesi de AFD Toplanmış sütunundan okunabilir. Bu örnekte 5 makine veya fazlasını ekleme şeklinde gerçekleştirilecek kapasite artırımı seviyelerinin ekonomik açıdan getirisi olmayacağı rahatlıkla söylenebilir.

Bu örnekte kapasite artırımı, kapasite planlama ve SİÇ kararlarının bütünleşik olarak alınabilmesi için bir çözüm yaklaşımı sunulmuştur. Talep düşmesi durumunda alınabilecek kapasite eksiltme kararları için de aynı yaklaşım kolaylıkla adapte edilerek polinom zamanlı bir çözüm verir. Bu durumda makine sayısı için üst sınır eldeki kapasite seviyesi olarak belirlenmeli, makine sayısı birer birer azaltılarak alt problemler çözülmelidir. Makine sayısı için doğal alt sınır 1 olacağından toplamda çözülen alt problem sayısı eldeki makine sayısı ile sınırlı olacak, taktik problemin çözümüne ihtiyaç duyulmayacaktır.

4.1.3. Özel Durum: Tek Makineli Temel SİÇ Problemi

Bölüm 4.1.1'de sunulan ve Bölüm 4.1.2'de sayısal olarak örneklendirilen yaklaşım problemin daha özel bir hali olan tek makineli temel SİÇ problemine de kolaylıkla uygulanabilir. Burada iki ayrı durum söz konusudur.

İlk Çözümü Bilinen Durum:

Bu özel durumda tek makineli bir sistemde ilk çözümü bilinen bir operasyonel SİÇ problemi ele alınmakta, aynı problemin iki makineli sistemdeki çözümünün eldeki çözüme bağlı olarak türetilmesi amaçlanmaktadır.

Bu durumda kesin çözüm yaklaşımı çizelgelenememiş işler ile tek makinelik bir operasyonel problem çözümünü içermektedir. Bu da bir en kısa yol problemine karşılık gelmektedir.

Dolayısıyla $O(n)$ zamanda tek bir problem için kesin çözüm elde ederek bütünleşik kararı vermek mümkündür (Solyalı ve Özpeynirci, 2009).

Eğer tek makineli sistemde birden fazla makine ekleme durumu söz konusu ise önceki bölümde anlatılan yöntem aynen kullanılabilir. Bir başka alternatif olarak eklenen her makinedeki çizelgeyi sabitlemek suretiyle alt problemlerde MMAA çözümü yerine hep tek makine artırımı yaparak en kısa yol çözümü bulmaktır. Fakat bu alternatifin kesin çözüm vermeyeceği, sezgisel bir yaklaşım olduğu unutulmamalıdır. Bu durumda her iki yaklaşım da polinom zamanlı olduğundan MMAA çözümü içeren kesin çözüm yaklaşımı rahatlıkla tercih edilebilir.

İlk Çözümü Bilinmeyen Durum:

Bu özel durum için tek makineli bir sistemde çözümü bilinmeyen bir operasyonel SİÇ problemi söz konusudur. Bu durumda temel operasyonel problemi bir kez çözerek (en kısa yol çözümüyle) optimal çizelge elde edilir. Daha önce olduğu gibi eldeki makinelerle işlenebilen (çizelgelenmiş) işlerin kümesi S ile ifade edilirse $n - |S|$ iş işlenemiyor (çizelgelenememiş) demektir. Bundan sonraki çözüm yaklaşımı Bölüm 4.1.1'de anlatılan ile aynı olacaktır.

4.2. BÜTÜNLEŞİK TEMEL SİÇ PROBLEMİ

Temel taktik SİÇ probleminin en genel amaç fonksiyonu makine sayısını değil makine maliyetini enazlamayı amaçlamaktadır. Temel operasyonel problemin en genel amaç fonksiyonu ise işlenebilecek iş kümesinin toplam ağırlığını enbüyüklemeyi amaçlar. Bu durumda temel SİÇ probleminin genel hali için bütünleşik kapasite ve çizelgeleme karar alınmasında kullanılabilecek bir matematiksel model bu iki amaç fonksiyonunu birleştiren bir model olmalıdır.

Önceki literatür incelendiğinde taktik ve operasyonel problemlerin tüm yazarlar tarafından ayrı ayrı çalışıldığı gözlemlenmiştir. Bu iki versiyonun özelliklerini birleştiren tek çalışma proje yürütücüsü tarafından projenin birinci döneminde yapılmış ve SCI tarafından taranan bir dergide yayımlanmıştır (Eliiyi, 2010a). Türkiye kaynaklı prestijli bir dergide basılan bu yayın projenin bir önceki bölümde sunulan ilk dönem çalışmalarından oluşmaktadır. Çalışmada makine maliyetlerinin eşit olduğu durum ele alınmıştır. Kapasite artırımı kararını verebilmek ve dolayısıyla optimal kapasite seviyesini belirleyebilmek için her bir ek özdeş makinenin kar katkısından yararlanılmakta, tekrarlı bir algoritma ile en uygun çözüm belirlenmektedir.

4.2.1. Bütünleşik Temel SİÇ için Matematiksel Model

Bütünleşik problem için projenin ikinci döneminde geliştirilen model operasyonel ve taktik modellerinin çözümünü aynı anda ele alarak hem toplam makine maliyetini hem de bu makinelerle işlenebilecek iş kümesinden elde edilebilecek toplam karı hesaplar. Dolayısıyla amaç fonksiyonunun makine maliyetleri düşüldükten sonraki net karı enbüyüklemeyi amaçladığı söylenebilir.

Modelde önceki bölümde tanımlanan karar değişkenleri kullanılmıştır. Bu değişkenlerle bütünleşik model aşağıdaki şekilde ifade edilir.

Bütünleşik Temel SİÇ (BTSİÇ) Modeli:

$$\text{Enb} \sum_{k=1}^{m}\sum_{j=1}^{n} w_j x_{jk} - \sum_{k=1}^{m} c_k y_k \qquad (8)$$

$$\sum_{k=1}^{m} x_{jk} \le 1 \qquad j = 1,...,n \qquad (2)$$

$$\sum_{j \in P_a} x_{jk} \le 1 \qquad k = 1,...,m \qquad \forall a \qquad (3)$$

$$x_{jk} \in \{0,1\} \qquad k = 1,...,m \qquad j = 1,...,n \qquad (4)$$

$$x_{jk} \le y_k \qquad k = 1,...,m \qquad j = 1,...,n \qquad (6)$$

$$y_k \in \{0,1\} \qquad k = 1,...,m \qquad (7)$$

Modelin amaç fonksiyonu (8) yukarıda açıklandığı gibi işlenebilecek iş kümesinin kar toplamlarından kullanılan makine maliyetlerini çıkarmak suretiyle bulunan net karı enbüyüklemektedir. (2) numaralı kısıt her işin en fazla bir makine tarafından işlenmesini sağlamakta, (3) numaralı kısıt her makinenin en fazla bir işi işlemesini temin etmekte, (4) numaralı kısıt da ikili değişkenleri tanımlamaktadır. İkinci modelde ek olarak gelen (6) numaralı kısıt iki karar değişkenini birbiriyle ilintilendirir ve kullanılmayan bir makinenin herhangi bir işi işlenmesini engeller. (7) numaralı kısıt ise ikili y değişkenini tanımlar. Bu modeldeki m sayısı makine sayısı için bir üst sınır (örneğin $m = n$) olarak belirlenmiş bir parametredir. Bu üst sınıra bir sonraki bölümde tekrar değinilecektir.

4.2.2. Problemin İşlemsel Karmaşıklığı

Bu bölümde yukarıda tanımlanan BTSİÇ modelinin işlemsel karmaşıklığı araştırılmıştır.

Üçüncü bölümde belirtildiği gibi temel operasyonel SİÇ probleminin polinom zamanda çözümü mümkündür. Bouzina ve Emmons (1996) problemin $n+1$ düğüme (node) ve $2n$ çizgiye (arc) sahip bir Minimum Maliyetli Ağ Akış Problemi (MMAA) olarak modellenebileceğini ve $O(mn\,logn)$ sürede çözülebileceğini göstermişlerdir. Temel taktik SİÇ problemi için makine sayısını enazlayacak optimal çözüm ise zaman ekseninde işlerin maksimum çakışma sayısının bulunmasıyla $O(n)$ zamanda hesaplanabilir (Hashimoto ve Stevens, 1971). Matematiksel olarak maksimum çakışma sayısı;

$Maks_a\{|P_a|\}$,
$|P_a|$: P_a kümesinin eleman sayısı,

olarak tanımlanır. Eliiyi (2004) ise $c_k \neq c$ durumundaki optimal çözümün, yani temel taktik SİÇ'de toplam makine maliyetini enazlayacak optimal çözümün $O(n\,logn)$ zamanda bulunabileceğini göstermiştir. Bu çözüm yönteminde öncelikle Hashimoto ve Stevens (1971) çözümü ile tüm işleri işleyebilecek minimum makine sayısı $Maks_a\{|P_a|\}$ olarak hesaplanmakta, daha sonra tüm makineler arasından en düşük maliyete sahip $Maks_a\{|P_a|\}$ adedi seçilmektedir. En düşük toplam maliyetli çözüm bu makinelerden oluşur.

Aşağıda geliştirilen algoritma BTSİÇ modelinin optimal çözümünün, MMAA çözümü ve Eliiyi (2004)'te geliştirilen fikri kullanarak polinom zamanda bulunabileceğini göstermektedir.

Algoritma(BTSİÇ):

(S0) $Z_O = 0$, $Z_{Ok} = 0$,
$Z_T = 0$, $Z_{Tk} = 0$,
$Z = Z_O - Z_T = 0$, $Z_k = 0$,
Sisteme dahil edilmeye aday m makineyi maliyetlerine (c_k) göre artan sırada indisle, yani $c_1 \le c_2 \le ... \le c_m$ olsun.

(S1) Makine sayısının $k = 1,..., m$ arasındaki tüm değerleri için:
k makine ve n işten oluşan bir sistem için MMAA çözümü bul,
$Z_{MMAA(k)}$: MMAA amaç fonksiyonu değeri olsun,
$X_{MMAA(k)}$: MMAA çözümünde işlenebilen iş kümesi olsun,
$Z_{Ok} = Z_{MMAA(k)}$,
$Z_{Tk} = \sum_{i=1}^{k} c_i y_i$,
$Z_k = Z_{Ok} - Z_{Tk}$,
Eğer $Z_k < Z_{k-1}$ ise
$s = k-1$,
(S2) adımına git,
Değilse
$s = k$,
devam et.

(S2) Optimal çözüm :
$Z = Z_s$,
k^*=s, ve
İşlenebilen optimal iş Kümesi: $X_{MMAA(k^*)}$ olarak bulunur.

Algoritma(BTSİÇ) öncelikle (S0) adımında sisteme dahil edilmeye aday tüm makineleri maliyetlerine göre artan şekilde sıralamaktadır. Daha sonra (S1) adımında makineler bu sırada ele alınmakta, sıradaki makine ile birlikte önceki tüm aday makineler ve n adet iş için bir MMAA çözümü bulunarak operasyonel problem optimal olarak çözülmektedir. Ele alınan makineyle birlikte toplam makine maliyetinin işlenebilecek iş kümesinin toplam getirisinden fazla veya $Z_k < Z_{k-1}$ olduğu durumda algoritma bir sonraki adıma geçmekte, aksi halde çözümü bir sonraki makine ile birlikte denemektedir. Bir sonraki adımda ise optimal çözüm ele alınan tüm makine adayları arasında en yüksek net getiriye sahip makine kümesi ve bunların işleyebildiği işler olarak bulunmaktadır.

Algoritmanın (S1) adımındaki durma koşulu ((S2) adımına geçme), yani $Z_k < Z_{k-1}$ koşulu, herhangi bir adımdaki amaç fonksiyonu değeri bir önceki adımda bulunandan daha düşük ise durmayı gerektirmektedir. Makine maliyetleri artan sırada olduğundan bu durma koşulu anlamlıdır. Koşulun "$Z_k \le Z_{k-1}$" yerine "$Z_k < Z_{k-1}$" şeklinde tanımlanması ise k. makinenin marjinal katkısının $(k-1)$. makinenin marjinal katkısına eşit olduğu durumda, yani başabaş noktasında da kapasite artırımı yapmayı desteklemektedir. Herhangi bir üretici veya hizmet sağlayıcının daha fazla müşteriye hizmet edebilmek ve ileriye dönük talep yaratabilmek amaçlı olarak başabaş noktasında da hizmet vermeye devam etmek isteyeceği varsayımıyla koşul bu şekilde kullanılmıştır. Bu kullanımın sonucu olarak kapasite seviyesi artarak daha fazla iş işlenecektir. Sayısal deney sonuçlarında bu kullanımın yarattığı alternatif optimal çözümlerin özelliklerine tekrar değinilecektir.

Sisteme aylak makine yaratmadan eklenebilecek makine sayısı için yukarıda anlatıldığı üzere $Maks_a\{|P_a|\}$ sayısı bir doğal üst sınır oluşturmaktadır. Dolayısıyla BTSİÇ modelinde ve algoritmanın (S0) adımında yer alan m sayısı $m = Maks_a\{|P_a|\}$ şeklinde belirlenebilir. Ancak

pratikteki problemlerde alınabilecek/kiralanabilecek makinelerin - veya daha genel ifadeyle kaynakların – toplam sayısı her zaman bu rakama ulaşmayabilir. Başka bir deyişle, sistemin kapasite artırımında kullanılabilecek kaynakların sayısı her zaman tüm işlerin işlenmesine yetmeyebilecek, bu konuda ek bir kısıt oluşabilecektir. Bu durumda m sayısı kapasite artırımında kullanılabilecek sisteme dahil edilmeye aday makine sayısını temsil edecektir. Dolayısıyla bu üst sınırın karar verici veya dış faktörler tarafından belirlenmesi mümkündür.

Temel SİÇ probleminde makineler özdeş ve paraleldir. Eklenecek makinelerin maliyetlerinin farklı oluşu tedarikçi farkıyla açıklanabilir.

Teorem 1: Algoritma(BTSİÇ) BTSİÇ modelinin optimal çözümünü $O(m^2 n\, logn)$ zamanda bulur.

İspat : Algoritmanın (S0) adımındaki sıralama işlemi en kötü ihtimalle $O(logn)$ zamanda yapılabilir. Bunun sebebi iş sayısının olabilecek makine sayısı için en büyük üst sınır olmasıdır (böyle bir durumda her iş bir makineye atanacaktır). Sonraki adımda Algoritma(BTSİÇ) makine sayısı alt sınırı olan 1 makineden en kötü ihtimalle üst sınır olan m makineye kadar tüm olası makine sayıları için operasyonel temel SİÇ probleminin optimal çözümünü $O(mn\, logn)$ zamanlı MMAA ile bulmakta, bu durumda (S1) adımının zamansal karmaşıklığı $O(m^2 n\, logn)$ olarak gerçekleşmektedir. Algoritma bu adımda ayrıca her makine sayısı için toplam makine maliyetini tek adımda hesaplayabilmektedir. (S1) adımındaki $Z_k < Z_{k-1}$ kontrolü ise aslında $Z_{Ok} - Z_{Ok-1} < c_k$ kontrolüne, yani son eklenen k. makinenin operasyonel problemin çözüm değerine katkısının maliyetinden fazla olup olmadığını kontrol etmektedir.

Dolayısıyla Algoritma(BTSİÇ) problem için olası tüm çözümleri gezerek içlerinde en iyisini seçtiğinden optimal çözüme ulaşması kesindir. Herhangi bir adımda $Z_k < Z_{k-1}$ eşitsizliği sağlandığında, makineler maliyetlerine göre artan şekilde sıralandığından sonraki makineler için hesaplama yapmaya gerek duyulmamaktadır. Zaman karmaşıklığı ise algoritmanın en zor adım olan (S1) adımı tarafından $O(m^2 n\, logn)$ olarak belirlenmektedir. □

Dolayısıyla Algoritma(BTSİÇ) takip eden sayfada verilen Tablo 3'ü doğru şekilde dolduracaktır. Tablodaki durum olabilecek en fazla hesaplamanın yapıldığı durumu temsil etmektedir.

4.2.3. Özel Durum: İlk Çözümü Bilinmeyen BTSİÇ

Şu ana kadar açıklanan BTSİÇ modeli ve geliştirilen algoritma sistemin kapasite artırımından önceki çizelgesini sabitlenmiş olarak ele almaktadır. Başka bir deyişle modeldeki n adet iş sistemde eldeki makinelerle işlenemeyen, kapasite artırımı sonrasında işlenmeye aday işlerden oluşmaktadır. Aynı şekilde m adet makine ise kapasite artırımı için aday makineleri temsil etmektedir. Dolayısıyla sistemin şu anki durumu ve ilgili çizelgesinin yani ilk çözümünün bilindiği ve değiştirilmediği durum ele alınmıştır. Bu varsayımın gerçekçi olduğu düşünülmektedir, zira kapasite artırım kararına kadar yürütülen çizelgelerin sabitlenmesi genelde istenen bir durum olabilir.

Tablo 3. Tüm makine sayıları için BTSİÇ çözümleri.

Makine sayısı	***Operasyonel Problemin Amaç Fonksiyonu Değeri (Z_{Ok})***	***Taktik Problemin Amaç Fonksiyonu Değeri (Z_{Tk})***	***BTSİÇ Probleminin Amaç Fonksiyonu Değeri (Z_k)***
0	0	0	0
. .			
k	$Z_{MMAA(k)}$	$\sum_{i=1}^{k} c_i y_i$	$Z_k = Z_{Ok} - Z_{Tk}$
. .			
Üst sınır = m	Eğer $m \geq Maks_a\{\|P_a\|\}$ ise $\sum_{j=1}^{n} w_j$, değilse $Z_{MMAA(m)}$.	$\sum_{i=1}^{m} c_i y_i$	$Z_k = Z_{Ok} - Z_{Tk}$
$k^* = \underset{k}{ArgMax}\{Z_k\}$	$Z_{MMAA(k^*)}$	$\sum_{i=1}^{k^*} c_i y_i$	$Z = \underset{k}{Max}\{Z_k\}$

İlk çözümün bilinmediği durumda ise sistemde varolan makine sayısı (m') ile çizelgesi sabitlenmemiş tüm işleri probleme katarak (bu durumda aday n adet işe daha önceden planı yapılmış fakat çizelgesi sabitlenmemiş işler de katılacaktır; yeni iş sayısına n' diyelim) ilk operasyonel çözümü bulmak gerekmektedir. Bu ilk problemin optimal çözümü için fazladan bir kez MMAA çözülmesi yeterli olacaktır. Optimal çözümde eldeki m' makine ile işlenebilen işlerin kümesi S ile ifade edilirse $n' - |S|$ iş işlenemiyor demektir.

Bu aşamadan sonra ise problemin geri kalanı Algoritma(BTSİÇ)'nın $n' - |S|$ iş ve m aday makine ile çalıştırılması yoluyla çözülecektir. Dolayısıyla ilk çözümü bilinmeyen durum için optimal çözüm algoritmasının işlem karmaşıklığı Algoritma(BTSİÇ) ile aynı olacaktır; fazladan çözülen bir MMAA problemi işlem karmaşıklığını etkilemeyecektir.

4.2.4. BTSİÇ için Örnek Problem

Bu bölümde Algoritma(BTSİÇ) çalışma prensiplerinin daha iyi anlaşılabilmesi açısından bir örnek problem üzerinde gösterilmiştir. Tablo 4'de özetlenen bu örnek bir sonraki bölümde sunulacak sayısal deneyde kullanılan veri setinden seçilmiş 200 işlik bir problemdir. Algoritmanın (S0) adımında yer alan aday makine sayısı olarak $Maks_a\{|P_a|\}$ üst sınırı kabul edilmiştir. Bu problem örneği için 15 makineye eşittir ve artan c_k maliyetleri sırasıyla 40, 40, 40, 40, 40, 40, 50, 50, 70, 70, 70, 80, 80, 80 ve 80'e eşittir.

Aşağıdaki tabloda algoritmanın adımları ve ilgili sonuçlar özetlenmiştir.

Tablo 4'de belirtildiği gibi Algoritma(BTSİÇ)'in 10. iterasyonunun (S1) adımındaki kontrolle sona erdiği görülmektedir. Dolayısıyla bu problem için kalan 5 makinenin getirilerinin hesaplanmasından, dolayısıyla 5 adet MMAA problemi çözüm zamanından tasarruf edilmiştir. 9 makinelik bu optimal çözümle 200 işin 181'i işlenmiş, potansiyel toplam iş getirisinin %92'si elde edilmiştir.

Tablo 4. Örnek problemdeki tüm aday makine sayıları için BTSİÇ çözümleri.

Makine sayısı	***Operasyonel Problemin Amaç Fonksiyonu Değeri (Z_{Ok})***	***Taktik Problemin Amaç Fonksiyonu Değeri (Z_{Tk})***	***BTSİÇ Probleminin Amaç Fonksiyonu Değeri (Z_k)***
0	0	0	0
1	204	40	164
2	399	80	319
3	583	120	463
4	757	160	597
5	909	200	709
6	1032	240	792
7	1139	290	849
8	1225	340	885
9	1296	410	886
10	1344	480	*864* (Z_{10} - Z_{11} = -12 < 0)
11	1373	550	823
12	1386	630	756
13	1399	710	689
14	1405	790	615
Üst sınır = 15	1410	870	540
k^*=9	$Z_{MMAA(k^*)}$=1296	$\sum_{i=1}^{9} c_i y_i = 410$	Z=Z_k=886

4.2.5. Sayısal Deney

Projenin ikinci döneminde BTSİÇ modelinin optimal çözümü için geliştirilen Algoritma(BTSİÇ)'nin performansı kapsamlı olarak sayısal deneylerle test edilmiştir. Bu bölümde deney tasarımı ve deneylerden elde edilen sayısal sonuçlar açıklanmaktadır.

Deney Tasarımı:

Sayısal deneyin tasarımında öncelikle probleme ait parametre değerleri için çeşitli seviyeler belirlenmiştir. Problemin genelliğini bozmadan bütün parametre değerlerinin tamsayı oldukları varsayılmıştır. Problemdeki iş sayıları için $n = 20, 50, 100, 200, 500$ değerleri ele alınmıştır. Burada 20 iş küçük bir probleme, 500 iş ise oldukça büyük bir probleme karşılık gelmektedir. Bu n iş daha önce açıklandığı gibi halihazırda sistemde bulunan kapasite ile işlenebilecek işleri değil, yeni alınacak makineler tarafından işlenmeye aday iş kümesini temsil etmektedir. Yani sistemin şimdiki kapasite seviyesiyle (makine sayısıyla) ilk çözümünün bilindiği varsayılmıştır.

İşlerin sisteme girme yani varış zamanları [0, 200] arasında düzgün dağılacak şekilde türetilmiştir ($r_j \quad U(0,200)$). İşlerin işlem sürelerinin [4,10] değerleri arasında kesikli düzgün dağıldığı varsayılmış ve buna göre türetilmiştir ($p_j \quad U(4,10)$).

İş ağırlıkları için üç parametre değer seviyesi belirlenmiştir. Buna göre:

- $w_j = 1$ parametre değer seviyesinde her işin getirisi kendi işlem süresine eşittir ($w_j = p_j, \forall j.$). Bu seviye bir işin getirisinin işlenme süresiyle orantılı olduğu durumları temsil etmek için tasarlanmıştır.

- $w_j = 2$ parametre değer seviyesi getirilerin (ağırlıkların) işlem süreleriyle aynı dağılımdan geldiği rassal durumu temsil etmektedir ($w_j \quad U(4,10)$). Bu durum düşük varyanslı bir rassal dağılıma karşılık gelmektedir.

- $w_j = 3$ parametre değer seviyesi ise getirilerin $w_j \quad U(4,20)$ dağılımından geldiği rassal durumu temsil etmektedir. Bu durum yüksek varyanslı bir rassal dağılıma karşılık gelmektedir.

Deney tasarımında makine maliyetleri dört farklı şekilde ele alınmıştır. Buna göre:

1. $c_k \quad U\{40,50,60,70,80\}$
2. $c_k = 40, \forall k$
3. $c_k = 60, \forall k$
4. $c_k = 80, \forall k$

olmak üzere dört seviye belirlenmiştir. İlk seviyede makine maliyetleri $\{40,50,60,70,80\}$ kümesi içinden düzgün dağılacak şekilde rassal olarak türetilmiştir. Olabilecek en düşük makine maliyeti olabilecek en düşük iş getirisinin 10 katı olacak şekilde belirlenmiştir. Özdeş makinelerin maliyetleri arasında tedarikçi farkı vb. nedenlerden dolayı kaynaklanabilecek farkın ise maksimum %100 olacağı varsayılarak en yüsek makine maliyeti en düşük maliyetin 2 katı olarak belirlenmiştir. Ara değerlerle beraber 5 farklı maliyet oluşmaktadır. İkinci parametre seviyesi tüm makine maliyetlerinin en düşük seviyede olduğu durumu temsil ederken üçüncü ve dördüncü seviyeler sırasıyla orta ve yüksek maliyet düzeylerine karşılık gelmektedir.

Yukarıda açıklanan deneysel tasarımda toplam 5 iş sayısı, 3 ağırlık ve 4 maliyet parametre seviyesi, dolayısıyla 60 farklı seviye bulunmaktadır. Her bir seviyeden 10 örnek problem türetilmiş, dolayısıyla toplamda 600 örneklik bir sayısal deney gerçekleştirilmiştir.

Sayısal deneyler 2 çekirdekli, 4 GB Ram ve 2.8 GHz işlemciye sahip Windows 7 işletim sistemli bir PC üzerinde gerçekleştirilmiştir. Algoritma(BTSİÇ)'nın kodlanmasında MS Visual Studio 2008 ortamında C# ve C++ programlama dilleri kullanılmış, MMAA probleminin çözümü için IBM ILOG CPLEX 12.1 solver kütüphanesinden yararlanılmıştır. Matematiksel modelin kodlanmasında ve çözümünde ise yine CPLEX 12.1 solverla bütünleşik ILOG OPL 6.3 model programlama dili kullanılmıştır.

Sonuçlar:

Açıklanan deney tasarımı sonrasında türetilen problem örnekleri Algoritma(BTSİÇ) kullanılarak çözülmüştür. Ayrıca BTSİÇ probleminin matematiksel modeli IBM ILOG OPL Versiyon 6.3 ortamında kodlanarak CPLEX 12.1 solver ile çözdürülmüş, sonuçlar karşılaştırılmıştır.

Sayısal deneyin sonuçları makine maliyetinin önceki bölümde açıklanan dört farklı seviyesi için birer tabloda (Tablo 5 - Tablo 8) gösterilmiştir. Buna göre Tablo 5 düzgün dağılan makine maliyetleri için deney sonuçlarını göstermektedir.

Tablolardaki ilk sütun problemdeki iş sayısını, ikinci sütun ise önceki bölümde açıklanan farklı getiri seviyelerini göstermektedir. "*Ort. m. UB"* olarak adlandırılan sütun ilgili parametre seviyesindeki 10 problem örneği için hesaplanmış makine üst sınırlarının ortalamasını göstermektedir. Bu üst sınır her bir problem örneği için $Maks_a\{|P_a|\}$ olarak hesaplanmakta ve optimal çözümde olabilecek maksimum makine sayısını belirlemektedir.

"Kull. ort. m. sayısı" sütunları Algoritma(BTSİÇ) çözümü ve modelin CPLEX çözümü için 10 problem örneğinde optimal çözümde kullanılan ortalama makine sayılarını vermektedir. Örneğin Tablo 5'de 20 işlik ve $w = 1$ parametre seviyesindeki 10 problem örneğinde kullanılabilecek maksimum makine sayısı 4 iken hem Algoritma(BTSİÇ) hem de CPLEX çözümlerinde optimal çözümde kullanılan makine sayısı 1 olarak gerçekleşmiştir. Bu durum herhangi bir örnek problemde ikinci bir makinenin amaç fonksiyonunu düşüreceğine işaret etmektedir. Yani ikinci en düşük maliyetli makinenin işleyebileceği işlerin toplam ağırlığı o makinenin maliyetinden daha düşüktür. Tablonun kimi satırlarında iki çözüm arasında makine kullanım sayılarında küçük farklar göze çarpmaktadır. Bu durum alternatif optimal çözümler arasındaki farklardan kaynaklanmaktadır.

"Ort. m. kullan. %" sütunları Algoritma(BTSİÇ) çözümü ve modelin CPLEX çözümü için 10 problem örneğinde optimal çözümde gerçekleşen ortalama makine kullanım (yararlanma, *utilization*) yüzdelerini vermektedir. Bu yüzde değerler makinenin planlama periyodunun, yani [0,200] aralığının ne kadarında işlem yaptığını göstermektedir. Örneğin Tablo 5'de 20 işlik ve $w = 2$ parametre seviyesindeki 10 problem örneğinde Algoritma(BTSİÇ) çözümünde kullanılan ortalama 1.1 makine için ortalama kullanım %40.1, CPLEX çözümünde ise %39.8'dir.

"İşlenen işler %" sütunları iki çözümde işlenmeye aday n adet işten sayı bazında yüzde kaçının işlendiğini (10 problem için ortalama) belirtmektedir. Örneğin Tablo 5'de 20 işlik ve $w = 2$ parametre seviyesindeki 10 problem örneğinde her iki çözümde de işlerin ortalama %67'si, yani 13.4 adet iş işlenmiştir.

"İşlenen ağırlık %" sütunları ise iki çözümde işlenmeye aday n adet işten getiri bazında yüzde kaçının işlendiğini (10 problem için ortalama) belirtmektedir. Bu yüzde değerler işlenen işlerin toplam ağırlığının sistemdeki n adet işin toplam ağırlığına bölünmesiyle bulunmaktadır. Örneğin Tablo 3'de 20 işlik ve $w = 2$ parametre seviyesindeki 10 problem örneğinde her iki çözümde de sistemdeki toplam ağırlığın %70.8'i optimal kapasite seviyesiyle işlenebilmiştir. Bu değer potansiyel getirinin ne kadarının elde edilebildiğinin bir göstergesidir.

Son olarak tablolardaki *"Süre"* sütunları ise iki çözüm için ortalama çözüm süresini saniye cinsinden ifade etmektedir.

Tablolar incelendiğinde öncelikle göze çarpan nokta geliştirilen algoritmanın günümüzde en gelişmiş solver olan CPLEX 12.1 ile karşılaştırıldığında büyük bir süre avantajına sahip

olmasıdır. Algoritma küçük problemlerde CPLEX ile karşılaştırılabilir sürelerde, büyük problemlerde ise çok daha kısa sürelerde optimal çözüme ulaşmaktadır. Özellikle 500 işlik problemlerde bu durum açıkça görülmektedir. Dolayısıyla geliştirilen algoritma süre açısından çok etkin bir algoritmadır ve daha büyük problemlere de rahatlıkla uygulanabileceği görülmektedir. Makine maliyet ve iş ağırlık seviyelerinin problem çözüm süresini etkilemediği, çözüm süresinin iş sayısıyla arttığı gözlenmektedir.

Tablo 5. Düzgün dağılan makine maliyetleri için deney sonuçları. $c_k \quad \mathrm{U}\{40,50,60,70,80\}$

		Ort. m.	Kul. ort. m. sayısı		Ort. m. kullan. (%)		İşlenen işler (%)		İşlenen ağırlık (%)		Süre (sn.)	
n	*w*	UB	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX
20	1	4	1.0	1.0	44.4	44.4	61.5	61.5	63.5	63.5	0.1	0.1
	2	3	1.1	1.1	40.1	39.8	67.0	67.0	70.8	70.8	0.1	0.1
	3	3	1.4	1.4	37.1	37.1	73.5	73.5	80.0	80.0	0.1	0.1
50	1	6	2.5	2.4	55.6	56.8	74.8	73.0	78.3	76.8	0.3	0.1
	2	5	2.5	2.5	54.2	53.8	76.2	76.0	80.3	80.3	0.3	0.1
	3	6	3.1	3.1	47.0	46.9	85.2	85.2	90.4	90.4	0.3	0.1
100	1	9	4.5	4.5	66.4	66.4	82.1	82.0	84.0	84.0	0.4	0.2
	2	9	4.5	4.5	63.5	63.5	83.4	83.5	87.5	87.5	0.4	0.1
	3	9	5.1	5.1	60.2	60.1	89.8	89.8	93.8	93.8	0.4	0.1
200	1	14	8.5	8.5	74.2	74.2	88.2	88.2	90.1	90.1	0.8	0.6
	2	14	8.2	8.2	73.3	73.3	87.1	87.1	90.8	90.8	0.8	0.6
	3	14	8.9	8.9	70.5	70.5	90.3	90.3	94.6	94.6	0.8	0.6
500	1	28	19.7	19.6	83.0	83.3	92.4	92.3	94.1	93.9	2.9	46.1
	2	28	18.4	18.4	84.5	84.6	89.9	89.9	93.2	93.2	2.8	44.6
	3	29	19.9	19.9	80.2	80.3	92.1	92.1	96.2	96.2	3.0	46.1

Tablo 6. Düşük seviyeli makine maliyetleri için deney sonuçları. $c_k = 40, \forall k$

		Ort. m.	Kul. ort. m. sayısı		Ort. m. kullan. (%)		İşlenen işler (%)		İşlenen ağırlık (%)		Süre (sn.)	
n	*w*	UB	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX
20	1	3	1.5	1.5	41.1	41.1	76.0	76.0	78.3	78.3	0.1	0.1
	2	3	1.2	1.2	44.6	44.7	73.0	73.0	75.3	75.3	0.1	0.1
	3	4	1.9	1.9	32.2	32.2	85.5	85.5	89.5	89.5	0.1	0.1
50	1	6	3.0	3.0	51.8	51.8	86.8	86.8	89.0	88.9	0.3	0.1
	2	5	3.0	3.0	48.8	48.7	86.4	86.4	89.3	89.3	0.3	0.1
	3	5	3.6	3.6	44.8	44.7	91.8	91.8	94.6	94.6	0.4	0.1
100	1	9	5.5	5.5	60.1	60.1	91.8	91.8	93.6	93.6	0.4	0.2
	2	8	5.3	5.3	58.5	58.4	90.6	90.6	93.3	93.3	0.4	0.2
	3	9	6.1	6.1	53.5	53.6	93.8	93.8	96.5	96.5	0.5	0.2
200	1	15	9.8	9.8	66.9	66.9	92.6	92.4	94.0	94.0	0.9	1.1
	2	14	9.4	9.4	68.7	68.7	92.3	92.3	94.5	94.5	0.8	0.9
	3	13	10.0	10.0	65.9	65.9	95.6	95.6	97.8	97.8	0.9	0.8
500	1	29	21.8	21.8	77.6	77.6	95.7	95.7	96.8	96.8	3.0	68.3
	2	30	21.2	21.1	78.0	78.3	94.1	94.0	96.2	96.1	3.0	98.1
	3	28	22.3	22.3	76.0	76.0	96.5	96.5	98.4	98.4	3.2	51.2

Tablo 7. Orta seviyeli makine maliyetleri için deney sonuçları. $c_k = 60, \forall k$

		Ort. m.	Kul. ort. m. sayısı		Ort. m. kullan. (%)		İşlenen işler (%)		İşlenen ağırlık (%)		Süre (sn.)	
n	*w*	UB	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX
20	1	4	1.0	1.0	43.2	43.2	58.5	58.5	59.6	59.6	0.1	0.1
	2	4	1.0	1.0	40.8	41.0	59.5	59.5	63.3	63.3	0.1	0.1
	3	3	1.6	1.6	35.6	35.6	80.0	80.0	84.8	84.8	0.1	0.1
50	1	5	2.5	2.5	58.1	58.1	79.0	78.6	81.6	81.6	0.3	0.1
	2	6	2.4	2.4	54.6	54.6	75.2	75.2	80.0	80.0	0.3	0.1
	3	6	3.1	3.1	48.8	48.8	87.8	87.8	92.3	92.3	0.3	0.1
100	1	9	5.0	5.0	62.2	62.2	87.0	87.0	89.2	89.2	0.4	0.2
	2	9	4.5	4.5	63.6	63.6	82.7	82.8	86.7	86.7	0.4	0.2
	3	9	5.4	5.4	57.5	57.5	88.9	88.9	93.9	93.9	0.4	0.2
200	1	14	9.0	9.0	71.2	71.2	89.1	89.1	91.3	91.3	0.8	1.1
	2	13	8.4	8.4	72.7	72.6	88.0	88.0	91.6	91.6	0.8	0.9
	3	14	9.0	9.0	70.9	70.8	91.5	91.5	95.3	95.3	0.8	1.0
500	1	28	20.0	19.9	82.4	82.7	93.2	93.0	94.8	94.7	2.9	84.2
	2	30	18.8	18.7	82.4	82.7	89.9	89.7	93.2	93.1	2.9	93.2
	3	30	21.0	20.9	78.3	78.6	94.0	93.8	97.0	96.9	3.1	85.9

Tablo 8. Yüksek seviyeli makine maliyetleri için deney sonuçları. $c_k = 80, \forall k$

		Ort. m.	Kul. ort. m. sayısı		Ort. m. kullan. (%)		İşlenen işler (%)		İşlenen ağırlık (%)		Süre (sn.)	
n	*w*	UB	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX
20	1	3	0.9	0.9	47.1	47.1	57.5	57.5	59.4	59.4	0.1	0.1
	2	4	0.9	0.9	42.5	42.8	56.5	56.5	59.8	59.8	0.1	0.1
	3	3	1.1	1.1	41.2	41.1	65.0	65.0	72.1	72.1	0.1	0.1
50	1	6	2.0	2.0	60.4	60.4	67.2	67.2	69.3	69.3	0.3	0.1
	2	6	1.9	1.9	57.6	57.6	64.0	63.8	68.6	68.6	0.3	0.1
	3	6	2.6	2.6	52.2	52.2	77.8	77.8	84.8	84.8	0.3	0.1
100	1	8	4.0	4.0	70.3	70.3	78.1	78.2	81.1	81.1	0.3	0.2
	2	8	3.8	3.8	66.1	65.9	75.4	75.3	80.9	80.9	0.3	0.2
	3	8	4.7	4.7	62.5	62.4	85.1	84.9	91.0	91.0	0.4	0.2
200	1	15	7.8	7.8	78.4	78.4	85.1	85.4	87.2	87.2	0.8	1.7
	2	14	7.2	7.2	76.4	76.3	81.1	81.2	85.7	85.7	0.7	1.1
	3	14	8.7	8.7	71.2	71.2	88.7	88.6	94.0	94.0	0.8	1.2
500	1	28	19.1	18.9	85.0	85.5	90.4	89.9	92.4	91.9	2.8	84.0
	2	28	17.4	17.4	86.4	86.7	87.0	87.0	91.0	91.0	2.7	115.9
	3	27	19.4	19.3	82.0	82.2	91.6	91.3	96.1	95.9	3.0	76.8

Algoritma ile CPLEX'in ürettiği alternatif çözümler tablolardaki küçük farklılıkları oluşturmaktadır. Bu farklılıklar incelendiğinde algoritmanın ürettiği çözümlerin işlenen iş yüzdesi açısından biraz daha iyi olduğu söylenebilir. Örnekler tek tek incelendiğinde ise CPLEX ve Algoritma(BTSİÇ)'nın verdiği alternatif optimal çözümler arasındaki temel farkın algoritmanın (S1) adımındaki "$Z_k < Z_{k-1}$" koşulundan kaynaklandığı görülmüştür. Algoritmayla yaratılan optimal çözümler daha fazla işi işlemekte, bunu daha fazla makine ile yapmaktadır.

Makine maliyetlerinin artması beklenen şekilde kapasite artırımı kararını olumsuz yönde etkilemektedir. Bu durum tablolarda kullanılan ortalama makine sayılarının karşılaştırılmasıyla anlaşılabilir. Özellikle son tabloda 20 işlik problemlerin $w = 1$ ve $w = 2$ seviyelerine ait iki örneğinde optimal çözümde kapasite artırımı 0 olarak gerçekleşmiştir. Bu da makine maliyetinin o örneklerde türetilen iş getirilerine göre çok yüksek geldiği anlamını taşımaktadır. Makine maliyetlerinin artmasının diğer bir sonucu da kullanım oranlarının

yükselmesi şeklinde izlenmektedir. Maliyet arttıkça aynı makinede işlenen iş sayısı artmakta, makine kullanım oranları da yükselmektedir.

Makine kullanım oranları küçük problemlerde ortalama %40 seviyesinde iken büyük problemlerde %85'lere çıkmaktadır. Bu durum büyük problemlerde çakışma yapan çok sayıda iş olması ve bunların farklı makinelerde işlenmesi, dolayısıyla planlama periyodunda işlerin makineleri daha çok dolduracak şekilde planlanabilmesinden kaynaklanmaktadır. Aynı sebeple işlenen işlerin ve işlenen ağırlığın yüzdesi de büyük problemlerde daha fazla olmuştur.

İşlenen işlerin ve işlenen ağırlığın yüzdeleri karşılaştırıldığında işlenen ağırlığın daha fazla olduğu görülmektedir. Yeni eklenen makineler sayı olarak işlerin belli bir yüzdesini işleyebilmekte, fakat potansiyel getiri seviyesine bakıldığında daha yüksek bir yüzdeye erişmektedir. Bu durum $w = 1$ seviyesi için çok belirgin değildir çünkü bu seviyede işlerin ağırlıkları işlem sürelerine eşittir. Ancak diğer seviyelerde fark belirginleşmektedir; işlenebilecek iş alternatifleri arasından w_j/p_j oranı daha yüksek işler seçilmekte ve yüzdeler arasındaki fark bu sayede artmaktadır.

Ortalama makine kullanım oranlarının iş ağırlık parametresinin seviyelerine göre değişiklikler göstermekte, getirilerin varyansı arttıkça kullanım oranları azalmaktadır. İşlenen işlerin ve işlenen ağırlığın yüzdeleri ise getirilerin varyansı arttıkça belirgin bir şekilde artmaktadır. Bu iki durum birlikte değerlendirildiğinde getiri varyansı arttıkça işlenebilecek işler arasından yine w_j/p_j oranı daha yüksek olanların seçilmesi sayesinde daha az makine kulllanımı ile daha fazla getiri sağlandığı söylenebilir.

4.3. KAR SEVİYESİ KISITLI TAKTİK SABİT İŞ ÇİZELGELEME PROBLEMİ (KTSİÇ)

Projede üzerinde çalışılan bir diğer problem temel SİÇ probleminde işlerin getirilerinin (karlarının) belli bir seviyeyi aşmasının istendiği durumdaki taktik problem olmuştur. Bu amaçla kar seviyesi kısıtı altında kapasite kararlarının incelenmesi amaçlı bir matematiksel model geliştirilmiştir. Bu bölümde kar seviyesi kısıtlı bu taktik problem (KTSİÇ) incelenmiştir. Problemin genel hali ilk olarak ayrıntılı bir şekilde incelenecek, kar seviyesi kısıtının yüzde olarak belirlendiği ikinci bir model üzerinde ayrıca durulacaktır.

4.3.1. Kar Seviyesi Kısıtlı Taktik Problem

KTSİÇ probleminde amaç fonksiyonu taktik modeldeki gibi toplam makine maliyetini enazlamaktır. Ancak bunu yaparken model aynı zamanda işlenebilecek iş kümesinin toplam getirisinin de belli bir alt limitin (B) üzerinde olmasını gözetmektedir. Problemin geçerliliği özellikle kapasite artırımı planındaki yatırım maliyetinin karşılığında kar katkısı beklentisinin belirlenmiş olduğu sistemlerde yüksek olacaktır.

Problemin modellenebilmesi için yeni bir parametreye ihtiyaç duyulmaktadır.

B: Kar getirisi için alt sınır; kapasite artırımından beklenen minimum kar katkısı seviyesi. Burada problemin olurlu bir çözümünün olabilmesi için B limitinin sistemdeki tüm işlerin toplam ağırlığını aşmayacak şekilde belirlenmesi gerekmektedir. Yani,

$\sum_{j=1}^{n} w_j \geq B$ koşulu sağlanmalıdır.

Buna göre model aşağıdaki şekilde ifade edilir:

Kar Seviyesi Kısıtlı Taktik SİÇ (KTSİÇ):

$$\text{Enk} \sum_{k=1}^{m} c_k y_k \qquad (1)$$

$$\sum_{k=1}^{m} x_{jk} \leq 1 \qquad j = 1,...,n \qquad (2)$$

$$\sum_{j \in P_a} x_{jk} \leq 1 \qquad k = 1,...,m \qquad \forall a \qquad (3)$$

$$x_{jk} \in \{0,1\} \qquad k = 1,...,m \qquad j = 1,...,n \qquad (4)$$

$$x_{jk} \leq y_k \qquad k = 1,...,m \qquad j = 1,...,n \qquad (5)$$

$$y_k \in \{0,1\} \qquad k = 1,...,m \qquad (6)$$

$$\sum_{k=1}^{m} \sum_{j=1}^{n} w_j x_{jk} \geq B \qquad (7)$$

Modelde amaç fonksiyonu temel taktik problemin amaç fonksiyonu ile aynıdır. Eklenen (7) numaralı kısıt ise istenen kar getirisi seviyesinin altında kalmamayı garanti etmektedir. Dolayısıyla model beklenen bir kar seviyesine olası en düşük toplam makine maliyetiyle ulaşmayı hedeflemektedir. Modelde belirtilen B alt limiti eğer sistemdeki tüm işlerin ağırlıkları toplamına eşit ise problemin klasik TSİÇ problemine dönüştüğü burada not edilmelidir.

Kapasite planlaması ve kapasite artırımı için düşünülen yatırım planlarının fizibilite analizinde bu tip kar beklentilerinin belirlenmesi önem arzetmektedir. Dolayısıyla geliştirilen model pratik olarak anlamlı ve geçerliliği yüksek bir modeldir. Ayrıca karar vericinin modeldeki kar seviyesi alt sınırını değiştirerek değişik senaryolarda farklı kar beklentileri altında kapasite yatırım alternatiflerini değerlendirmek suretiyle duyarlılık analizi yapması da mümkündür. Bu esneklik proje kapsamında geliştirilen karar destek sisteminde ele alınmıştır.

Karşılaştırmalı analiz için BTSİÇ modeli aşağıda tekrar hatırlatılmaktadır.

Bütünleşik Temel SİÇ (BTSİÇ) Modeli:

$$\text{Enb} \sum_{k=1}^{m} \sum_{j=1}^{n} w_j x_{jk} - \sum_{k=1}^{m} c_k y_k \qquad (1')$$

$$\sum_{k=1}^{m} x_{jk} \leq 1 \qquad j = 1,...,n \qquad (2)$$

$$\sum_{j \in P_a} x_{jk} \leq 1 \qquad k = 1,...,m \qquad \forall a \qquad (3)$$

$$x_{jk} \in \{0,1\} \qquad k = 1,...,m \qquad j = 1,...,n \qquad (4)$$

$$x_{jk} \leq y_k \qquad k = 1,...,m \qquad j = 1,...,n \qquad (5)$$

$$y_k \in \{0,1\} \qquad k = 1,...,m \qquad (6)$$

KTSİÇ modeli dikkatle incelendiğinde aslında KTSİÇ probleminin BTSİÇ probleminin özel bir durumuna karşılık geldiği gözlenebilir. Bunun nedeni temelde şöyle açıklanabilir: Algoritma(BTSİÇ)'nin Tablo 3'deki her satırı yukarıdan aşağıya doğru nasıl dolduracağı Bölüm 4.2.2'de gösterilmiştir. KTSİÇ problemi Tablo 3'deki Z_{Ok} değerlerini kısıt, Z_{Tk} değerlerini ise amaç fonksiyonu olarak almaktadır. Dolayısıyla KTSİÇ probleminin optimal çözümü Tablo 3'de yukarıdan aşağıya doğru Z_{Ok} değerinin B limitini aştığı ilk satırda, karşılık gelen Z_{Tk} değeri olacaktır. Bu durumda BTSİÇ problemi çözüldüğü zaman KTSİÇ problemi de çözülmüş olacaktır. BTSİÇ probleminin çözümünde tüm satırlar için ek olarak Z_k değeri de bulunmakta, bu değerlerin en küçüğü optimal çözümü oluşturmaktadır.

Aşağıda verilen Algoritma(KTSİÇ) bu problemi polinom zamanda optimal olarak çözmektedir. Algoritma(KTSİÇ), Algoritma(BTSİÇ)'nin problem için uyarlanması yoluyla oluşturulmuştur.

Algoritma(KTSİÇ):

(S0) $Z_{Ok} = 0$,
Sisteme dahil edilmeye aday m makineyi maliyetlerine (c_k) göre artan sırada indisle, yani $c_1 \le c_2 \le ... \le c_m$ olsun.

(S1) Makine sayısının $k = 1,..., m$ arasındaki tüm değerleri için:
k makine ve n işten oluşan bir sistem için MMAA çözümü bul,
$Z_{MMAA(k)}$: MMAA amaç fonksiyonu değeri olsun,
$X_{MMAA(k)}$: MMAA çözümünde işlenebilen iş kümesi olsun,
$Z_{Ok} = Z_{MMAA(k)}$.
Eğer $Z_{MMAA(k)} \ge B$ ise
$k^* = k$,
(S2) adımına git.
Değilse devam et.

(S2) Optimal çözüm: $Z = \sum_{i=1}^{k^*} c_i y_i$,ve işlenebilen optimal iş kümesi: $X_{MMAA(k^*)}$ olarak bulunur.

Teorem 2: Algoritma(KTSİÇ) KTSİÇ modelinin optimal çözümünü $O(m^2 n \log n)$ zamanda bulur.

İspat 2: Algoritmanın (S0) adımındaki sıralama işlemi en kötü durumda $O(\log n)$ zamanda yapılabilir. Sonraki adımda Algoritma(KTSİÇ) makine sayısı alt sınırı olan 1 makineden en kötü durumda üst sınır olan m makineye kadar tüm olası makine sayıları için operasyonel temel SİÇ probleminin optimal çözümünü $O(mn \log n)$ zamanlı MMAA ile bulmakta (Bouzina ve Emmons, 1996), bu durumda (S1) adımının en kötü durum için zamansal karmaşıklığı $O(m^2 n \log n)$ olarak gerçekleşmektedir. Algoritma istenen getirinin sağlandığı ilk makine sayısında durmaktadır. Dolayısıyla Algoritma(KTSİÇ) problem için olası tüm çözümleri gezerek içlerinde en iyisini seçtiğinden optimal çözümü vermektedir. Zaman karmaşıklığı ise (S1) adımı tarafından $O(m^2 n \log n)$ olarak belirlenmektedir. □

Algoritma(KTSİÇ) ile doldurulabilecek Tablo 9 aşağıda gösterilmiştir.

Tablo 9. İlgili makine sayıları için KTSİÇ çözümleri.

Makine sayısı	***Operasyonel Problemin Amaç Fonksiyonu Değeri (Z_{Ok})***	***Kar Getirisi için Alt Sınırı Geçti mi?***	***Taktik Problemin Amaç Fonksiyonu Değeri (Z_{Tk})***
0	0	Hayır	0
. .		Hayır	
k	$Z_{MMAA(k)}$	Hayır	$\sum_{i=1}^{k} c_i y_i$
. .			
*k**	$Z_{MMAA(k^*)}$	Evet	$\sum_{i=1}^{k^*} c_i y_i$
*k**	$Z_{MMAA(k^*)}$		$Z = \sum_{i=1}^{k^*} c_i y_i$

4.3.2. Yüzde Kar Seviyesi Kısıtlı Taktik Problem

Bu bölümde B parametresinin pratik durumlarda yüzde olarak tanımlanabileceği varsayımından yola çıkılarak problemin diğer bir durumu tanımlanmaktadır.

Rezervasyon sistemlerinde ve dolayısıyla SİÇ modellerinde sistemde yapılmaya aday, hazır durumdaki tüm işler önceden belirlidir, rezervasyonları tanımlanmıştır. Bu durumda sistemde bulunan tüm aday işlerin $(j = 1, \dots, n)$ toplam getirisi kolaylıkla $\sum_{j=1}^{n} w_j$ formülü ile hesaplanabilir. Dolayısıyla B alt limiti de bu toplam potansiyel getirinin bir yüzdesi olarak tanımlanabilir, yani nominal bir değer yerine daha gerçekçi oransal bir alt limit değerinden söz edilebilir. Örneğin bir rezervasyon sisteminde gelen ve gelebilecek rezervasyonların toplam potansiyel değerinin yüzde seksenini kazanmayı hedeflemek söz konusu olabilir, böyle bir durumda B limiti bir yüzde olarak tanımlanacaktır. Modelin (7) numaralı kısıtı aşağıdaki şekilde değişmektedir:

$$\sum_{k=1}^{m}\sum_{j=1}^{n} w_j x_{jk} \geq B \sum_{j=1}^{n} w_j \qquad (7')$$

Burada belirtmek gerekir ki $B = \%100$ olarak belirlenirse problem klasik TSİÇ'ye dönüşmektedir. Parametre yüzde olarak tanımlandığında çözüm yöntemi temelde değişmeyecektir, ancak Algoritma(KTSİÇ)'nin (S0) ve (S1) adımında küçük bazı düzenlemeler ve eklemeler gerekmektedir. Aşağıda verilen algoritma bu problemi polinom zamanda optimal olarak çözmektedir.

<u>Algoritma(KTSİÇ) (*B* yüzde olarak verildiğinde)</u>:

(S0) $Z_{Ok} = 0$,

$Z_{UB} = \sum_{j=1}^{n} w_j$,

Sisteme dahil edilmeye aday *m* makineyi maliyetlerine (c_k) göre artan sırada indisle, yani $c_1 \le c_2 \le ... \le c_m$ olsun.

(S1) Makine sayısının *k* = 1,..., *m* arasındaki tüm değerleri için:

k makine ve *n* işten oluşan bir sistem için MMAA çözümü bul,

$Z_{MMAA(k)}$: MMAA amaç fonksiyonu değeri olsun,

$X_{MMAA(k)}$: MMAA çözümünde işlenebilen iş kümesi olsun,

$Z_{Ok} = Z_{MMAA(k)}$.

Eğer $\frac{Z_{MMAA(k)}}{Z_{UB}} \ge B$ ise

$k^* = k$,

(S2) adımına git.

Değilse devam et.

(S2) Optimal çözüm:

$Z = \sum_{i=1}^{k^*} c_i y_i$,ve işlenebilen optimal iş kümesi: $X_{MMAA(k^*)}$ olarak bulunur.

Teorem 3: Yukarıda verilen algoritma B parametresinin yüzde olarak tanımlandığı durumda KTSİÇ modelinin optimal çözümünü $O(m^2 n\, logn)$ zamanda bulur.

İspat 3: Algoritmanın (S0) adımındaki sıralama işlemi en kötü durumda $O(logn)$ zamanda yapılabilir. Sonraki adımda makine sayısı alt sınırı olan 1 makineden en kötü durumda üst sınır olan m makineye kadar tüm olası makine sayıları için operasyonel temel SİÇ probleminin optimal çözümü $O(mn\, logn)$ zamanlı MMAA ile bulunmakta, bu durumda (S1) adımının en kötü durum için zamansal karmaşıklığı $O(m^2 n\, logn)$ olarak gerçekleşmektedir. Algoritma istenen getiri yüzdesinin sağlandığı ilk makine sayısında durmaktadır. Dolayısıyla problem için olası tüm çözümleri gezerek içlerinde en iyisini seçtiğinden optimal çözümü vermektedir. Zaman karmaşıklığı ise (S1) adımı tarafından $O(m^2 n\, logn)$ olarak belirlenmektedir.□

4.3.3. KTSİÇ için Örnek Problem

Bu bölümde yüzde kar seviyesi kısıtlı örnek bir problem üzerinde Algoritma(KTSİÇ)'nın nasıl çalıştığı gösterilmiştir. Tablo 10'da özetlenen ve Algoritma(BTSİÇ) için seçilen 200 işlik bir problemdir. B seviyesi %60 olarak seçilmiştir. Algoritmanın (S0) adımında yer alan aday makine sayısı olarak yine $Maks_a\{|P_a|\}$ üst sınırı kabul edilmiştir. Bu problem örneği için 15 makineye eşittir ve artan c_k maliyetleri sırasıyla 40, 40, 40, 40, 40, 40, 50, 50, 70, 70, 70, 80, 80, 80 ve 80'e eşittir. Problemdeki 200 işe ait toplam potansiyel getiri 1410 (Z_{UB}) hedeflenen minimum %60 getiriye denk gelen alt limit ise 846'dır.

Tablo 10'da algoritmanın adımları ve ilgili sonuçlar özetlenmiştir. Tablodan görüldüğü gibi Algoritma(KTSİÇ)'in 5. iterasyonunun (S1) adımındaki kontrolle sona erdiği görülmektedir. Dolayısıyla bu problem için kalan 10 makinenin getirilerinin hesaplanmasından, dolayısıyla 5 adet MMAA problemi çözüm zamanından tasarruf edilmiştir. %60'lık kar seviyesi hedefi için toplam maliyeti 200 olan ilk 5 makine kullanılmış, bu optimal çözümle 200 işin 125'i işlenmiş, potansiyel toplam iş getirisinin %64.5'i, ortalama %91'lik kullanım oranıyla elde edilmiştir. CPLEX ile elde edilen optimal çözümdeyse 200 işin 117'si işlenmiş, toplam iş getirisinin %60.3'üne, ortalama %85'lik bir makine kullanım oranıyla ulaşılmıştır. Algoritmanın çözüm zamanı CPLEX çözümünden yaklaşık 8 kat daha hızlıdır.

4.3.4. Gözlemler

Bu bölümde KTSİÇ ile ilgili bazı gözlem ve yorumlar ele alınmaktadır.

BTSİÇ ve KTSİÇ problemlerinde kullanılan w_j parametresi karar verici tarafından farklı şekillerde tanımlanabilir. Örneğin bu parametre işlerin müşteri önceliklerini de gözeten görece ağırlıklarını temsil edebilir. Bu durumda w_j parametresi maddi bir getiriye karşılık gelmeyecek ve BTSİÇ modelinin amaç fonksiyonu değeri anlamsız hale gelebilecektir. Ancak KTSİÇ modeli böyle bir durumda geçerliliğini korur; yalnızca ilgili kısıttaki alt limit değerinin de (B) aynı cinsten tanımlanması yeterli olacaktır. Başka bir deyişle ifade etmek gerekirse iş ağırlıklarının kar getirilerine karşılık gelecek şekilde tanımlanmadığı durumlarda BTSİÇ yerine KTSİÇ modelinin kullanımı daha uygundur.

İş ağırlıklarının kar getirileri olarak tanımlandığı durumlarda ise BTSİÇ ve KTSİÇ problemlerinde söz konusu olan kapasite artırımında yapılacak ilk yatırımın işlerden elde edilecek getiriyle karşılanmasıdır. Bu durumda her iki model de kullanılabilir, BTSİÇ problemi net karı maksimize etmeyi amaçlarken KTSİÇ probleminde belirlenen bir minimum getiri söz konusudur.

Tablo 10. Örnek problemdeki tüm aday makine sayıları için KTSİÇ çözümleri.

Makine sayısı	***Operasyonel Problemin Amaç Fonksiyonu Değeri (Z_{Ok})***	***Kar Getirisi için Alt Sınırı (846) Geçti mi?***	***Taktik Problemin Amaç Fonksiyonu Değeri (Z_{Tk})***
0	0	*Hayır*	0
1	204	*Hayır*	40
2	399	*Hayır*	80
3	583	*Hayır*	120
4	757	*Hayır*	160
5	*909 ≥ 846*	***Evet***	*200*
6	1032	*Evet*	240
7	1139	Evet	290
8	1225	Evet	340
9	1296	Evet	410
10	1344	Evet	480
11	1373	Evet	550
12	1386	Evet	630
13	1399	Evet	710
14	1405	Evet	790
Üst sınır = 15	1410	Evet	870
k*=5	ZMMAA(k*) = 909	909 ≥ 846 (1410*0.6)	$Z_{TK} = \sum_{i=1}^{5} c_i y_i = 200$

Burada dikkat edilmesi gereken bir nokta planlama periyodunun çok uzun olması durumunda ilk yatırım zamanı ile işlerin geliş zamanları arasında, dolayısıyla da iş karlarının sisteme giriş zamanları arasında uzun süreler olabileceğidir. Böyle bir durumda amaç fonksiyonu veya kısıtlarda iş getirilerinin (w_j) mutlak değerleri yerine paranın zaman değerini de gözeten net bugünkü değerler olarak kullanılması daha uygun olabilir. Herhangi bir j işinin ağırlığının net bugünkü değeri:

$$P(w_j) = \frac{w_j}{(1+i)^t}$$

formülüyle hesaplanır. Eğer j işinden elde edilecek getiri işin geliş zamanında (r_j) gerçekleşiyorsa yukarıdaki formülde $t = r_j$ olarak alınır. Eğer işin getirisi sisteme iş tamamlandığında (d_j) giriyorsa $t = d_j$ olarak alınması gerekir. Formüldeki i parametresi birim zamandaki efektif faiz oranını temsil etmektedir.

4.3.5. KTSİÇ için Sayısal Deney

Projenin üçüncü döneminde B parametresinin yüzde olarak tanımlandığı durumdaki KTSİÇ modelinin optimal çözümü için geliştirilen algoritmanın performansı kapsamlı olarak sayısal deneylerle test edilmiştir. Deney tasarımının bu model için yapılması yüzde B seviyelerinin daha gerçekçi ve rahat belirlenebilir olmasındandır. Modelin bu formu için alınan sonuçlar modeller çok benzer olduğundan diğer form için de aynen geçerli olacaktır. Bu bölümde deney tasarımı ve deneylerden elde edilen sayısal sonuçlar açıklanmaktadır.

Deney Tasarımı:

Sayısal deneyin tasarımında öncelikle probleme ait parametre değerleri için çeşitli seviyeler belirlenmiştir. Bu seviyeler BTSİÇ problemi için belirlenen ve Bölüm 4.2.5'te açıklanan seviyelerle aynıdır. Bu bölümde ek olarak tanımlanan B seviyeleri açıklanmaktadır.

Modeldeki B parametresi için düşük, orta ve yüksek kar seviyelerine karşılık gelen üç parametre düzeyi belirlenmiştir. Buna göre

- Seviye 1: $B = \%40$, düşük seviye kar beklentisini temsil eden durumu,
- Seviye 2: $B = \%60$, orta seviye kar beklentisini temsil eden durumu,
- Seviye 3: $B = \%80$, yüksek seviye kar beklentisini temsil eden durumu temsil etmektedir.

Bölüm 4.2.5'te açıklanan parametre seviyeleriyle birlikte deneysel tasarımda toplam 5 iş sayısı, 3 ağırlık, 4 maliyet parametre seviyesi ve 3 B seviyesi, dolayısıyla 180 farklı seviye bulunmaktadır. Her bir seviyeden 10 örnek problem türetilmiş, dolayısıyla toplamda 1800 örneklik büyük ölçekli bir sayısal deney gerçekleştirilmiştir.

Sayısal deneyler 2 çekirdekli, 4 GB Ram ve 2.8 GHz işlemciye sahip Windows 7 işletim sistemli bir PC üzerinde gerçekleştirilmiştir. Algoritmanın kodlanmasında MS Visual Studio 2008 ortamında C# ve C++ programlama dilleri kullanılmış, MMAA probleminin çözümü için IBM ILOG CPLEX 12.1 solver kütüphanesinden yararlanılmıştır. Matematiksel modelin kodlanmasında ve çözümünde ise yine CPLEX 12.1 solverla bütünleşik ILOG OPL 6.3 model programlama dili kullanılmıştır.

Sonuçlar:

Türetilen problem örnekleri Algoritma(KTSİÇ) (B yüzde olarak verildiğinde) kullanılarak çözülmüştür. Ayrıca KTSİÇ probleminin matematiksel modeli IBM ILOG OPL Versiyon 6.3

ortamında kodlanarak CPLEX 12.1 solver ile çözdürülmüş, sonuçlar karşılaştırılmıştır. CPLEX çözümü için süre limiti 1200 saniye olarak verilmiş, bu süre içinde çözülemeyen örnekler raporlanmıştır.

Sayısal deneyin sonuçları makine maliyetinin önceki bölümde açıklanan dört farklı seviyesi için birer tabloda gösterilmiştir. Buna göre Tablo 11 düzgün dağılan makine maliyetleri için deney sonuçlarını göstermektedir.

Tablolardaki ilk sütun problemdeki iş sayısını, ikinci sütun ise önceki bölümde açıklanan farklı getiri seviyelerini göstermektedir. "*Ort. m. UB"* olarak adlandırılan sütun ilgili parametre seviyesindeki 10 problem örneği için hesaplanmış makine üst sınırlarının ortalamasını göstermektedir. Bu üst sınır her bir problem örneği için $Maks_a\{|P_a|\}$ olarak hesaplanmakta ve optimal çözümde olabilecek maksimum makine sayısını belirlemektedir.

"Kull. ort. m. sayısı" sütunları Algoritma(KTSİÇ) çözümü ve modelin CPLEX çözümü için 10 problem örneğinde optimal çözümde kullanılan ortalama makine sayılarını vermektedir. Bu değerler problemin amaç fonksiyonunu etkilemektedir. Örneğin Tablo 11'de 20 işlik ve $B = 1$ $(B = \%40)$ ve $w = 1$ $(w_j = p_j, \forall j)$ parametre seviyesindeki 10 problem örneğinde kullanılabilecek maksimum makine sayısı 4 iken hem algoritma hem de CPLEX optimal çözümlerinde kullanılan makine sayısı 1 olarak gerçekleşmiştir. Bu durum istenilen kar seviyesini tek bir makine ile sağlayabildiğimizi ve makine maliyetinin bu durumda enazlandığını göstermektedir. Ek bir makine amaç fonksiyonu değerini gereksiz yere yükseltecektir. Tablonun tüm satırlarında iki çözüm arasında makine kullanım sayıları eşittir. Bu eşitlik amaç fonksiyonlarının iki çözümde de optimal olmasından kaynaklanmaktadır.

"Ort. m. kullan. %" sütunları algoritma çözümü ve modelin CPLEX çözümü için 10 problem örneğinde optimal çözümde gerçekleşen ortalama makine kullanım (yararlanma, *utilization*) yüzdelerini vermektedir. Bu yüzde değerler makinenin planlama periyodunun, yani [0,200] aralığının ne kadarında işlem yaptığını göstermektedir. Örneğin Tablo 11'de 500 işlik ve $B = 1$ ve $w = 2$ parametre seviyesindeki 10 problem örneğinde algoritma çözümünde kullanılan ortalama 6 makine için ortalama kullanım %93.9, CPLEX çözümünde ise %86.6'dır.

"İşlenen işler %" sütunları iki çözümde işlenmeye aday n adet işten sayı bazında yüzde kaçının işlendiğini (10 problem için ortalama) belirtmektedir. Örneğin Tablo 11'de 500 işlik ve $B = 1$ ve $w = 2$ parametre seviyesindeki 10 problem örneğinde algoritma çözümünde işlerin ortalama %38.7'si, yani 193.5 adet iş işlenmiştir. CPLEX çözümünde ise bu değer %35.8'dir.

"İşlenen ağırlık %" sütunları ise iki çözümde işlenmeye aday n adet işten getiri bazında yüzde kaçının işlendiğini (10 problem için ortalama) belirtmektedir. Bu yüzde değerler işlenen işlerin toplam ağırlığının sistemdeki n adet işin toplam ağırlığına bölünmesiyle bulunmaktadır. Örneğin Tablo 11'de 500 işlik ve $B = 1$ ve $w = 2$ parametre seviyesindeki 10 problem örneğinde algoritma çözümünde sistemdeki toplam ağırlığın ortalama %45'i optimal kapasite seviyesiyle işlenebilmiştir. Bu değer potansiyel getirinin ne kadarının elde edilebildiğinin bir göstergesidir ve problemin kar seviyesi kısıtının nasıl sağlandığını göstermektedir. Algoritmadan elde edilen yüzde, istenen kar seviyesi yüzdesi olan %40'ın üzerindedir, dolayısıyla çözüm olurlu bir çözümdür. CPLEX çözümünde ise bu değer %41.4 olarak yine istenen kar seviyesi yüzdesinin üzerindedir.

Son olarak tablolardaki *"Süre"* sütunları ise iki çözüm için ortalama çözüm süresini saniye cinsinden ifade etmektedir.

Tablolar incelendiğinde öncelikle göze çarpan nokta geliştirilen algoritmanın günümüzde en gelişmiş solver olan CPLEX 12.1 ile karşılaştırıldığında büyük bir süre avantajına sahip olmasıdır. Algoritma küçük problemlerde CPLEX ile karşılaştırılabilir sürelerde, büyük problemlerde ise çok daha kısa sürelerde optimal çözüme ulaşmaktadır. Özellikle 500 işlik problemlerde bu durum açıkça görülmektedir. Tablo 11'in 500 işlik kısımlarında en sağda yer alan değerler CPLEX'in 1200 saniye zaman limiti içerisinde optimal çözümü üretemeden çıktığı örnek problem sayısını göstermektedir. Örneğin 500 işlik ve $B = 1$ ve $w = 2$ parametre seviyesindeki 10 problem örneğinin 9 tanesi CPLEX tarafından 1200 saniye içinde çözülememiştir. Bu durumda bu süre içinde CPLEX tarafından üretilen en iyi tam sayılı çözüm amaç fonksiyonu için üst sınır değeri olarak alınmıştır. Her ne kadar bu üst sınır değerleri algoritmanın bulduğu değerlere eşit yani optimum değerler olarak oluşsa da yine de büyük bir süre avantajı göze çarpmaktadır. Dolayısıyla geliştirilen kesin çözüm algoritması süre açısından çok etkin bir algoritmadır ve daha büyük problemlere de rahatlıkla uygulanabileceği görülmektedir.

Makine maliyet seviyelerinin CPLEX çözümlerini önemli derecede etkilediği görülmüştür. Özellikle birbirinden farklı rassal maliyetlere sahip makinelerden oluşan problemlerde (Tablo 11) CPLEX çözüm süreleri diğerlerine göre çok yüksektir. Bunun sebebi şöyle açıklanabilir. Problemde makine maliyetlerinin birbirine eşit olduğu durumda (Tablo 12, 13, 14) amaç fonksiyonu aslında makine sayısını enazlama problemine dönüşmektedir, bu da maliyet enazlamaya göre kolay bir problemdir. Dolayısıyla makineler arasındaki fark ortadan kalkmakta ve optimizasyon motoru etkin çalışmaktadır. Ancak maliyetler birbirinden farklı olduğunda yani genel modelde, özellikle büyük problemlerde CPLEX optimal çözümü erken aşamalarda bulmuş olsa bile bu çözümün optimal olduğunu doğrulama konusunda zorlanmakta ve verilen süre limitinde çözüm üretememektedir. Geliştirilen algoritma ise tüm maliyet seviyeleri için gürbüz (robust) bir performans göstermekte ve çok kısa sürelerde optimal çözüm üretmektedir.

İş ağırlık seviyelerinin algoritmanın çözüm süresini etkilemediği, çözüm süresinin iş sayısıyla arttığı gözlenmektedir. Kar yüzdesi seviyelerinin de algoritma performansını etkilemediği kolaylıkla gözlemlenmektedir, ancak aynı durum CPLEX çözümü için geçerli değil gibi gözükmektedir. Kar seviye alt limiti yükseldikçe ortalama çözüm sürelerinin özellikle en büyük problemlerde hafifçe arttığı gözlemlenebilir. Dolayısıyla algoritma bu açıdan da üstündür.

Deney sonucu gözlemlenen en önemli sonuçlardan biri de algoritmanın makine kullanım oranı ve işlerin işlenme oranlarında CPLEX'e göre bariz bir üstünlük göstermesidir. Örneğin Tablo 11'de algoritmadan elde edilen optimal çözümlerde ortalama makine kullanım oranı (tüm problemler için) %69 olarak bulunmuş, bu yüzde CPLEX'in ürettiği çözümde %65 olarak gerçekleşmiştir. Büyük problemlerde kullanım farkının daha da yüksek olduğu gözlemlenmiştir. Tüm tablolarda bu durum tekrar etmektedir. Dolayısıyla algoritma aynı amaç fonksiyonu değeri ile aynı makineleri kullanarak CPLEX'e göre daha fazla kullanım oranı olan alternatif optimal çözümler üretmektedir.

Aynı şekilde işlenen işlerin ve işlenen ağırlığın yüzdeleri karşılaştırıldığında algoritma ile işlenen ağırlığın CPLEX'e göre daha fazla olduğu görülmektedir. Tüm B seviyeleri için algoritmanın işlediği iş ve ağırlık yüzdesi istenilen kısıtı CPLEX'in üzerinde yüzdelerle sağlamaktadır. Bu durum karar verici için oldukça avantajlı ve tercih edilecek bir durumdur, çünkü aynı maliyetle daha fazla iş işlenerek hem makine kullanım oranları artırılmakta, hem de daha fazla getiri elde edilmektedir. Dolayısıyla aslında geliştirilen algoritma bir bakıma çok kriterli bir çözüm algoritması gibi davranmaktadır. MMAA çözümleri maliyet enazlamanın yanında ikinci bir amaç fonksiyonu olarak bu minimum maliyetle işlenebilecek maksimum

ağırlığı belirlemekte ve karar verici için en avantajlı alternatif optimal çözüme ulaşmaktadır. Bu bakımdan da çok etkin bir algoritma olduğu gözlemlenmiştir.

Ortalama makine kullanım oranlarının iş ağırlık parametresinin seviyelerine göre değişiklikler göstermekte, getirilerin varyansı arttıkça kullanım oranları azalmaktadır. İşlenen işlerin ve işlenen ağırlığın yüzdeleri ise getirilerin varyansı arttıkça belirgin bir şekilde artmaktadır. Bu iki durum birlikte değerlendirildiğinde getiri varyansı arttıkça işlenebilecek işler arasından w_j/p_j oranı daha yüksek olanların seçilmesi sayesinde daha az makine kulllanımı ile daha fazla getiri sağlandığı söylenebilir.

BTSİÇ modeli için yapılan deney sonuçlarıyla KTSİÇ için yapılanlar karşılaştırıldığında göze çarpan temel fark makine maliyetlerinin artmasının kapasite artırımı kararını KTSİÇ için olumsuz etkilememesi olarak özetlenebilir. BTSİÇ probleminde artan makine maliyetleri beklenen şekilde kapasite artırımı kararını olumsuz yönde etkilemektedir, maliyet arttıkça aynı makinede işlenen iş sayısı artmakta, makine kullanım oranları da yükselmektedir. Ancak KTSİÇ için belirleyici olan kar seviyesi kısıtıdır, dolayısıyla bu kısıtı sağlayacak makineler her şekilde kullanılacaktır. Tablolarda bir önceki deneyden farklı olarak bu sebeple optimalde 0 makineye hiçbir şekilde rastlanmamaktadır.

Kullanılan makine sayıları karşılaştırıldığında BTSİÇ modelinin genelde KTSİÇ modelinden daha fazla makineyi kullanıma soktuğu gözlemlenmektedir. Aynı maliyetli makineler için tablolar karşılaştırıldığında KTSİÇ'nin en yüksek kar getirisi alt limiti için bile BTSİÇ modelindeki makine sayılarına erişmediği gözlemlenmiştir. Bu beklenen bir durumdur, çünkü KTSİÇ modeli yalnızca belirlenen kar seviyesi kısıtını geçmeyi hedeflemekte, esas amaç olarak maliyet minimizasyonu yapmaktadır. BTSİÇ ise ne kadar az karlı olursa olsun pozitif kar getiren her makineyi kullanıma almaktadır. Buna bağlı olarak beklenen ve gerçekleşen diğer önemli farklar ise makine kullanım oranları ve işleme yüzdeleri arasındadır. KTSİÇ modeli BTSİÇ'ye göre daha az sayıda makineyi daha yüksek kullanım oranlarıyla kullanmaktadır. Buna karşılık BTSİÇ modelinde (makine maliyetlerine bağlı olarak) daha fazla iş ve ağırlık yüzdesi işlenmektedir. Elbette KTSİÇ'deki B seviyelerinin artırılması yoluyla BTSİÇ'deki işleme yüzdelerine ulaşılabileceği unutulmamalıdır.

İki model için çözüm süreleri karşılaştırıldığında CPLEX'in KTSİÇ'de BTSİÇ'ye göre daha uzun sürelerde çözüm ürettiği, hatta birçok örnekte verilen süre limitini aştığı gözlemlenmiştir. Ancak geliştirilen algoritma aynı konfigürasyonda bir önceki algoritmayla benzer veya daha kısa sürelerde optimal çözümleri üretmiştir.

Özetlemek gerekirse projenin bu döneminde KTSİÇ için geliştirilen algoritma çok kısa sürelerde optimal çözüm üreten, yukarıda bahsedilen ek avantajlara sahip etkin bir algoritmadır.

Tablo 11. Düzgün dağılan makine maliyetleri için deney sonuçları (c_k~*U*{40,50,60,70,80}).

n	*B*	*w*	Ort. m. UB	Kul. ort. m. sayısı Algo.	Kul. ort. m. sayısı CPLEX	Ort. m. kullan. (%) Algo.	Ort. m. kullan. (%) CPLEX	İşlenen işler (%) Algo.	İşlenen işler (%) CPLEX	İşlenen ağırlık (%) Algo.	İşlenen ağırlık (%) CPLEX	Süre (sn.) Algo.	Süre (sn.) CPLEX	
20	1	1	4	1	1	44.4	44.4	61.5	61.5	63.5	63.5	0.0	0.2	
		2	3	1	1	41.0	40.9	64.0	64.0	68.0	68.0	0.0	0.1	
		3	3	1	1	39.4	39.4	59.5	59.5	67.3	67.3	0.0	0.1	
	2	1	4	1	1	42.3	39.1	66.0	64.5	68.4	66.4	0.0	0.1	
		2	3	1	1	41.0	40.8	64.0	64.0	68.0	68.0	0.0	0.1	
		3	3	1	1	39.2	38.7	63.5	62.0	70.9	68.7	0.0	0.1	
	3	1	4	2	2	29.8	28.0	90.0	85.5	91.5	85.7	0.1	0.1	
		2	3	2	2	30.7	28.7	92.5	85.5	94.0	86.0	0.1	0.1	
		3	3	2	2	30.7	29.3	89.5	85.5	93.1	88.3	0.1	0.1	
50	1	1	6	1	1	69.1	62.5	47.2	40.8	49.7	42.3	0.1	0.1	
		2	5	1	1	63.8	61.5	41.8	39.2	45.7	43.1	0.1	0.1	
		3	6	1	1	62.3	62.8	39.0	39.0	47.6	47.6	0.1	0.1	
	2	1	6	2	2	60.3	59.1	66.4	65.0	69.5	68.0	0.1	0.1	
		2	5	2	2	58.4	55.7	67.4	64.8	71.8	68.8	0.1	0.1	
		3	6	2	2	55.6	51.4	66.6	61.0	74.4	66.8	0.1	0.1	
	3	1	6	3	3	50.1	48.4	84.2	81.2	86.5	83.6	0.2	0.2	
		2	5	3	3	49.2	47.0	84.6	80.4	88.1	83.5	0.2	0.1	
		3	6	3	3	47.6	44.5	84.2	78.6	89.6	82.6	0.2	0.1	
100	1	1	9	2	2	84.8	83.0	45.7	44.8	47.9	46.9	0.1	0.2	
		2	9	2	2	76.5	61.3	47.7	38.7	53.7	43.6	0.1	0.2	
		3	9	2	2	74.8	57.1	47.4	36.0	56.4	41.6	0.1	0.3	
	2	1	9	3	3	77.9	76.7	63.9	62.8	66.0	65.0	0.1	0.2	
		2	9	3	3	71.2	62.8	64.4	57.1	70.5	62.8	0.1	0.4	
		3	9	3	3	72.2	61.9	63.2	55.1	72.0	63.2	0.1	0.5	
	3	1	9	5	5	64.8	61.8	83.6	79.0	85.3	81.1	0.2	0.8	
		2	9	4	4	65.2	63.4	80.3	78.1	85.0	82.7	0.2	0.3	
		3	9	4	4	66.3	63.0	78.7	75.1	85.6	82.1	0.2	0.2	
200	1	1	14	3	3	95.7	95.4	39.8	39.7	41.2	41.1	0.2	0.4	
		2	14	3	3	84.7	73.8	41.5	36.3	47.7	42.0	0.2	1.1	
		3	14	3	3	84.3	69.3	39.9	33.4	49.8	41.8	0.2	2.8	
	2	1	14	5	5	89.6	85.3	62.0	59.4	64.3	61.2	0.3	2.5	
		2	14	5	5	83.3	76.9	58.8	54.4	65.3	60.7	0.2	8.2	
		3	14	4	4	83.1	81.9	52.1	51.5	62.8	62.1	0.2	5.7	
	3	1	14	7	7	80.7	79.2	80.1	78.2	82.2	80.6	0.4	2.5	
		2	14	7	7	77.6	73.7	79.4	76.0	84.5	81.0	0.4	7.0	
		3	14	6	6	79.1	76.4	73.1	70.7	82.2	80.3	0.3	22.5	
500	1	1	28	7	7	100.0	100.0	40.3	40.4	40.8	40.4	1.7	10.1	
		2	28	6	6	93.9	86.6	38.7	35.8	45.0	41.4	1.3	1199.1	9*
		3	29	5	5	92.5	90.7	32.2	31.6	41.8	40.8	1.1	127.5	1
	2	1	28	11	11	98.3	95.3	60.2	59.0	62.3	60.4	1.9	1084.9	9
		2	28	9	9	93.1	92.6	53.7	53.5	61.0	60.5	2.6	490.8	4
		3	29	9	9	92.5	89.3	50.9	49.4	63.1	61.3	1.6	843.6	7
	3	1	28	16	16	92.1	91.1	79.8	79.2	82.2	81.3	2.0	972.4	8
		2	28	14	14	90.0	89.3	75.4	74.7	81.6	80.7	2.0	738.0	6
		3	29	13	13	89.9	88.6	70.6	69.7	81.5	80.6	1.7	848.5	7

* CPLEX'te 1200 saniye içinde çözülemeyen örnek problem sayısı (10 örnek üzerinden).

Tablo 12. Düşük seviyeli makine maliyetleri için deney sonuçları. ($c_k = 40, \forall k$)

			Ort. m.	Kul. ort. m. sayısı		Ort. m. kullan. (%)		İşlenen işler (%)		İşlenen ağırlık (%)		Süre (sn.)	
n	*B*	*w*	UB	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX
20	1	1	3	1	1	46.5	46.5	60.5	60.0	63.0	63.0	0.0	0.1
		2	3	1	1	46.5	46.4	66.5	66.5	69.3	69.3	0.0	0.1
		3	4	1	1	39.9	39.9	60.0	60.0	67.3	67.3	0.0	0.1
	2	1	3	1	1	42.4	39.3	72.0	64.0	74.6	66.3	0.0	0.1
		2	3	1	1	45.2	44.0	69.5	67.0	72.3	69.7	0.0	0.1
		3	4	1	1	39.9	40.0	60.0	60.0	67.3	67.3	0.0	0.1
	3	1	3	2	2	33.4	32.5	89.0	86.5	90.3	88.1	0.1	0.1
		2	3	2	2	32.8	30.1	92.5	86.0	93.2	86.4	0.1	0.1
		3	4	2	2	30.2	29.2	87.0	84.5	91.2	88.1	0.1	0.1
50	1	1	6	1	1	75.6	75.6	41.4	41.6	43.3	43.3	0.1	0.1
		2	5	1	1	61.5	62.1	39.6	39.6	44.6	44.6	0.1	0.1
		3	5	1	1	62.9	62.2	39.2	39.2	46.5	46.5	0.1	0.1
	2	1	6	2	2	62.8	57.6	69.4	64.6	71.9	66.0	0.1	0.1
		2	5	2	2	56.8	52.9	69.0	64.2	73.5	68.0	0.1	0.1
		3	5	2	2	56.2	52.1	67.0	62.6	75.1	69.4	0.1	0.1
	3	1	6	3	3	51.8	49.2	86.8	82.8	88.9	84.5	0.2	0.1
		2	5	3	3	48.8	45.6	86.4	81.0	89.3	83.4	0.2	0.1
		3	5	3	3	48.8	45.8	85.4	79.8	89.7	83.7	0.2	0.1
100	1	1	9	2	2	83.6	82.1	45.7	44.9	47.6	46.8	0.1	0.1
		2	8	2	2	74.7	59.5	47.7	38.5	53.2	42.8	0.1	0.1
		3	9	2	2	72.9	55.4	45.8	35.7	54.3	42.6	0.1	0.1
	2	1	9	3	3	78.0	76.3	63.8	62.8	66.6	65.2	0.1	0.2
		2	8	3	3	71.3	61.4	65.3	57.4	70.6	62.0	0.1	0.2
		3	9	3	3	70.1	59.8	63.2	54.2	71.9	61.6	0.1	0.2
	3	1	9	4	4	68.0	65.3	81.7	78.5	84.6	81.0	0.2	0.2
		2	8	4	4	65.5	64.0	79.6	78.0	84.0	81.9	0.2	0.2
		3	9	4	4	65.4	63.9	77.1	75.2	83.9	81.6	0.2	0.2
200	1	1	15	3	3	93.9	88.8	42.7	40.2	44.4	41.5	0.2	0.8
		2	14	3	3	84.9	75.5	41.4	36.4	47.4	40.6	0.2	0.8
		3	13	3	3	83.9	68.6	41.0	33.7	51.0	40.4	0.2	0.7
	2	1	15	5	5	88.7	87.5	61.3	60.4	63.8	63.0	0.3	1.3
		2	14	5	5	82.6	76.5	59.7	54.7	66.0	60.3	0.3	1.0
		3	13	4	4	83.1	81.3	52.0	50.6	63.0	61.1	0.2	0.8
	3	1	15	7	7	78.3	75.8	80.4	78.2	83.1	80.4	0.4	1.4
		2	14	7	7	77.9	75.9	78.1	75.9	83.0	80.7	0.3	1.1
		3	13	6	6	79.5	78.5	72.3	71.3	81.9	80.6	0.3	1.0
500	1	1	29	7	7	100.0	100.0	39.7	40.3	40.6	40.2	1.4	38.2
		2	30	6	6	93.8	91.7	38.1	35.9	44.7	40.1	1.2	21.3
		3	28	5	5	93.3	93.0	31.7	31.4	41.3	40.1	1.1	16.6
	2	1	29	11	11	98.6	96.1	60.3	60.1	62.1	60.5	2.0	65.8
		2	30	9	9	93.1	93.2	54.2	53.9	61.6	60.2	1.9	47.8
		3	28	9	9	92.8	92.4	51.4	51.3	63.5	60.1	1.7	39.5
	3	1	29	16	16	92.3	90.3	79.5	79.1	81.9	80.1	2.5	66.3
		2	30	14	14	89.8	89.6	75.5	74.9	81.5	80.3	1.9	66.1
		3	28	13	13	90.2	90.1	70.8	70.7	81.1	80.1	2.0	51.6

Tablo 13. Orta seviyeli makine maliyetleri için deney sonuçları. ($c_k = 60, \forall k$)

			Ort. m.	Kul. ort. m. sayısı		Ort. m. kullan. (%)		İşlenen işler (%)		İşlenen ağırlık (%)		Süre (sn.)	
n	*B*	*w*	UB	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX
20	1	1	4	1	1	43.2	43.2	58.5	58.5	59.6	59.6	0.0	0.1
		2	4	1	1	40.8	41.1	59.5	59.5	63.3	63.3	0.0	0.1
		3	3	1	1	41.8	42.1	62.0	62.0	69.0	69.0	0.0	0.1
	2	1	4	2	2	38.3	34.4	72.0	62.5	73.6	62.8	0.1	0.1
		2	4	1	1	38.5	36.0	68.0	61.5	71.8	65.2	0.0	0.1
		3	3	1	1	40.6	40.2	65.0	63.0	72.2	69.5	0.0	0.1
	3	1	4	2	2	30.5	29.6	88.0	85.5	89.8	86.8	0.1	0.1
		2	4	2	2	30.8	28.8	89.5	85.0	91.5	87.8	0.1	0.1
		3	3	2	2	31.3	28.4	91.5	83.5	94.2	85.9	0.1	0.1
50	1	1	5	1	1	68.4	59.0	50.0	40.2	52.9	42.5	0.1	0.1
		2	6	1	1	63.4	60.6	41.4	39.6	46.9	44.8	0.1	0.1
		3	6	1	1	62.4	62.4	39.6	39.6	47.3	47.3	0.1	0.1
	2	1	5	2	2	63.0	61.8	69.4	68.2	72.5	71.0	0.1	0.1
		2	6	2	2	57.5	53.7	67.4	62.8	72.8	67.8	0.1	0.1
		3	6	2	2	56.0	50.0	67.2	60.2	75.2	68.0	0.1	0.1
	3	1	5	3	3	51.8	48.1	87.4	81.6	89.3	82.9	0.2	0.1
		2	6	3	3	49.5	46.4	85.0	79.6	88.6	83.0	0.2	0.1
		3	6	3	3	49.3	42.7	86.2	75.6	91.2	82.5	0.2	0.1
100	1	1	9	2	2	82.8	79.4	45.1	43.3	47.5	45.5	0.1	0.1
		2	9	2	2	74.8	64.4	46.6	40.1	52.7	45.5	0.1	0.1
		3	9	2	2	71.9	57.0	44.6	35.6	54.4	43.4	0.1	0.1
	2	1	9	3	3	76.6	73.4	62.7	60.3	65.9	63.1	0.1	0.1
		2	9	3	3	71.6	62.7	64.1	56.7	69.7	61.5	0.1	0.3
		3	9	3	3	69.0	57.6	61.6	52.5	71.6	61.9	0.1	0.2
	3	1	9	4	4	66.9	65.3	80.9	78.6	83.7	81.5	0.2	0.2
		2	9	4	4	65.3	63.6	79.4	77.3	83.8	81.6	0.2	0.2
		3	9	4	4	64.3	62.6	75.3	73.3	83.7	81.7	0.2	0.2
200	1	1	14	3	3	94.6	88.5	42.1	38.9	44.3	40.9	0.2	0.7
		2	13	3	3	85.3	73.9	41.8	36.2	47.9	41.1	0.2	0.6
		3	14	3	3	85.1	68.8	40.6	33.2	50.5	40.6	0.2	0.6
	2	1	14	5	5	88.7	85.6	61.7	60.1	64.4	62.1	0.3	1.3
		2	13	5	5	82.7	77.5	58.8	55.1	65.4	60.6	0.3	0.9
		3	14	4	4	84.1	81.7	52.6	51.3	63.5	61.4	0.2	0.9
	3	1	14	7	7	78.6	76.8	80.0	78.3	82.7	80.7	0.4	1.3
		2	13	7	7	77.2	73.7	79.6	76.2	84.5	80.6	0.4	1.1
		3	14	6	6	80.0	78.4	73.7	72.0	82.9	80.5	0.3	1.1
500	1	1	28	7	7	100.0	98.5	40.9	40.8	42.0	40.8	1.3	30.8
		2	30	6	6	93.8	92.4	38.5	36.3	45.0	40.1	1.2	23.3
		3	30	5	5	93.5	93.2	32.5	32.2	41.7	40.1	1.1	19.3
	2	1	28	11	11	98.5	94.8	60.9	60.3	63.0	60.5	2.4	58.1
		2	30	9	9	92.8	92.9	53.6	53.4	61.0	60.3	1.9	47.5
		3	30	9	9	92.9	92.6	51.7	51.3	63.1	60.3	1.7	40.2
	3	1	28	15	15	92.7	90.4	79.6	78.7	82.2	80.1	2.0	67.2
		2	30	14	14	89.6	89.8	75.2	75.0	81.2	80.2	2.0	65.0
		3	30	13	13	89.5	89.9	71.1	71.3	81.3	80.2	1.9	55.6

Tablo 14. Yüksek seviyeli makine maliyetleri için deney sonuçları. ($c_k = 80, \forall k$)

			Ort. m.	Kul. ort. m. sayısı		Ort. m. kullan. (%)		İşlenen işler (%)		İşlenen ağırlık (%)		Süre (sn.)	
n	*B*	*w*	UB	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX
20	1	1	3	1	1	46.3	46.3	63.0	63.0	64.9	64.9	0.0	0.1
		2	4	1	1	41.1	41.7	61.5	61.5	65.4	65.4	0.0	0.1
		3	3	1	1	41.9	41.9	61.5	61.5	69.2	69.2	0.0	0.0
	2	1	3	1	1	43.5	41.3	72.5	66.0	74.2	68.0	0.0	0.1
		2	4	1	1	40.9	40.1	64.5	62.0	68.4	66.1	0.0	0.1
		3	3	1	1	41.6	40.3	64.5	61.5	72.2	70.0	0.0	0.0
	3	1	3	2	2	32.4	30.8	89.0	85.5	90.6	86.3	0.1	0.1
		2	4	2	2	30.6	29.1	87.5	83.0	90.5	86.0	0.1	0.1
		3	3	2	2	31.0	28.0	89.5	80.5	93.1	83.9	0.1	0.0
50	1	1	6	2	2	64.3	51.4	55.2	41.0	57.0	42.4	0.1	0.1
		2	6	1	1	63.5	61.1	41.2	39.0	46.4	44.1	0.1	0.1
		3	6	1	1	63.6	63.2	39.2	39.2	47.3	47.3	0.1	0.1
	2	1	6	2	2	60.4	59.8	67.2	66.6	69.3	68.6	0.1	0.1
		2	6	2	2	56.6	53.1	66.2	62.2	70.8	66.1	0.1	0.1
		3	6	2	2	56.6	48.3	66.8	57.2	75.0	64.3	0.1	0.1
	3	1	6	3	3	50.4	48.3	84.6	81.8	86.8	83.1	0.2	0.1
		2	6	3	3	48.3	45.8	85.6	81.2	87.9	83.0	0.2	0.1
		3	6	3	3	48.7	44.8	84.2	77.6	89.8	82.3	0.2	0.1
100	1	1	8	2	2	83.8	82.7	46.4	45.7	48.3	47.7	0.1	0.1
		2	8	2	2	75.8	62.9	47.5	40.1	53.4	45.2	0.1	0.1
		3	8	2	2	74.5	55.8	46.2	34.9	55.1	42.0	0.1	0.2
	2	1	8	3	3	77.2	73.8	63.8	61.2	66.8	63.8	0.1	0.2
		2	8	3	3	70.7	61.2	64.5	56.6	70.6	61.6	0.1	0.2
		3	8	3	3	70.3	58.3	63.2	52.9	73.0	62.3	0.1	0.1
	3	1	8	5	5	66.8	63.2	83.4	78.7	85.9	81.0	0.2	0.2
		2	8	4	4	64.5	62.2	79.1	76.2	84.0	81.3	0.2	0.2
		3	8	4	4	65.7	62.6	77.2	73.6	85.2	81.6	0.1	0.2
200	1	1	15	3	3	95.3	93.0	40.3	38.9	42.2	40.9	0.2	0.6
		2	14	3	3	84.1	74.8	41.7	36.8	47.8	41.9	0.2	0.6
		3	14	3	3	84.5	67.2	40.1	32.1	50.4	40.4	0.2	0.5
	2	1	15	5	5	89.9	87.6	61.7	60.4	64.3	62.7	0.3	1.4
		2	14	5	5	82.6	78.5	59.1	55.4	65.3	60.6	0.3	1.1
		3	14	4	4	83.6	80.9	52.1	50.3	63.1	60.9	0.2	1.0
	3	1	15	7	7	81.6	79.6	80.2	78.5	82.8	80.7	0.4	1.5
		2	14	7	7	77.0	73.7	79.7	76.3	84.6	80.5	0.4	1.1
		3	14	7	7	77.9	75.4	73.7	71.4	83.3	80.4	0.3	1.2
500	1	1	28	7	7	100.0	99.0	40.9	40.9	41.6	40.5	1.3	32.0
		2	28	6	6	93.6	91.4	38.2	36.2	44.6	40.1	1.1	21.7
		3	27	5	5	92.8	92.9	31.7	31.3	41.3	40.1	1.1	16.0
	2	1	28	11	11	98.8	96.2	59.7	59.4	61.8	60.2	2.4	61.9
		2	28	9	9	93.1	93.1	53.8	53.5	61.2	60.2	1.9	41.5
		3	27	9	9	92.4	92.1	51.6	50.9	63.6	60.1	1.7	30.8
	3	1	28	15	15	92.8	91.5	78.7	78.5	81.3	80.1	2.8	83.3
		2	28	14	14	90.6	90.6	75.4	75.0	81.4	80.1	2.0	59.4
		3	27	13	13	90.2	90.5	70.5	70.4	81.2	80.1	1.8	44.1

4.4. MAKİNE ÇALIŞMA ZAMANLARINI BELİRLEME (ÇZB) PROBLEMİ

Projede dördüncü olarak temel SİÇ probleminde kapasite kararlarının incelenmesi için hem hangi işin hangi makinede işleneceğini, hem kapasite seviyesini, hem de hangi makinenin ne kadar süre çalışacağını (çalışma zamanı) belirleyen oldukça kapsamlı bir model üzerinde çalışılmıştır. Bu bölümde bu problem incelenmekte, aşağıda problemin genel ve özel durumları ayrıntılı bir şekilde anlatılmaktadır.

Bir rezervasyon sisteminde eğer makinelerin satın alma yerine sadece işleri işledikleri sürelerde kiralandığı varsayılırsa bahsedilen problemin kullanım alanları daha iyi anlaşılabilir. Bu durumda bir makinenin toplam çalışma zamanı o makineye ödenecek toplam kiralama maliyetini direkt olarak etkileyeceğinden model çalışma zamanlarını da belirleyecektir. Problemdeki birim zaman maliyetler kiralama maliyeti yerine makinelerin birim zaman çalışma maliyetleri (operasyon maliyeti) olarak da düşünülebilir. Bu da uygulama alanının genişliği konusunda bir fikir verebilir.

Problemin uygulama alanlarına bir örnek olarak çoklu sunucu kullanan bir veri aktarım sistemindeki veri transferleri düşünülebilir. Sunucuların kullanıldıkları süre içinde maliyetlendirilebildiği böyle sistemlerde kısa sürelerde çok sayıda ve değişen büyüklüklerde veri transferi yapıldığını varsayarsak (ör. GSM operatörlerinin baz istasyonu kullanımları) problemdeki iş sayısının oldukça büyük olacağı kestirilebilir. Problemin bu tip sistemlerde kullanımı çok kısa sürelerde defalarca çözüm üretmeyi gerektirebileceğinden kısa sürelerde kaliteli çözümler almak çok önemlidir.

Zaman kısıtları altında operasyonel problem için literatürde önde gelen çalışmalar proje yürütücüsü tarafından gerçekleştirilmiştir. Çalışma zamanı kısıtları altında operasyonel sabit iş çizelgeleme problemi daha önce Eliiyi ve Azizoğlu (2010b, 2011) tarafından çalışılmıştır. İlk çalışmada (Eliiyi ve Azizoğlu, 2010b) problemin NP-zor olduğu gösterilmiş ve problem için etkin üst ve alt sınır yöntemleri içeren bir dal-sınır algoritması geliştirilmiştir. İkinci çalışmada (Eliiyi ve Azizoğlu, 2011) ise çok iyi sonuçlar veren ağ akış temelli sezgisel yöntemler geliştirilmiştir. Bu projede ise çalışma zamanı bir kısıt olarak düşünülmemiş, daha genel ve esnek bir yapı gözetilerek makinelerin çalışma zamanları birer karar değişkeni olarak ele alınmıştır.

Bu genel ve uygulama alanı geniş problemin literatürde daha önce çalışılmadığı görülmektedir. SİÇ ve makine birim zaman maliyetlerini içeren önceki bir çalışma Huang ve Lloyd (2003) tarafından yapılmıştır. Bu çalışmada problem yalnızca taktik açıdan ele alınmış ve işlerin ağırlıkları gözetilmemiştir. Sistemdeki makine sayısının tüm işleri işlemeye yetecek kadar olduğu varsayımıyla makinelerin toplam çalışma maliyeti minimize edilmeye çalışılmıştır. Dolayısıyla Huang ve Lloyd (2003) tarafından çalışılan problem ile bu projede çalışılan makine çalışma zamanlarını belirleme problemi birbirinden tamamen farklı iki problemdir. Yazarlar işlerin değişik makinelerde değişik işlem sürelerine sahip olduğu bir sistem düşünmüş, iki makine sınıfı olduğunda problemin ağ akış problemi bazlı bir yöntemle polinom zamanlı çözülebileceğini, ikiden çok makine sınıfı içinse problemin NP-zor olduğunu ispatlamıştır.

Bir sonraki bölümde problem için geliştirilen model sunulmaktadır.

4.4.1. ÇZB Problemi için Matematiksel Model

ÇZB probleminde öncekilere ek olarak bir karar değişkeni ve parametreye ihtiyaç duyulmaktadır. Bunları sırasıyla

t_k : k makinesinin toplam çalışma zamanı,

R_k : k makinesinin birim zaman çalışma maliyeti

olarak tanımlayalım. Her makinenin çalışma zamanı o makinede işlenen işlerin toplam işlem süresi olarak, yani

$$t_k = \sum_{j=1}^{n} p_j x_{jk}, \forall k$$

şeklinde ifade edilecektir.

Bu durumda işlerin toplam ağırlığını maksimize ederken kapasite seviyesine ve her makinenin ne kadar çalışacağına karar verecek ÇZB modeli aşağıdaki gibi ifade edilir:

Enb $\sum_{k=1}^{m}\sum_{j=1}^{n} w_j x_{jk} - \sum_{k=1}^{m} R_k t_k$ (8)

$\sum_{k=1}^{m} x_{jk} \le 1 \qquad j = 1,...,n$ (2)

$\sum_{j \in P_a} x_{jk} \le 1 \qquad k = 1,...,m \qquad \forall a$ (3)

$x_{jk} \in \{0,1\} \qquad k = 1,...,m \qquad j = 1,...,n$ (4)

$t_k = \sum_{j=1}^{n} p_j x_{jk} \qquad k = 1,...,m$ (9)

$t_k \ge 0 \qquad k = 1,...,m$ (10)

Yukarıda verilen amaç fonksiyonu (8) yapılabilecek tüm işlerin kar toplamlarından kullanılan makinelerin kiralama maliyetlerini çıkarmak suretiyle bulunan net karı enbüyüklemeyi ifade etmektedir. (9) numaralı kısıt ise makine çalışma zamanlarını belirlemektedir. (2) ve (3) numaralı kısıtlar atama kısıtlarıdır, (4) ve (10) ise değişkenlerin yapılarını belirleyici kısıtlardır. Bu modelde de taktik modeldeki gibi m parametresi makine sayısı için bir üst sınır olarak alınmaktadır, yani sistemde bu m makineden kaç tanesinin kullanılacağına aslında model karar verecektir.

4.4.2. Özel Durumlar

Bu bölümde problemin bazı özel durumları üzerinde durulmakta ve bu özel durumların işlem karmaşıklıkları ispatlanmaktadır.

Durum 1: Makine sayısı biliniyor, makinelerin birim zaman çalışma maliyetleri aynı

Bu özel durum sistemde kullanılacak makine sayısının önceden bilindiğini kabul etmektedir. Yani modelde verilen m değeri kullanılacak makine sayısı için kesin rakamı ifade etmekte, bu durumda problem operasyonel probleme dönüşmektedir. İkinci varsayım olarak sistemde kiralanacak tüm makinelerin kiralama maliyeti açısından özdeş olduğu (yani $R_k = R, \forall k$) varsayılırsa bu özel durum ortaya çıkmaktadır. Bu durum özdeş makinelerin hepsinin aynı tedarikçiden kiralanmasıyla veya operasyonel çalışma maliyetlerinin aynı olmasıyla açıklanabilir. Problemin bu özel durumunun polinom zamanda çözülebileceği ve işlem karmaşıklığı aşağıdaki teorem ve ispatı ile ifade edilmiştir.

Teorem 4: Eğer sistemde kullanılacak makine sayısı önceden biliniyorsa ve tüm makinelerin birim maliyetleri birbirine eşitse ($R_k = R, \forall k$) problem $O(mn\,logn)$ zamanda çözülebilir.

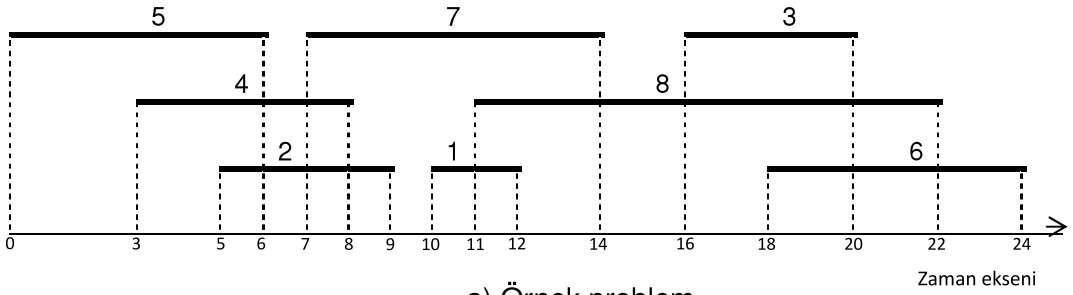
İspat 4: Bu problem $n+1$ düğüme (node) ve $2n$ çizgiye (arc) sahip bir Minimum Maliyetli Ağ Akış Problemi (MMAA) olarak modellenebilir ve $O(mn\,logn)$ sürede çözülebilir (Bouzina ve Emmons, 1996). Modelleme şu adımları izleyerek yapılır:

1. İşler hazır olma zamanlarına göre kronolojik sırada dizilir.
2. İçindeki her düğüm bir işe karşılık gelen bir düğüm kümesi $s = V_1, \dots, V_n$ yaratılır.
3. Ek olarak bir kukla düğüm $t = V_{n+1}$ yaratılır.
4. Her düğüm kendinden bir sonraki düğüme sıfır maliyet ve m birim kapasite ile bağlanır.
5. Her j işine ait V_j düğümü kendinden sonra gelen ve j işiyle zaman ekseninde çakışmayan (üstüste binmeyen) ilk işe ait düğüme bağlanır. Eğer çakışmayan bir iş bulunamıyorsa (V_j, t) çizgisi yaratılır. Yaratılan tüm bu çizgilerin maliyeti $(Rp_j - w_j)$ ve kapasitesi bir birimdir.

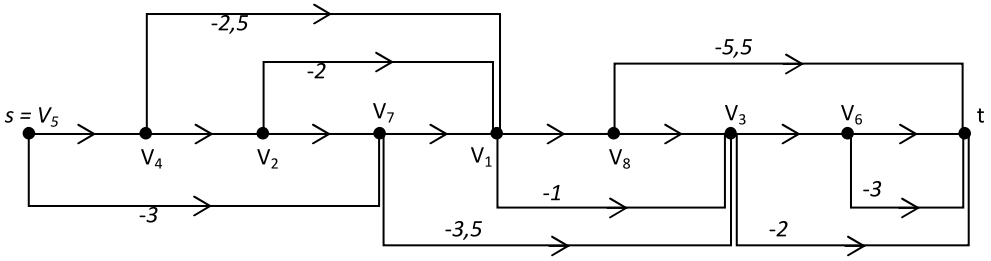
Ortaya çıkan ağda s düğümünden t düğümüne m birim akış gönderecek şekilde MMAA problemi çözülür. Çözüm $O(mn\,logn)$ zamanda bulunabilir. □

Bu özel duruma karşılık gelen bir modelleme örneği Şekil 4'de gösterilmiştir. Bu örnekte Şekil 4(a)'da zaman ekseni üzerinde gösterilen işler iki paralel özdeş makinede çizelgelenecektir. Koyu renk çizgiler işlere, üzerlerindeki sayılar iş numaralarına karşılık gelmektedir. Varsayalım bu örnekte işlerin ağırlıkları işlem sürelerine eşit olarak alınsın, yani $w_j = p_j, \forall j$ olsun. Ayrıca makine birim zaman maliyetlerini 0.5 birim olarak kabul edelim. Şekil 4(b)'de örnek probleme karşılık gelen ağ akış modeli verilmiştir.

Modelde maliyeti verilen çizgiler için kapasite bir birim, maliyeti verilmeyen çizgiler için maliyetler sıfır, kapasiteler ise $m = 2$ birimdir. Çizgilerin maliyetleri şu şekilde hesaplanır: Örneğin 5 numaralı işe ait çizginin maliyetini hesaplarken işin 6 birim zaman süreceği ve bu süre içinde 3 birim (= 6 zaman birimi x 0.5 birim maliyet) makine çalışma maliyeti getireceği dikkate alınır. Bu durumda çizginin maliyeti 3 (maliyet) - 6 (5 numaralı işin ağırlığı) = -3 olarak bulunur. Bu ağ üzerinde iki birim akış gönderecek şekilde MMAA problemi çözüldüğünde optimal çözüme ulaşılacaktır.



a) Örnek problem.



b) Karşılık gelen ağ yapısı.

Şekil 4. Makine çalışma zamanlarının belirlenmesi probleminin MMAA problemine dönüştürülmesi.

Durum 2: Tek makineli sistem

Bu özel durumda sistemde tek makine olduğu kabul edilmektedir ($m = 1$). Makinenin birim zaman çalışma (kiralama) maliyeti R olarak varsayıldığında bu özel durumun polinom zamanda çözülebileceği ve işlem karmaşıklığı aşağıdaki teorem ve ispatı ile ifade edilmiştir.

Teorem 5: Eğer sistemde tek makine varsa problem $O(n)$ zamanda çözülebilir.

İspat 5: Bu problem $n+1$ düğüme (node) ve 2*n* çizgiye (arc) sahip bir En Kısa Yol (EKY) Problemi olarak modellenebilir ve özel yapısı sebebiyle $O(n)$ sürede çözülebilir. Modelleme şu adımları izleyerek yapılır:

1. İşler hazır olma zamanlarına göre kronolojik sırada dizilir.
2. İçindeki her düğüm bir işe karşılık gelen bir düğüm kümesi $s = V_1, \dots, V_n$ yaratılır.
3. Ek olarak bir kukla düğüm $t = V_{n+1}$ yaratılır.
4. Her düğüm kendinden bir sonraki düğüme sıfır maliyet ile bağlanır.
5. Her j işine ait V_j düğümü kendinden sonra gelen ve j işiyle zaman ekseninde çakışmayan (üstüste binmeyen) ilk işe ait düğüme bağlanır. Eğer çakışmayan bir iş bulunamıyorsa (V_j, t) çizgisi yaratılır. Yaratılan tüm bu çizgilerin maliyeti $(Rp_j - w_j)$ ve kapasitesi bir birimdir.

Ortaya çıkan ağda s düğümünden t düğümüne en kısa yol problemi çözülür. Solyalı ve Özpeynirci (2009) ortaya çıkan bu problemin bir dinamik programlama yaklaşımıyla $O(n)$ zamanda çözülebileceğini göstermişlerdir. □

Durum 3: Makine sayısı biliniyor

Bu özel durumda yine sistemde kullanılacak makine sayısının önceden bilindiği varsayılmakta, ancak bu sefer genel R_k kabul edilmektedir, yani makinelerin birim zaman maliyetleri birbirinden farklıdır.

Bu durumda $t_k = \sum_{j=1}^{n} p_j x_{jk},\ k = 1,...,m$ eşitliğinin amaç fonksiyonuna alınması ile problem aslında şu hale dönüşmektedir:

Enb $\sum_{k=1}^{m}\sum_{j=1}^{n} w_j x_{jk} - \sum_{k=1}^{m}\sum_{j=1}^{n} R_k p_j x_{jk} = \sum_{k=1}^{m}\sum_{j=1}^{n} (w_j - R_k p_j) x_{jk}$ (8')

$\sum_{k=1}^{m} x_{jk} \le 1 \qquad j = 1,...,n$ (2)

$\sum_{j \in P_a} x_{jk} \le 1 \qquad k = 1,...,m \qquad \forall a$ (3)

$x_{jk} \in \{0,1\} \qquad k = 1,...,m \qquad j = 1,...,n$ (4)

Yukarıdaki modelde eğer $w_{jk} = w_j - R_k p_j$ olarak tanımlanırsa amaç fonksiyonu

Enb $\sum_{k=1}^{m}\sum_{j=1}^{n} w_{jk} x_{jk}$ (8'')

haline gelir. Yani problem iş ağırlıklarının iş ve makinelere bağlı olarak değişiklik gösterdiği daha genel bir operasyonel probleme dönüşmektedir. Bu problem Eliiyi ve Azizoğlu (2009) tarafından çalışılmış bir problemdir. Aşağıdaki Teorem bu özel durumun NP-zor olduğunu ispatlamaktadır.

Teorem 6: Sistemdeki makine sayısının bilindiği durumda makine zamanlarını belirleme problemi kuvvetli şekilde NP-zor bir problemdir (strongly NP-hard).

İspat 6: Eliiyi ve Azizoğlu (2009) iş ağırlıklarının iş ve makinelere bağlı olarak değişiklik gösterdiği genel operasyonel sabit iş çizelgeleme problemini çalışmış ve birden fazla makine sınıfı için problemin kuvvetli şekilde NP-zor olduğunu göstermişlerdir.

Makine çalışma zamanı belirleme probleminin bu özel durumunda ise iş ağırlıkları makine bağımlı olarak $w_{jk} = w_j - R_k p_j$ şeklinde tanımlanmakta, bu durum farklı tüm makine birim maliyetleri için farklı makine sınıfları yaratmaktadır. Bu durumda genel R_k için iki problem özdeştir ve bu özel durum da kuvvetli şekilde NP-zor bir problemdir. □

4.4.3. Genel ÇZB Problemi için Yaklaşık Çözüm Algoritması

Bu bölümde ÇZB probleminin makine sayısının bilinmediği, genel R_k ve birden fazla makineli genel durumu incelenmektedir. Öncelikle problemin işlem karmaşıklığı Teorem 6 yardımıyla Teorem 7'de ispatlanmaktadır.

Teorem 7: Makine zamanlarını belirleme problemi kuvvetli şekilde NP-zor bir problemdir (strongly NP-hard).

İspat 7: Sistemdeki makine sayısının bilindiği durumda makine zamanlarını belirleme probleminin kuvvetli şekilde NP-zor bir problem olduğu Teorem 5 ile ispatlanmış olduğundan ek olarak makine sayısının da bilinmediği genel problem de kuvvetli şekilde NP-zordur. Çünkü bu problem makine üst limiti kadar sayıda sıfır maliyetli kukla makine yaratarak makine sayısının bilindiği probleme dönüştürülebilir. Bu durumda kukla makinelere atanan işler işlenmemiş sayılacaktır. □

Yukarıda kuvvetli şekilde NP-zor olduğu ispatlanan ÇZB problemine bir çözüm yaklaşımı geliştirebilmek için öncelikle problemde tüm makinelerin birim zaman maliyetlerine doğru küçükten büyüğe sıralı olduklarını düşünelim. Yani $R_1 \leq R_2 \leq \cdots \leq R_m$ olsun. Bu durumda aşağıda verilen algoritma bu problem için polinom zamanda ($O(nm)$) bir yaklaşık çözüm üretmektedir. Algoritma(ÇZB) tekrarlı EKY çözümlerine dayalı, problemin yapısal özelliklerinden faydalanan etkin bir algoritmadır.

Algoritma(ÇZB):

(S0) $Z_{LB} = 0$,
$X_{LB} = \varnothing$,
Sisteme dahil edilmeye aday m makineyi maliyetlerine (c_k) göre artan sırada indisle, yani $R_1 \leq R_2 \leq \ldots \leq R_m$ olsun. Aynı şekilde işleri hazır olma zamanlarına göre kronolojik sırada diz, yani $r_1 \leq r_2 \leq \ldots \leq r_n$ olsun. $A = \{1,\ldots,n\}$ kümesi henüz atanmamış tüm işlerin kümesi olarak tanımlansın. A kümesini güncelle: $(R_1 p_j - w_j) < 0$ olan tüm işleri A kümesinden çıkar. Bu işler hiçbir optimal çözümde yer alamazlar.

(S1) Makine sayısının $k = 1,\ldots, m$ arasındaki değerleri için:
Tek makine ve A kümesinde kalan işlerle aşağıdaki şekilde bir EKY çözümü bul:

1. İçindeki her düğüm A kümesindeki bir işe karşılık gelen bir düğüm kümesi $s = V_1, V_2,\ldots, V_{|A|}$ yaratılır.
2. Ek olarak bir kukla düğüm $t = V_{|A|+1}$ yaratılır.
3. Her düğüm kendinden bir sonraki düğüme sıfır maliyet ile bağlanır.
4. Her j işine ait V_j düğümü kendinden sonra gelen ve j işiyle zaman ekseninde çakışmayan (üstüste binmeyen) ilk işe ait düğüme bağlanır. Eğer çakışmayan bir iş bulunamıyorsa (V_j, t) çizgisi yaratılır. Yaratılan tüm bu çizgilerin maliyeti $(R_k p_j - w_j)$ birimdir.
5. Ortaya çıkan ağda s düğümünden t düğümüne en kısa yol problemi çözülür.

$Z_{EKY(k)}$: Makine k için çözülen EKY probleminin amaç fonksiyonu değeri,

$X_{EKY(k)}$: Makine k için çözülen EKY probleminde işlenen işlerin kümesi.

A kümesinden atanan işleri çıkar: $A = A - X_{EKY(k)}$.

Eğer ($Z_{EKY(k)} \geq 0$) ve ($X_{EKY(k)} \neq \varnothing$) ise

$Z_{LB} = Z_{LB} + Z_{EKY(k)}$,

$X_{LB} = X_{LB} + X_{EKY(k)}$,

$k^* = k$,

A kümesini güncelle: $(R_{(k+1)}p_j - w_j) < 0$ olan tüm işleri *A* kümesinden çıkar.
A kümesi boş değilse devam et.
Değilse (S2) adımına git.

(S2) Çözüm:
$Z = Z_{LB}$ ve işlenebilen optimal iş kümesi: X_{LB} olarak bulunur.

Algoritma(ÇZB)'nin çalışma prensibi Tablo 15'de özetlenmektedir. Algoritma en ucuzdan başlayarak her makine için bir EKY çözdürmekte, bir önceki makineye atanan işleri işlenecek iş kümesinden silerek sonraki makinelerle devam etmektedir. Devam koşulu sağlanamadığında algoritma durarak k^* makineye sahip çözümü $Z = \sum_{k=1}^{k^*} Z_{EKY(k)}$ olarak vermektedir.

Algoritmanın devam koşulu daha önceki algoritmalarda olduğu gibi işleri başabaş noktasında da işlemeyi sağlayacak şekilde yapılandırılmıştır. Böylelikle aynı net getiriyle sistemdeki işlerin daha yüksek bir yüzdesinin işlenmesi hedeflenmiştir.

Tablo 15. EKY çözümleri.

Makine no	***Operasyonel Problemin Amaç Fonksiyonu Değeri***	***Devam koşulu***	***Problemin Amaç Fonksiyonu Değeri***
0	0		0
1	$Z_{EKY(1)}$	$(Z_{EKY(1)} \geq 0),(X_{EKY(1)} \neq \varnothing),(A \neq \varnothing)$	$Z = Z_{EKY(1)}$
. .			
*k**-1	$Z_{EKY(k^*-1)}$	$(Z_{EKY(k^*-1)} \geq 0),(X_{EKY(k^*-1)} \neq \varnothing),(A \neq \varnothing)$	$Z = \sum_{k=1}^{k^*-1} Z_{EKY(k)}$
*k**	$Z_{EKY(k^*)}$	$(Z_{EKY(k^*)} \geq 0),(X_{EKY(k^*)} \neq \varnothing),(A \neq \varnothing)$	$Z = \sum_{k=1}^{k^*} Z_{EKY(k)}$
*k**+1	$Z_{EKY(k^*+1)}$	$(Z_{EKY(k^*+1)} < 0)$ veya $(X_{EKY(k^*+1)} = \varnothing)$	$Z = \sum_{k=1}^{k^*} Z_{EKY(k)}$
*k**			$Z = \sum_{k=1}^{k^*} Z_{EKY(k)}$

Sonraki bölümlerde bu algoritmanın çalışmasıyla ilgili örnek bir problem verilecek ve performans testi için yapılan sayısal deney açıklanacaktır.

4.4.4. ÇZB için Örnek Problem

Bu bölümde Algoritma(ÇZB)'nin çalışma prensibi bir sonraki bölümde sonuçları özetlenen deney veri setine ait 500 işlik bir problem üzerinde gösterilmiştir. Tablo 16'da bu problem özelinde makine maliyetlerine göre sıralı olarak elde edilen EKY çözümleri, o makinede

işlenen ve toplamda atanmayan iş sayıları ve her adımdaki amaç fonksiyonu değerleri sunulmuştur. (S0) adımında yer alan aday makine sayısı olarak yine $Maks_a\{|P_a|\}$ üst sınırı kabul edilmiştir ve bu problem örneği için 26 makineye eşittir.

Artan R_k makine birim zaman maliyetleri sırasıyla 1 (7 adet), 1.25 (5 adet), 1.5 (3 adet), 1.75 (7 adet) ve 2'ye (4 adet) eşittir. Problemdeki 500 işe ait toplam potansiyel getiri 3485 ve başlangıçtaki (S0) adımında getireceği gelire oranla karsız olan iş sayısı 222'dir. Böylelikle algoritma ilk iterasyonda 500 yerine, en ucuz makineye göre karlı olan $|A| = 228$ işe ait makine çalışma zamanlarını belirlemeye başlamıştır.

Aşağıdaki tabloda algoritmanın adımları ve ilgili sonuçlar özetlenmiştir.

Tablo 16. Örnek ÇZB problemindeki tüm aday makineleri için EKY çözümleri.

Makine no	***Operasyonel Problemin Amaç Fonksiyonu Değeri ($Z_{EKY(k)}$)***	***Atanan iş sayısı / Devam koşulu sağlanıyor mu?***	***Kalan toplam iş sayısı (|A|)***	***Problemin Amaç Fonksiyonu Değeri (Z)***
0	0	0 / *Evet*	278	0
1	152	35 / *Evet*	243	152
2	105	30 / *Evet*	213	257
3	90	29 / *Evet*	184	347
4	59	23 / *Evet*	161	406
5	41	24 / *Evet*	137	447
6	30	17 / *Evet*	120	477
7	20	15 / Evet	7	497
8	3.25	3 / Evet	4	500.25
9	0.25	2 / Evet	2	500.5
10	0	1 / Evet	1	500.5
11	0.75	1 / ***Hayır***	0	501.25
*k**=11				*Z* = 501.25

Yukarıdaki tabloda belirtildiği gibi Algoritma(ÇZB)'nin 11. iterasyonunun (S1) adımındaki kontrolle sona erdiği görülmektedir. Bu sezgisel çözümle elde edilen amaç fonksiyonu değeri optimal değer olan 510'un %1.7 altındadır. Optimaldeki sonuçta 12 makineye atanmış 187 işe karşılık, bu algoritmada 11 makinede 180 iş işlenmiştir. Yukarıda da belirtildiği gibi bu işleri belirlemeye problemdeki 500 iş arasından değil, öncelikle sadece en ucuz makineye göre karlı olan 278 iş arasından başlanmıştır. Benzer şekilde 7. iterasyonun (S1) adımında 8. makinenin birim zaman maliyetinin 1.25'e çıkmasıyla ek olarak bir anda 98 iş daha elenmiştir. 500 işli problemlerin optimal çözümleri ortalamada 3.54 saniyede çözülürken, algoritmanın bu problemlerdeki ortalaması yaklaşık 0.003 saniyedir.

4.4.5. Sayısal Deney

Projenin bu döneminde ÇZB probleminin yaklaşık çözümü için geliştirilen algoritmanın performansı sayısal bir deneyle test edilmiştir. Bu bölümde deney tasarımı ve elde edilen sayısal sonuçlar açıklanmaktadır.

Deney Tasarımı:

Sayısal deneyin tasarımında öncelikle probleme ait parametre değerleri için çeşitli seviyeler belirlenmiştir. Problemin genel parametreleri için olan seviyeler temelde Bölüm 4.3.5'de KTSİÇ modeli için açıklanan seviyelerle aynıdır. Bu bölümde yalnızca farklılıklara değinilecektir.

Problemdeki iş sayıları için önceki $n = 20,50,100,200,500$ değerlerinin yanında üçüncü bölümün başında veri transferi gibi sistemleri temsil edebilecek daha büyük problemlere örnek olarak $n = 1000$ ve $n = 2000$ parametre değerleri de ele alınmıştır. Bu değerler oldukça büyük problemlere karşılık gelmektedir ve algoritma ile CPLEX çözüm süreleri arasındaki farkı vurgulayacağı düşünülmüştür.

Problemde makine birim zaman maliyetleri dışındaki tüm parametre değerlerinin tamsayı oldukları varsayılmıştır. Makine birim zaman maliyetleri ise ondalık değerler olarak belirlenmiş ve iki farklı şekilde ele alınmıştır. Hatırlanacağı üzere önceki problemler için düzgün dağılan makine maliyetleri $c_k \sim U\{40,50,60,70,80\}$ şeklinde belirlenmişti. ÇZB problemi için düzgün dağılan makine birim maliyetleri de bu değerlerden hareketle belirlenmiş ve yüksek birim maliyet seviyesi $c_k/40$, düşük birim maliye seviyesi ise $c_k/80$ olarak ele alınmıştır. Bu değerler problem örnekleri üzerinde çeşitli ön denemeler sonrasında belirlenmiştir. Makinelerin birim zaman çalışma maliyetlerinin birbiriyle aynı olduğu problem polinom zamanda çözülebilir olduğundan (bkz. Bölüm 3.1.1) bu durum deneye dahil edilmemiştir. Sonuç olarak R parametresi için aşağıdaki düzeyler oluşmuştur:

- Seviye 1: $R_k \quad U\{1,1.25,1.5,1.75,2\}$, yüksek seviye birim maliyetler

- Seviye 2: $R_k \quad U\{0.5,0.625,0.75,0.875,1\}$, düşük seviye birim maliyetler.

ÇZB modelinde B parametresi olmadığından bu parametre deneyde kullanılmamıştır.

Yukarıda açıklanan deneysel tasarımda toplam 7 iş sayısı, 3 ağırlık, 2 maliyet parametre seviyesi, dolayısıyla 42 farklı seviye bulunmaktadır. Her bir seviyeden 10 örnek problem türetilmiş, dolayısıyla toplamda 420 örneklik bir sayısal deney gerçekleştirilmiştir.

Sayısal deneyler 2 çekirdekli, 4 GB Ram ve 2.8 GHz işlemciye sahip Windows 7 işletim sistemli bir PC üzerinde gerçekleştirilmiştir. Algoritmanın kodlanmasında MS Visual Studio 2008 ortamında C++ programlama dili kullanılmış, EKY probleminin çözümü de bu kod içinde yapılmıştır. Matematiksel modelin kodlanmasında ve optimal çözümünde ise CPLEX 12.1 solverla bütünleşik ILOG OPL 6.3 model programlama dili kullanılmıştır.

Sonuçlar:

Açıklanan deney tasarımı sonrasında türetilen problem örnekleri Algoritma(ÇZB) kullanılarak çözülmüştür. Ayrıca problemin matematiksel modeli IBM ILOG OPL Versiyon 6.3 ortamında kodlanarak CPLEX 12.1 solver ile optimal olarak çözdürülmüş, sonuçlar karşılaştırılmıştır. CPLEX çözümü için süre limiti 1200 saniye olarak verilmiş, bu süre içinde çözülemeyen örnekler raporlanmıştır.

Sayısal deneyin sonuçları Tablo 17'de gösterilmiştir. Tablodaki ilk sütun problemdeki iş sayısını, ikinci sütun ise önceki bölümde açıklanan farklı maliyet seviyelerini göstermektedir.

"*Ort. m. UB"* sütunu ilgili parametre seviyesindeki 10 problem örneği için hesaplanmış makine üst sınırlarının ($Maks_a\{|P_a|\}$) ortalamasını göstermektedir.

"Kull. ort. m. sayısı" sütunu Algoritma(ÇZB) çözümü ve CPLEX çözümü için 10 problem örneğinde kullanılan ortalama makine sayılarını vermektedir. *"Ort. m. kullan. %"* sütunu algoritma çözümü ve CPLEX çözümü için 10 problem örneğinde gerçekleşen ortalama makine kullanım (yararlanma, *utilization*) yüzdelerini vermektedir. Bu yüzde değerler makinenin planlama periyodunun, yani [0,200] aralığının ne kadarında işlem yaptığını göstermektedir. *"İşlenen işler %"* sütunu iki çözümde işlenmeye aday n adet işten sayı bazında yüzde kaçının işlendiğini (10 problem için ortalama) belirtmektedir. *"İşlenen ağırlık %"* sütunu ise n adet işten getiri bazında yüzde kaçının işlendiğini (10 problem için ortalama) belirtmektedir. Bu yüzde değerler işlenen işlerin toplam ağırlığının sistemdeki *n* adet işin toplam ağırlığına bölünmesiyle bulunmaktadır. *"Süre"* sütunu iki çözüm için ortalama çözüm süresini saniye cinsinden ifade etmektedir. Son olarak *"Fark %"* sütunu ise Algoritma(ÇZB) ile bulunan çözümün amaç fonksiyonu değerinin optimale ortalamada yüzde olarak ne kadar yaklaştığını göstermektedir. Bu sütundaki değerler iki çözüm arasındaki farkın optimale bölünmesiyle yüzde olarak bulunmakta ve 10 problem örneği üzerinden ortalama olarak sunulmaktadır. Tabloda $n = 2000$ için bazı fark değerlerinin eksi olması o örneklerde algoritmanın 1200 saniye limitli CPLEX çözümlerinden daha iyi çözümler ürettiğini göstermektedir.

Tablo 17 incelendiğinde geliştirilen algoritmanın çözüm değerlerinin özellikle büyük problemlerde optimale çok yaklaştığı gözlenmektedir. Bu durum algoritmanın çök kaliteli çözümleri anında (tüm süreler sıfır saniyeye çok yakındır) ürettiğini göstermektedir. Özellikle 1000 ve 2000 işlik problemlerde bu durum açıkça görülmektedir. Tablonun 2000 işlik kısımlarında en sağda yer alan değerler CPLEX'in 1200 saniye zaman limiti içerisinde optimal çözümü üretemeden çıktığı örnek problem sayısını göstermektedir. Örneğin 2000 işlik ve $R = 2$ ve $w = 1$ parametre seviyesindeki 10 problem örneğinin hiçbiri CPLEX tarafından 1200 saniye içinde çözülememiştir. Bu süre içinde CPLEX tarafından üretilen en iyi tam sayılı çözüm amaç fonksiyonu için üst sınır değeri olarak alınmıştır. Algoritmanın bu durumların hepsinde CPLEX'ten iyi sonuçlar ürettiği gözlenmektedir, ve bu çözümleri çok büyük bir süre avantajı ile bulmaktadır. Bölüm başında belirtilen uygulamalar gözönüne alındığında daha büyük problemlerin de söz konusu olacağı değerlendirilmiştir, ve bu problemlerde paket program kullanımının etkin olmayacağı barizdir. Geliştirilen yaklaşık çözüm algoritması süre ve çözüm kalitesi açısından çok etkin bir algoritmadır ve daha büyük problemlere de rahatlıkla uygulanabileceği görülmektedir.

Ayrıca 2000 işlik ve $R = 2$ ve $w = 1$ parametre seviyesinde tablonun yanında en sağda görülen (5) rakamı CPLEX'in 1200 saniyelik süre içinde 5 problem örneğinde hiçbir olurlu çözümü üretemeden çıktığını göstermektedir. Bu durum problemin kombinatoryel yapısına dair bir ipucu vermektedir. Bu örnekler ortalamaya CPLEX hiçbir çözüm üretemediğinden katılamamıştır, ancak ilgili *"Fark %"* sütunundaki %-4.7 değeri değerlendirilirken algoritmanın aslında CPLEX'ten bu değerden çok daha üstün olduğu göz önünde tutulmalıdır.

Algoritma tarafından kullanılan makine sayısının genelde CPLEX çözümünden yüksek olduğu gözlenmektedir. Bu durum algoritmanın devam koşulunun ($Z_{EKY(k)} \geq 0$) olması ile açıklanabilir. Yeni eklenen herhangi bir makinenin net getirisi olarak tanımlanabilecek $Z_{EKY(k)}$ değerinin sıfıra eşit olduğu durumda da algoritma bu makineyi sisteme eklemekte, dolayısıyla başabaş noktasında fazla sayıda makineyle daha fazla işi işlemeyi destekleyecek şekilde davranmaktadır. Bu sebeple makine kullanım oranları da beklenen şekilde düşmektedir. İşlenen işlerin sayı ve ağırlık yüzdesi de algoritmada daha yüksek olarak gerçekleşmektedir. Algoritmanın optimal çözümü yakalayamadığı durumlarda bile kimi zaman bu değerlerin optimal çözümden daha yüksek olması dikkat çekicidir.

Tablo 17. ÇZB problemi için deney sonuçları.

			Ort. m.	Kul. ort. m. sayısı		Ort. m. kullan. (%)		İşlenen işler (%)		İşlenen ağırlık (%)		Süre (sn.)			
n	*R*	*w*	UB	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Fark (%)	
20	1	1	4	1	0	23.8	0.0	36.0	0.0	33.5	0.0	0.0	0.0	0.0	
		2	3	2	2	11.3	11.9	38.5	34.0	43.1	39.8	0.0	0.0	12.1	
		3	3	3	2	16.7	20.2	64.5	66.5	77.4	80.8	0.0	0.1	5.7	
	2	1	4	4	3	18.8	26.2	94.0	92.0	93.9	92.9	0.0	0.1	5.3	
		2	3	3	2	17.8	22.3	77.5	78.0	82.5	84.0	0.0	0.0	6.2	
		3	3	3	3	20.3	22.9	84.0	89.5	89.5	95.3	0.0	0.1	6.5	
50	1	1	6	1	0	30.8	0.0	16.2	0.0	14.7	0.0	0.0	0.0	0.0	
		2	5	3	3	18.5	15.3	36.2	30.0	40.7	35.1	0.0	0.1	3.7	
		3	6	4	4	26.7	27.7	64.8	67.2	76.7	80.7	0.0	0.1	5.5	
	2	1	6	6	4	29.4	42.7	95.6	91.4	95.4	92.9	0.0	0.1	3.9	
		2	5	4	4	30.3	31.9	77.6	80.0	82.3	85.5	0.0	0.1	6.1	
		3	6	5	5	28.6	32.1	87.6	90.8	93.1	95.9	0.0	0.0	3.7	
100	1	1	9	1	0	14.0	0.0	7.1	0.0	5.1	0.0	0.0	0.0	0.0	
		2	9	4	4	31.6	27.5	42.9	38.2	50.0	46.1	0.0	0.1	3.7	
		3	9	6	6	35.0	35.9	66.3	67.2	78.9	80.9	0.0	0.1	3.5	
	2	1	9	9	7	39.8	48.7	98.1	96.1	98.4	97.1	0.0	0.1	2.3	
		2	9	6	6	41.8	42.4	77.4	81.3	83.0	87.1	0.0	0.1	5.0	
		3	9	8	8	38.4	40.2	88.8	91.4	94.3	96.4	0.0	0.1	2.6	
200	1	1	14	1	0	16.8	0.0	0.8	0.0	0.6	0.0	0.0	0.0	0.0	
		2	14	5	5	43.0	39.4	35.4	35.6	41.9	42.5	0.0	0.2	2.9	
		3	14	9	9	44.7	45.1	61.3	63.6	75.1	77.8	0.0	0.2	4.0	
	2	1	14	14	12	49.6	59.0	98.6	97.6	98.7	98.2	0.0	0.3	2.2	
		2	14	10	11	47.9	48.5	74.6	77.5	81.4	84.2	0.0	0.2	4.5	
		3	14	13	13	47.3	48.4	86.2	89.0	93.4	95.3	0.0	0.3	2.5	
500	1	1	28	0	0	58.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
		2	28	10	12	48.9	43.2	36.2	36.7	43.1	43.9	0.0	0.3	3.3	
		3	29	18	18	57.4	56.8	62.1	63.6	76.3	77.9	0.0	1.6	2.9	
	2	1	28	28	23	60.6	74.8	97.7	95.6	97.8	96.6	0.0	10.6	2.5	
		2	28	20	21	60.1	59.1	73.4	76.1	80.5	83.1	0.0	3.2	3.7	
		3	29	25	26	58.5	58.5	86.0	88.1	93.6	94.9	0.0	5.6	1.9	
1000	1	1	50	0	0	62.3	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	
		2	50	16	19	58.3	53.9	34.9	36.4	41.6	43.4	0.0	1.2	2.8	
		3	49	29	31	69.4	67.1	62.1	63.8	76.2	78.0	0.0	45.4	2.2	
	2	1	50	50	39	68.9	86.6	98.0	95.0	98.1	96.1	0.0	577.7	2.3	
		2	50	33	37	70.2	65.4	71.5	74.1	78.8	81.3	0.0	106.6	3.1	
		3	49	42	44	68.9	66.5	85.4	86.9	93.2	94.3	0.0	355.5	1.5	
2000	1	1	91	0	0	66.7	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.0	
		2	90	31	35	59.7	55.2	34.1	35.7	40.9	42.6	0.0	38.4	3.0	
		3	93	54	61	72.6	69.9	60.8	65.1	75.1	78.8	0.1	1201.3	-0.8	10*
	2	1	91	91	43	74.2	49.2	97.6	50.8	97.7	54.0	0.1	1207.7	-4.7	10 (5)
		2	93	57	83	77.4	70.6	69.8	87.1	77.4	91.3	0.1	1203.8	-0.2	10
		3	93	78	93	73.8	71.7	84.7	96.3	92.8	98.6	0.1	1201.3	-3.5	10

* CPLEX'te 1200 saniye içinde çözülemeyen örnek problem sayısı (10 örnek üzerinden)

(5) : CPLEX tarafından 1200 saniyede hiçbir çözüm üretilemeyen örnek sayısı (10 örnek üzerinden)

İş ağırlık seviyeleri, iş sayısı veya makine maliyet seviyelerinin algoritmanın çözüm süresini etkilemediği gözlenmektedir. Dolayısıyla geliştirilen algoritma tüm maliyet seviyeleri için gürbüz (robust) bir performans göstermekte ve anında çözüm üretmektedir. Optimal çözüm içinse durum farklıdır: Makine birim zaman maliyetleri düştüğünde çözüm sürelerinin yükseldiği gözlenmektedir. Bunun sebebi daha düşük maliyetli makineler olduğunda daha fazla olurlu çözüm alternatifi oluşmasındandır.

Yüksek birim maliyetli çözümler incelendiğinde kimi problem örneklerinde hiç makine açılmadığı görülmektedir. Örneğin tablonun ilk satırında ortalaması gösterilen 10 problem örneğinin hiçbirinde hiçbir işi işlemek karlı değildir, dolayısıyla CPLEX tarafından kullanılan

makine sayısı ortalaması sıfır olmuştur. Aynı problem kümesinde 10 problemin 10'unda da algoritma optimal çözümü yakalamış, ancak kullandığı makine sayısı ortalaması yukarıda bahsedilen devam koşulu sebebiyle bir makine olarak gerçekleşmiştir.

Algoritmanın optimale yaklaşma ortalaması tablodan da görüldüğü üzere çok yüksektir, tüm örnekler için optimalden ortalama %2.7 daha düşük bir çözüm üretilmiş, toplam 420 örnek problem içinden 81 adedinde optimal sonuç bulunmuştur. $n = 1000$ için ortalama fark %2'lere gerilemekte, $n = 2000$ içinse algoritmanın ortalama performansı CPLEX'in 1200 saniyede ürettiğinin üzerine çıkmaktadır. Bu farkın daha büyük problemlerde daha belirgin hale gelmesi rahatlıkla beklenebilir.

Bu sonuçlardan hareketle ÇZB problemi için geliştirilen algoritmanın problemin yapısal özelliklerini çok iyi kullanan ve anında çözüm üretebilen, özellikle büyük problemlerde rahatlıkla yararlanılabilecek çok etkin bir algoritma olduğu söylenebilir. Algoritmanın herhangi bir paket programdan bağımsız kullanılabilecek olması da pratik uygulamalarda ayrı bir avantaj oluşturacaktır.

4.4.6. ÇZB Problemi için Kısıt Programlama Modeli

Projenin üçüncü döneminde ÇZB probleminin yaklaşık çözümü için geliştirilen algoritmanın yanında bir de Kısıt Programlama Modeli geliştirilmiştir. Bu bölümde bu model ve ilgili sonuçlar açıklanacaktır.

Kısıt programlama veya kısıtlı programlama (KP) değişkenler arasındaki ilişkilerin kısıtlarla ifade edildiği ve kısıt tatmin uygulamalarına dayalı bir programlama tekniğidir (Rossi vd., 2006). Günümüzde çizelgeleme bazlı birçok yöneylem araştırması problemi yapay zeka araştırmacıları tarafından kısıt tatmini yaklaşımları ile ele alınmaktadır. KP yazılımının sağladığı esneklik ile matematiksel programlama daha kolay ve etkin olabilmekte ve bazı kombinatoryel problemlerde tam sayılı programlamaya göre daha iyi sonuçlar elde edilmektedir. Literatürde KP yaklaşımını içeren çok sayıda çalışma bulunmaktadır. KP'nin kullanım alanları ve teknikleri hakkında bilgi edinmek için Rossi vd. (2006) ve Benhamou vd. (2007) iyi kaynaklar oluşturmaktadır.

ÇZB problemi için atama bazlı bir kısıt programlama modeli geliştirilmiştir. Bu modelde önceki parametreler aynen kullanılmıştır, ancak öncekilerden farklı olarak tek bir tamsayı karar değişkenine ihtiyaç duyulmaktadır:

x_j : j işinin atandığı makinenin indisi , $x_j \in \{1,...,2m\}$

Bu tamsayı değişken önceki ikili değişkenlerden farklı olarak her işin atandığı makineyi temsil etmektedir. Problemi atama bazlı bir modelle temsil etmek amacıyla değişkenin değer kümesi $x_j \in \{1, \ldots, 2m\}$ olarak belirlenmiştir. Makine sayısı için üst limitin m olduğu gözönüne alınırsa modelde m adet kukla makine tanımlandığı anlaşılabilir. Bu kukla makinelerin kullanımı modelin kısıtı açıklanırken anlatılacaktır.

Böyle bir değişkenle normal bir doğrusal tamsayılı matematiksel programlama modelinde amaç fonksiyonunu ifade etmek mümkün değildir. Ancak KP'de yer alan "değişken indisleme" (variable indexing) tekniği ile modelin amaç fonksiyonu aşağıdaki şekilde yazılabilmektedir:

Enb $\sum_{j=1}^{n} w_{j[x_j]}$ (11)

Amaç fonksiyonunda her işin atandığı makinedeki ağırlıklarının toplamı enbüyüklenmektedir. Toplamın içindeki $w_{j[x_j]}$ terimi aslında w_{jk} parametresindeki k indisi yerine x_j değişkenini yukarıda bahsedilen teknikten faydalanarak kullanmakta, bu sayede tamsayılı modeldeki iki toplam işaretini tek işarete indirebilmektedir. w_{jk} parametresi tanımlanırken tüm kukla makinelere ait w_{jk} değerleri sıfır olarak alınmaktadır.

Modelin tek kısıtı tüm değişkenlerin birbirinden farklı değerler almasını sağlayan ve aşağıdaki şekilde ifade edilen "*Hepsi Farklı*" (*All Different*) kısıtıdır:

$HepsiFarklı\left(x_j, \forall j \in P_a\right) \quad \forall a$ (12)

Bu global KP kısıtı her zaman aralığında ($\forall a$, daha önceki modellerden hatırlanacağı üzere), o aralıkta sistemde bulunan işlere ait her bir x_j değişkeninin farklı bir değer almasını sağlamaktadır. Bu sayede herhangi bir zaman aralığında sistemde bulunan her iş ayrı bir makineye atanacaktır. Karar değişkeninin tek indisli olarak tanımlanması sistemde kesintili işlemeyi (preemption) önlemekte, bir x_j yalnızca tek bir değer alacağından tüm zaman aralıklarında aynı değeri alması sağlanmaktadır.

Sistemde herhangi bir zamanda en fazla $Maks_a\{|P_a|\} = m$ kadar iş bulunabilir. Tanımlanan m adet kukla makine burada devreye girmektedir. Eğer bir iş w_{jk} değeri sıfır olarak tanımlanmış olan kukla makinelerden herhangi birine atanırsa o iş işlenmemiş sayılacaktır. Herhangi bir aralıkta, o aralıktaki hiçbir işin işlenmeyebileceği varsayımıyla kukla makine sayısı m olarak tanımlanmakta, bu sebeple sistemdeki makine sayısı $2m$ olmaktadır.

Yukarıda tanımlanan KP modeli deneysel olarak test edilmiş ve modelin doğruluğu ve geçerliliği ölçümlenmiştir. Model ILOG OPL Kısıt Programlama modülünde kodlanmış ve bir önceki bölümde tanımlanan problemler için aynı konfigürasyona sahip bir bilgisayar üzerinde çalıştırılmıştır. Bu tek kısıtlı atama bazlı model problemi doğru şekilde temsil etmektedir. Ancak tanımlanan model her ne kadar tek kısıta sahip etkin bir model gibi görünse de optimal çözüme çok uzun sürelerde ulaştığı gözlemlenmiştir. En küçük problem kümesi olan 20 işlik problemlerde bile kimi ölçümlerde modelin çözüm süresi 1200 saniye limitinin üzerine çıkmaktadır. 50 işlik problemlerin çözüm zamanları ise çok daha uzundur. 1200 saniye limitinde alınan sonuçlar ise Algoritma(ÇZB) ile karşılaştırılmayacak kadar kötü olduğundan raporlanmamıştır. Dolayısıyla bu sonuçlar modeli daha büyük problemler için çalıştırmayı gereksiz kılmıştır.

KP'den alınan uzun çözüm sürelerinin problemin yüksek kombinatoryel doğası ve tanımlanan ek m adet makinenin yarattığı simetriye bağlı olabileceği düşünülmektedir. Elde edilen sonuçlar ÇZB problemi özelinde geliştirilen KP modelinin Algoritma(ÇZB) ile karşılaştırıldığında çok verimsiz olduğunu göstermiştir.

4.5. BÜTÜNLEŞİK DEĞİŞKEN İŞ ÇİZELGELEME PROBLEMİ

Projenin son dönemlerinde temel değişken iş çizelgeleme (DİÇ) probleminde kapasite kararlarının incelenmesi amaçlı bir matematiksel model geliştirilmiştir. Bu bölümde bu bütünleşik problem incelenecektir. Problem genel haliyle ele alınacak, özel durumlar bazı parametre değerlerinin değişmesiyle ifade edilebilecektir.

Literatür incelendiğinde aynen SİÇ'de olduğu gibi DİÇ'de de problemin taktik ve operasyonel versiyonlarının ayrı ayrı çalışıldığı gözlemlenmektedir. DİÇ problemi SİÇ'ye göre çok daha zor bir problemdir. Literatürde SİÇ üzerine çok sayıda araştırma bulunmasına rağmen daha genel ve gerçek durumları temsil edebilen DİÇ üzerine fazla sayıda çalışma bulunmamaktadır. Bu alanda ilk çalışma Gertsbakh ve Stern (1978) tarafından taktik DİÇ problemi için yapılmış, sonrasında konu üzerinde uzun süre uğraşılmamıştır. Yakın zamanda DİÇ üzerinde çalışan az sayıda araştırmacı arasında Gabrel (1995), Rojanasoonthon vd. (2003), Rojanasoonthon ve Bard (2005) ve Garcia ve Lozano (2005) sayılabilir. Gabrel (1995), Rojanasoonthon vd. (2003), ve Rojanasoonthon ve Bard (2005) operasyonel DİÇ üzerinde çalışmış, Garcia ve Lozano (2005) ise iki aşamalı bir operasyonel DİÇ problemini araştırmıştır. Görüldüğü gibi tüm bu araştırmacılar problemi ya taktik ya da operasyonel açıdan ele almışlar, sezgisel yöntemlerle uğraşmışlardır, dolayısıyla bütünleşik bir modele rastlanmamıştır.

4.5.1. Bütünleşik DİÇ Problemi için Matematiksel Model

Bütünleşik DİÇ modeli için yeni parametre ve karar değişkenlerine ihtiyaç duyulmaktadır. Her zamanki gibi sistemde işlenmeyi bekleyen n adet iş ve kapasite seviyesi için üst sınır olarak m adet makine olduğunu varsayalım. Değişken iş çizelgelemede bir işin sisteme girme zamanı (hazır olma zamanı) ve işlenmeye başlaması için son zaman belirlidir. Herhangi bir j işinin sisteme girme zamanını r_j ile gösterirsek (bu parametre sabit iş çizelgelemedeki r_j ile aynıdır), bu j işinin herhangi bir makine tarafından işlenmeye başlaması için son zaman $b_j(\geq r_j)$ ile ifade edilir. Sabit makine maliyetleri yine c_k parametresiyle tanımlanır.

Belirtmek gerekir ki b_j parametresi sabit iş çizelgelemedeki d_j parametresinden farklıdır. Değişken iş çizelgelemede r_j ile b_j zamanları arasında geçen süre "değişkenliği" sağlayan süredir: sisteme girmiş olan j işi bu kadar süre sistemde işlenmeden bekleyebilir. Ancak b_j zamanında işleme alınmamış bir iş sistem için kayıp demektir. Sabit iş çizelgelemede ise $b_j = r_j$ olarak alınmaktadır, dolayısıyla işlerin başlama zamanı sisteme giriş zamanlarına eşit ve sabittir. Bu durumda aslında sabit iş çizelgeleme probleminin değişken iş çizelgeleme probleminin özel bir hali olduğu söylenebilir.

Bir işin özdeş makinelerden herhangi birindeki işlem süresi p_j parametresiyle ifade edilir. Projede problem makinelerin işleyebilirlik kısıtlarının olduğu daha genel haliyle çalışılmıştır. Bu amaçla j işinin k makinesi üzerindeki ağırlığı (getirisi) w_{jk} olarak belirlenmiştir. Bu genel ağırlık parametresi Eliiyi ve Azizoğlu'nun (2009) makine bağımlı ağırlıklara sahip işler için tanımında olduğu gibi aşağıdaki şekilde tanımlanmıştır.

$$w_{jk} = \begin{cases} w_j, \text{ eğer iş } j \text{ makine } k \text{ üzerinde işlenebiliyorsa} \\ \text{-1, diğer türlü} \end{cases}$$

Parametre ile işleyebilirlik kısıtları akıllı bir modellemeyle devreye girmektedir. Eğer bir makine bir işi işleyebiliyorsa o işin o makinede bir w_j getirisi olacaktır. Aksi halde negatif bir

getiri tanımlanmakadır. Bu durumda maksimizasyon modeli negatif getirili atamaları yapmayacak ve dolayısıyla işleyebilirlik kısıtları sağlanmış olacaktır.

Her zamanki gibi tüm parametrelerin negatif olmayan tamsayılardan oluştuğu ve parçalı işlemeye ve kesintiye (partial processing and preemption) izin verilmediği varsayılmaktadır.

SİÇ modelinden farklı olarak DİÇ modelinde işlerin bitiş zamanları önceden belli değildir. Bir j işi r_j ile b_j zamanları arasında herhangi bir anda işlenmeye başlayabileceği için $r_j + p_j$ ile $b_j + p_j$ zamanları arasında model tarafından belirlenecek bir zamanda bitecektir. Bu durumda SİÇ'deki zaman aralığı tanımlamaları geçersiz olmaktadır. DİÇ problemi için zaman aralıkları aşağıda verilmektedir.

Birim zaman süreli T adet zaman aralığı $\{t_1, t_2, \dots, t_T\}$ dizisi olarak tanımlanır. T parametresi planlama periyodunun uzunluğunu belirlemektedir. Bu durumda P_a kümesi $[t_a, t_{a+1})$ aralığında işlenmesi gereken işleri içermektedir $(a = 1,2, \dots, T-1)$. Matematiksel olarak:

$$P_a = \left\{ j \mid r_j \le t_a, b_j + p_j - 1 \ge t_a \right\}$$

şeklinde ifade edilir. Bir işin tüm zaman aralıklarını kapsayan kümeyi de $J_t(j)$ ile ifade edersek, bu küme şu şekilde tanımlanır:

$$J_t(j) = \left\{ r_j, \dots, b_j + p_j - 1 \right\}$$

Bütünleşik model için karar değişkenleri ise aşağıdaki gibi tanımlanır:

$$x_{jka} = \begin{cases} 1, & \text{eğer iş } j\text{'nin } a \text{ zaman aralığı } k \text{ makinesinde işlenecekse} \\ 0, & \text{diğer türlü} \end{cases}$$

$$y_{jk} = \begin{cases} 1, & \text{eğer iş } j \text{ makine } k\text{'da işlenecekse} \\ 0, & \text{diğer türlü} \end{cases}$$

$$z_k = \begin{cases} 1, & \text{eğer makine } k \text{ kullanılıyorsa} \\ 0, & \text{diğer türlü} \end{cases}$$

Modelin kısıtları aşağıda listelenmiştir:

- Eğer bir işin bir zaman aralığı atandıysa o işe ait tüm zaman aralıkları aynı makineye atanmalıdır:

$$\sum_{a \in J_t(j)} x_{jka} = p_j y_{jk}, \qquad j = 1, \dots, n \qquad k = 1, \dots, m$$

- Bir iş en fazla bir makineye atanabilir:

$$\sum_{k=1}^{m} y_{jk} \le 1, \qquad j = 1, \dots, n$$

- Her bir zaman aralığında her bir makine en fazla bir iş işleyebilir:

$$\sum_{j \in P_a} x_{jka} \leq 1 \text{ ,} \qquad a = 1,...,T-1 \qquad k = 1,...,m$$

- Eğer bir işin bir aralığı atandıysa tüm aralıkları birbiri ardına aynı makineye atanmalıdır:

$$p_j x_{jka} - p_j x_{j,k,a+1} + \sum_{t=a+2}^{b_j + p_j - 1} x_{jkt} \leq p_j \text{ , } j = 1,...,n \qquad k = 1,...,m \qquad a \in J_t(j)$$

- Eğer bir makineye herhangi bir iş atandıysa o makine kullanımdadır:

$$y_{jk} \leq z_k \text{ ,} \qquad j = 1,...,n \qquad k = 1,...,m$$

- Karar değişkenleri için ikililik tanımlamaları:

$$x_{jka} \in \{0,1\} \text{ , } y_{jk} \in \{0,1\} \text{ , } z_k \in \{0,1\} \text{ , } j = 1,...,n \qquad k = 1,...,m \qquad a = 1,...,T-1$$

Bu kısıtlarla modelin amaç fonksiyonu makine maliyetleri düşüldükten sonra oluşan net getiriyi maksimize edecektir:

$$\text{Enb} \sum_{j=1}^{n} \sum_{k=1}^{m} w_{jk} y_{jk} - \sum_{k=1}^{m} c_k z_k$$

Eliiyi, Korkmaz ve Çiçek (2009) DİÇ'nin operasyonel versiyonu üzerinde çalışmış ve problemin kuvvetli şekilde NP-zor olduğunu göstermişlerdir. Makine maliyetlerinin olmadığı (sıfıra eşit olduğu) durumda yukarıdaki model operasyonel versiyona dönüşmektedir. Dolayısıyla daha genel bir durumu temsil ettiğinden bütünleşik problem de çözülmesi zor bir problemdir.

Bu noktadan hareketle problemin yapısal özelliklerini dikkate alan özelleştirilmiş sezgisel yöntemlerle çözüm arayışına gitmenin verimli olacağı düşünülmüştür. Eliiyi, Korkmaz ve Çiçek'in (2009) çalışmasında kısıt çizgesi bazlı rassallaştırılmış bir sezgisel yöntemin operasyonel problem için çok iyi sonuçlar verdiği görülmektedir. Projenin bu aşamasında bu yöntemdeki fikirler kullanılarak yeni problem için bir sezgisel yöntem geliştirilmiştir. Yöntem bir sonraki bölümde açıklanmaktadır.

4.5.2. Bütünleşik DİÇ Problemi için Algoritma

Proje yürütücüsünün daha önceki çalışmalarından edindiği tecrübe (örnek: Eliiyi, Korkmaz ve Çiçek (2009)) özellikle değişken iş çizelgelemede probleme özel geliştirilen sezgisel yöntemlerin meta-sezgisellerden daha etkin olduğuna işaret etmektedir. Bu doğrultuda projenin son döneminde problem için rassallaştırılmış bir sezgisel yöntem geliştirilmiştir. Algoritmanın adımları aşağıda açıklanmaktadır. İlk algoritma dış döngüyü oluşturmakta, her makine sayısı için bir sonraki algoritmayı çağırmaktadır. Son iki algoritma ise AlgoritmaDİÇ'nin içinde iyileştirme amaçlı kullanılmaktadır.

AlgoritmaBDİÇ:

S0. $z_0^* = 0$. Makineleri c_k'larına göre artan sırada indisle, yani $c_1 \leq c_2 \leq \cdots \leq c_m$ olsun.

S1. Tüm kapasite seviyeleri için $(k = 1, \ldots, m)$

AlgoritmaDİÇ'yi k makine ile çalıştır. Elde edilen amaç fonksiyonu değeri z_k^* olsun.

Eğer $z_k^* < z_{k-1}^*$ ise dur. Bu durumda z_{k-1}^* elde edilen en iyi çözümdür.

Değilse bir sonraki makine sayısı ile devam et.

AlgoritmaDİÇ:

S0. $z_k^* = 0, i = 0.$

S1. $i = i + 1,$

Her bir iş için ağırlık başına çakışma ortalamasını hesapla. Bunun için;

1. İşlerin tümünü kendi r_j'lerinde başlayacakmış gibi düşünerek her iş için bu durumda oluşacak çakışmaları hesapla. Bir iş için toplam çakışma o işin diğer her bir iş ile zaman ekseni üzerinde üstüste binme miktarlarının toplamı olarak hesaplanır. Bir j işi için bu şekilde hesaplanan toplam çakışma o_{1j} olsun.

2. İşlerin tümünü kendi b_j'lerinde başlayacakmış gibi düşünerek her iş için bu durumda oluşacak çakışmaları hesapla. Bir j işi için bu şekilde hesaplanan toplam çakışma o_{2j} olsun.

3. İşlerin tümünü kendi r_j ile b_j zamanları arasında herhangi bir anda başlayacakmış gibi düşünerek her iş için bu durumda oluşacak çakışmaları hesapla. Bir j işi için bu şekilde hesaplanan toplam çakışma o_{3j} olsun.

4. Her j işi için $(o_{1j} + o_{2j} + o_{3j})/w_j$ oranını hesapla. İşleri bu orana göre artan sırada indisle.

Sistemdeki tüm işler için $(j = 1, \ldots, n)$

Rasgele bir atama yöntemi seçerek işi sıradaki uygun makineye ata:

1. İşi $[r_j, b_j]$ aralığında olabilecek en erken zamanda ata.

2. İşi $[r_j, b_j]$ aralığında olabilecek en geç zamanda ata.

3. İşi $[t, b_j]$ aralığında olabilecek en erken zamanda ata. Burada t zamanı $[r_j, b_j]$ aralığında rasgele bir sayıdır.

4. İşi $[r_j, t]$ aralığında olabilecek en geç zamanda ata. Burada t zamanı $[r_j, b_j]$ aralığında rasgele bir sayıdır.

S2. Atanabilecek iş veya makine kalmadığında bu iterasyona ait amaç fonksiyonu değerini z_i olarak hesapla.

Algoritmaİyileştirme1, *Algoritmaİyileştirme2* ve *Algoritmaİyileştirme3*'ü ardarda çalıştır. İyileşme olduğu durumda z_i ve atamaları güncelle.

Eğer $z_i > z_k^*$ ise $z_k^* = z_i$ ve yapılan atamaları en iyi atama olarak sakla.

Eğer $i = IterasyonSayısı$ ise S3'e git, değilse S1'e git.

S3. z_k^* ve en iyi atama çıktı olarak verilir.

Algoritmaİyileştirme1:

S0. $z = z_i$, $A = Atanmış\ işlerin\ kümesi$, $B = Atanmamış\ işlerin\ kümesi$.

B kümesindeki işleri w_j değerlerine göre azalan sırada indisle.

S1. B kümesindeki tüm işler için ($s = 1,2, \dots, |B|$) ve kullanımdaki her k makinesi için

B kümesinde sıradaki her s işi için dene: Makine üzerindeki diğer işlerin yerini değiştirmeden s işi k makinesine atanabiliyor mu?

Eğer mümkünse atamayı gerçekleştir, A ve B kümelerini güncelle, amaç fonksiyonunu $z = z + w_s$ olarak güncelle. B kümesindeki bir sonraki işle devam et. Tüm atanmamış işler kontrol edildiğinde dur.

Algoritmaİyileştirme2:

S0. $z = z_i$, $A = Atanmış\ işlerin\ kümesi$, $B = Atanmamış\ işlerin\ kümesi$.

B kümesindeki işleri w_j değerlerine göre azalan sırada indisle.

S1. Farklı makinelerde (k ve l) atanmış tüm iş çiftleri (j ve p) için (j işi k makinesine ve p işi l makinesine atanmış durumda iken)

1. Eğer mümkünse atanmış diğer hiçbir işin yerini değiştirmeden p işini k makinesine ve j işini l makinesine ata,

2. B kümesinde sıradaki her s işi için dene: Makine üzerindeki diğer işlerin yerini değiştirmeden s işi k veya l makinesine atanabiliyor mu?

Eğer mümkünse atamayı gerçekleştir, A ve B kümelerini güncelle, amaç fonksiyonunu $z = z + w_s$ olarak güncelle. B kümesindeki bir sonraki işle devam et. Tüm atanmamış işler kontrol edildiğinde dur.

Eğer B kümesindeki hiçbir işi eklemek mümkün değilse S1.1'de yapılan yer değişikliğini iptal et, ve bir sonraki iş çiftine geç (S1'in başına dön).

Algoritmaİyileştirme3:

S0. $z = z_i$, $A = Atanmış\ işlerin\ kümesi$, $B = Atanmamış\ işlerin\ kümesi$.

B kümesindeki işleri w_j değerlerine göre azalan sırada indisle.

S1. B kümesindeki her *j* işi için sırasıyla dene:

Kullanımdaki her k makinesi için sırasıyla dene:

1. j işinin $[r_j, r_j + p_j]$ aralığında *k* makinesinde çakıştığı işleri r_j'den $r_j + p_j$'ye doğru ara, bu aralıkta çakışan ilk iş = *sol_iş*,

2. *sağ_iş= sol_iş* 'in bitiş zamanından sonra başlayan ilk iş,

3. *sol_iş* ve ondan önce gelen tüm işleri çakışma olmayacak şekilde mümkün olduğunca geriye kaydır, benzer şekilde *sağ_iş* ve ondan sonra gelen tüm işleri mümkün olduğunca ileriye kaydır.

4. *j* işi *k* makinesine *sol_iş* ve *sağ_iş* işleri arasında oluşan boşluğa atanabiliyor mu?

5. Eğer mümkünse atamayı gerçekleştir, A ve B kümelerini güncelle, amaç fonksiyonunu $z = z + w_j$ olarak güncelle.

Geliştirilen karmaşık sezgisel yapıyı özetle açıklamak gerekirse, dış döngüyü oluşturan *AlgoritmaBDİÇ* bir makineden başlayarak her kapasite seviyesi için *AlgoritmaDİÇ* 'yi çağırmakta ve amaç fonksiyonu değerlerini karşılaştırarak en iyi çözümü sağlayan kapasite seviyesinde durmaktadır.

AlgoritmaDİÇ problemin doğasındaki değişkenliği rassallaştırılmış bir yapıyla ele almaktadır. Buna göre algoritma deneysel olarak belirlenen olan $IterasyonSayısı$ kadar döngü (iterasyon) yapmakta, bu döngülerin her birinde değişken işlerin atanma zamanları rassal olarak belirlenmektedir. İşlerin en erken ve en geç başlama zamanlarında atanacakları durumlar da ele alınarak her iş için diğer işlerle ortalama çakışması hesaplanmaktadır. Daha sonra her j işi için $(o_1 + o_2 + o_3)/w_j$ oranının hesaplanarak işlerin bu orana göre artan sırada indislenmesiyle çakışması düşük ve/veya ağırlığı yüksek işlerin önce atanması sağlanmaktadır. Daha sonra işler bu sırada eldeki makinelere yine rassallaştırılmış bir şekilde atanmaktadır. Yapılan tüm döngüler sonucunda eldeki makine sayısıyla elde edilen en iyi amaç fonksiyonu değerine sahip çözüm dış döngüye beslenmektedir.

AlgoritmaDİÇ her döngüde atamaları gerçekleştirdikten sonra *Algoritmaİyileştirme1*, A*lgoritmaİyileştirme2* ve *Algoritmaİyileştirme3*'ü ardarda çağırarak iyileşme olduğu durumda amaç fonksiyonu ve atamaları güncellemektedir. *Algoritmaİyileştirme2* kısaca bir "Yer değiştir

ve Ekle" sezgiseli olarak, *AlgoritmaİyileştirmeЗ* ise bir "Kaydır ve Ekle" sezgiseli olarak tanımlanabilir. Her iki algoritma da *AlgoritmaDİÇ*'nin o anki iterasyonunda eldeki çözümü atanamamış işleri çözüme ekleyerek iyileştirmeyi amaçlamaktadır. Bunun için ilk algoritma farklı makinelere atanmış iş çiftlerinin yerlerini değiştirerek ortaya çıkarılabilecek boşluklardan faydalanmaya çalışırken, ikinci algoritma ise aynı makinedeki işlerin yerlerini kaydırarak zaman boşluğu yaratmaya çalışmaktadır. Böylece her iterasyonda iyileştirilmiş bir aday çözüm elde edilmekte, aday çözümlerin en iyisi ise *AlgoritmaDİÇ*'nin çıktısı olmaktadır.

Geliştirilen ve yukarıda açıklanan matematiksel modelin ve sezgisel yöntemin sayısal analizi bir sonraki bölümde açıklanmaktadır.

4.5.3. Sayısal Deney

Projenin son döneminde BDİÇ modelinin yaklaşık çözümü için geliştirilen algoritmanın performansı kapsamlı olarak sayısal deneylerle test edilmiştir. Bu bölümde deney tasarımı ve deneylerden elde edilen sayısal sonuçlar açıklanmaktadır.

Deney Tasarımı:

Sayısal deneyin tasarımında öncelikle probleme ait parametre değerleri için çeşitli seviyeler belirlenmiştir. Bu seviyeler projede incelenen diğer problemler için belirlenen seviyelerle benzerdir. Tasarım bu bölümde tekrar hatırlatılmakta, ek olarak tanımlanan parametre seviyeleri açıklanmaktadır.

Problemin genelliğini bozmadan bütün parametre değerlerinin tamsayı oldukları varsayılmıştır. Problemdeki iş sayıları için $n = 20, 50, 100, 200, 500$ değerleri ele alınmıştır.

İşlerin sisteme girme yani varış zamanları $[0, 200]$ arasında düzgün dağılacak şekilde türetilmiştir. İşlerin işlem süreleri için yüksek varyans ve düşük varyansa sahip iki farklı seviye belirlenmiştir. İlk seviye için işlem sürelerinin $[4, 10]$ değerleri arasında kesikli düzgün dağıldığı varsayılmış $(p_j = 1)$, daha yüksek varyansa sahip ikinci seviye içinse $(p_j = 2)$ $p_j \sim U[4,20]$ seçilmiştir.

Deney tasarımı tüm makinelerde tüm işlerin işlenebildiği varsayımıyla ele alınmıştır. Bu durumda $w_{jk} = w_j, \forall k$ olmaktadır. İş ağırlıkları için üç parametre değer seviyesi belirlenmiştir:

- $w_j = 1$ parametre değer seviyesinde her işin getirisi kendi işlem süresine eşittir $(w_j = p_j, \forall j)$. Bu seviye bir işin getirisinin işlenme süresiyle orantılı olduğu durumları temsil etmek için tasarlanmıştır.

- $w_j = 2$ parametre değer seviyesi getirilerin (ağırlıkların) işlem süreleriyle aynı dağılımdan geldiği rassal durumu temsil etmektedir $(w_j \sim U[4,10])$. Bu durum düşük varyanslı bir rassal dağılıma karşılık gelmektedir.

- $w_j = 3$ parametre değer seviyesi ise getirilerin $w_j \sim U[4,20]$ dağılımından geldiği rassal durumu temsil etmektedir. Bu durum yüksek varyanslı bir rassal dağılıma karşılık gelmektedir.

Deney tasarımında makine maliyetleri üç farklı şekilde ele alınmıştır. Buna göre:

- Seviye 1. $c_k \sim U\{80, 100, 120, 140, 160\}$
- Seviye 2: $c_k = 80, \forall k$
- Seviye 3: $c_k = 160, \forall k$

olarak belirlenmiştir. İlk seviyede makine maliyetleri $\{80, 100, 120, 140, 160\}$kümesi içinden düzgün dağılacak şekilde rassal olarak türetilmiştir. Olabilecek en düşük makine maliyeti olabilecek en düşük iş getirisinin 20 katı olacak şekilde belirlenmiştir. Özdeş makinelerin maliyetleri arasında tedarikçi farkı vb. nedenlerden dolayı kaynaklanabilecek farkın ise maksimum %100 olacağı varsayılarak en yüsek makine maliyeti en düşük maliyetin 2 katı olarak belirlenmiştir. Ara değerlerle beraber 5 farklı maliyet oluşmaktadır. İkinci parametre seviyesi tüm makine maliyetlerinin en düşük seviyede olduğu durumu temsil ederken üçüncü seviye yüksek maliyet düzeyine karşılık gelmektedir.

Deney tasarımında ayrıca BDİÇ modelinde işlerin değişken bekleme süreleri standby zamanı parametresi $b_j - r_j$ için düşük ve yüksek varyanslara sahip iki seviyede düşünülmüştür, Buna göre

- Seviye 1: $b_j - r_j \sim U[0,10]$
- Seviye 2: $b_j - r_j \sim U[0,20]$

olarak belirlenmiştir. Düzgün dağılım aralığına sıfır değerlerinin de dahil edilmesiyle problemin SİÇ'yi de kapsayacak sekilde genelleştirilmesi amaçlanmıştır.

Yukarıda açıklanan deneysel tasarımda toplam 5 iş sayısı, 2 işlem süresi, 3 ağırlık, 3 maliyet parametre seviyesi ve 2 standby zamanı seviyesi, dolayısıyla 180 farklı seviye bulunmaktadır. Her bir seviyeden 10 örnek problem türetilmiş, dolayısıyla toplamda 1800 örneklik büyük ölçekli bir sayısal deney gerçekleştirilmiştir.

Sayısal deneyler 2 çekirdekli, 4 GB Ram ve 2.8 GHz işlemciye sahip Windows 7 işletim sistemli bir PC üzerinde gerçekleştirilmiştir. Algoritmanın kodlanmasında MS Visual Studio 2008 ortamında C# ve C++ programlama dilleri kullanılmıştır. Matematiksel modelin kodlanmasında ve çözümünde ise yine CPLEX 12.1 solverla bütünleşik ILOG OPL 6.3 model programlama dili kullanılmıştır.

Sonuçlar:

Açıklanan deney tasarımı sonrasında türetilen problem örnekleri Algoritma(BDİÇ) kullanılarak çözülmüştür. Algoritma 100 iterasyon boyunca çalıştırılmış ve her iterasyonda tüm iyileştirme algoritmaları uygulanmıştır. Ayrıca BDİÇ probleminin matematiksel modeli IBM ILOG OPL Versiyon 6.3 ortamında kodlanarak CPLEX 12.1 solver ile 1200 saniye limitiyle çözdürülmüş, sonuçlar karşılaştırılmıştır. CPLEX çözümünün 1200 saniyeyi aştığı durumlarda bu zaman limitinde üretilen en iyi olurlu çözüm CPLEX çözümü olarak alınmıştır.

Sayısal deneyin sonuçları makine maliyetinin önceki bölümde açıklanan üç farklı seviyesi için birer tabloda gösterilmiştir. Buna göre Tablo 18 düzgün dağılan makine maliyetleri için deney

sonuçlarını, Tablo 19 ve 20 ise sırasıyla düşük ve yüksek maliyetli makineler için sonuçları göstermektedir. 500 işlik problemlerde CPLEX hafıza hatası verdiğinden bu problemler için CPLEX ve Algoritma(BDİÇ) karşılaştırılması yapılmamıştır.

Tablolardaki ilk sütun problemdeki iş sayısını, ikinci sütun ise önceki bölümde açıklanan farklı standby zamanı seviyelerini göstermektedir. Sonraki iki sütun farklı getiri ve işlem süresi seviyelerini temsil etmektedir. "*Ort. m. UB"* olarak adlandırılan sütun ilgili parametre seviyesindeki 10 problem örneği için hesaplanmış makine üst sınırlarının ortalamasını göstermektedir. Bu üst sınır her bir problem örneği için $Maks_a\{|P_a|\}$ olarak hesaplanmakta ve optimal çözümde olabilecek maksimum makine sayısını belirlemektedir. Bu değer hesaplanırken bir işin kapladığı aralık $[r_j, b_j + p_j]$ olarak alınmış, böylece kötümser bir üst limit bulunmuştur.

"Kull. ort. m. sayısı" sütunları Algoritma(BDİÇ) çözümü ve modelin CPLEX çözümü için 10 problem örneğinde optimal çözümde kullanılan ortalama makine sayılarını vermektedir. Örneğin Tablo 18'de 50 işlik ve $b - r = 1$, $w = 1$ ve $p = 2$ parametre seviyesindeki 10 problem örneğinde kullanılabilecek maksimum makine sayısı 9 iken Algoritma(BDİÇ) ortalamada 3 makineyle çözüm üretmiş, CPLEX çözümlerinde ise kullanılan makine sayısı 2 olarak gerçekleşmiştir.

"Ort. m. kullan. (%)" sütunları Algoritma(BDİÇ) çözümü ve modelin CPLEX çözümü için 10 problem örneğinde gerçekleşen ortalama makine kullanım (yararlanma, *utilization*) yüzdelerini vermektedir. Bu yüzde değerler makinenin planlama periyodunun ne kadarında işlem yaptığını göstermektedir. Örneğin Tablo 18'de 50 işlik ve $b - r = 1$, $w = 1$ ve $p = 2$ parametre seviyesindeki 10 problem örneğinde Algoritma(BDİÇ) çözümünde kullanılan ortalama 3 makine için ortalama kullanım %69, CPLEX çözümünde ise bu oran kullanılan 2 makine için ortalama %47'dir.

"İşlenen işler (%)" sütunları iki çözümde işlenmeye aday n adet işten sayı bazında yüzde kaçının işlendiğini (10 problem için ortalama) belirtmektedir. Örneğin Tablo 18'de 50 işlik ve $b - r = 1$, $w = 1$ ve $p = 2$ parametre seviyesindeki 10 problem örneğinde algoritma tarafından üretilen çözümde işlerin ortalama %81'i işlenirken CPLEX çözümünde bu oran %41 seviyesindedir.

"İşlenen ağırlık (%)" sütunları ise iki çözümde işlenmeye aday n adet işten getiri bazında yüzde kaçının işlendiğini (10 problem için ortalama) belirtmektedir. Bu yüzde değerler işlenen işlerin toplam ağırlığının sistemdeki n adet işin toplam ağırlığına bölünmesiyle bulunmaktadır. Yine Tablo 18'de 50 işlik ve $b - r = 1$, $w = 1$ ve $p = 2$ parametre seviyesindeki 10 problem örneğinde algoritma çözümünde sistemdeki toplam ağırlığın %80'i işlenebilmiştir. Bu değer potansiyel getirinin ne kadarının elde edilebildiğinin bir göstergesidir. CPLEX çözümünde bu oran %42'dir.

Tablolardaki *"Süre"* sütunları iki çözüm için ortalama çözüm süresini saniye cinsinden ifade etmektedir. Tablodan görüleceği üzere 50 iş ve daha büyük problemlerin çok büyük bir kısmı 1200 saniye limitinde optimal olarak çözülememiştir. *"Fark (%)"* sütunu algoritma ve CPLEX'in ürettiği çözümlerin amaç fonksiyonu değerleri arasındaki yüzde farkı $100 * (z_{CPLEX} - z_{Algoritma(BDİÇ)})/z_{CPLEX}$ olarak vermektedir. Dolayısıyla eksi değerler algoritmanın CPLEX'ten daha iyi çözüm ürettiğini işaret etmektedir.

Son olarak *"CPLEX ZERO"* sütunu 1200 saniyelik zaman limiti içerisinde CPLEX'in 10 problem örneğinden kaçı için hiç olurlu çözüm üretemeden veya negatif amaç fonksiyonu değerleri bularak çıktığını göstermektedir. Bu problem örnekleri için *"Fark (%)"* değerleri

hesaplanamamaktadır. Bu sütundaki yüksek değerler problemin yüksek kombinatoryel yapısının bir göstergesidir.

Tablo 18. Düzgün dağılan makine maliyetleri için deney sonuçları. (c=1)

				Ort. m.	Kul. ort. m. sayısı		Ort. m. kullan. (%)		İşlenen işler (%)		İşlenen ağırlık (%)		Süre (sn.)			CPLEX
n	*b-r*	*w*	*p*	UB	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Fark (%)	ZERO*
20	1	1	1	4	1	1	44	45	75	75	76	76	0.1	4.4	2.8	
			2	5	1	1	60	60	60	61	64	65	0.1	159.0	4.2	
		2	1	4	1	1	43	43	74	74	77	77	0.1	1.7	0.0	
			2	5	1	1	25	27	31	31	32	32	0.1	10.1	2.6	
		3	1	4	1	1	43	43	75	75	80	80	0.1	3.3	0.2	
			2	5	1	1	51	52	62	62	68	69	0.1	31.5	3.5	
	2	1	1	5	1	1	52	53	89	89	89	89	0.1	6.9	1.0	
			2	5	1	1	65	65	67	68	69	70	0.1	273.1	3.0	
		2	1	4	1	1	52	52	91	91	92	92	0.0	2.2	0.8	
			2	5	1	1	51	52	56	55	58	59	0.1	144.2	11.0	
		3	1	5	1	1	49	50	86	86	89	90	0.1	3.5	1.3	
			2	5	1	1	63	63	69	69	76	76	0.1	50.3	2.2	
50	1	1	1	8	2	2	70	66	69	75	69	75	0.4	1200.1	-0.1	
			2	9	3	2	69	47	81	41	80	42	0.8	1200.1	-937.2	2
		2	1	8	2	2	62	66	79	69	82	73	0.6	1200.1	-1.4	
			2	9	2	1	71	31	52	19	55	20	0.3	1164.2	-764.0	5
		3	1	7	2	2	59	61	85	83	89	88	0.6	1200.1	0.5	
			2	10	2	2	65	55	69	42	76	46	0.6	1200.1	-404.9	
	2	1	1	10	2	2	64	67	89	67	89	68	0.7	1200.1	-32.9	
			2	10	3	1	71	33	87	18	86	18	1.1	1200.1	-1356.6	3
		2	1	9	2	1	71	66	79	65	82	69	0.6	1200.1	-26.2	
			2	10	2	1	76	31	65	22	69	24	0.5	1200.1	-320.9	5
		3	1	9	2	2	65	62	91	78	93	83	0.8	1200.1	-13.8	
			2	11	2	1	72	34	73	26	77	29	0.8	1200.1	-998.5	2
100	1	1	1	12	4	1	68	28	86	16	85	16	3.2	1200.1	-2015.9	4
			2	15	6	0	71	0	86	0	85	0	5.2	1200.3		10
		2	1	12	3	1	69	29	82	19	86	21	2.9	1200.1	-968.4	4
			2	15	3	1	74	3	52	1	56	1	1.9	1200.3		10
		3	1	13	4	2	66	33	89	29	93	31	4.0	1200.1	-277.7	2
			2	14	5	0	71	10	74	2	80	2	4.0	1200.3	-343.8	9
	2	1	1	15	4	1	71	39	93	21	93	22	4.7	1200.2	-628.9	2
			2	19	6	0	75	4	89	1	88	1	7.1	1200.1	-25550.0	9
		2	1	16	3	1	76	50	85	26	87	28	3.4	1200.1	-560.5	
			2	19	4	1	79	5	66	4	70	4	3.7	1201.2		10
		3	1	14	4	2	71	40	93	36	96	39	5.0	1200.1	-538.9	
			2	16	5	1	76	5	82	3	87	3	6.4	1200.1	-487.0	9
200	1	1	1	21	7	0	71	0	90	0	89	0	24.1	1200.2		10
			2	21	11	0	75	0	89	0	87	0	39.9	1200.2		10
		2	1	22	6	0	74	2	82	1	86	1	20.2	1201.8		10
			2	22	6	0	78	0	61	0	65	0	21.2	1202.2		10
		3	1	19	7	3	72	10	90	7	94	7	27.1	1200.3	-3851.2	9
			2	18	9	0	75	0	78	0	84	0	36.0	1200.2		10
	2	1	1	26	7	0	76	16	93	3	92	3	33.1	1202.5	-2977.1	7
			2	28	12	0	75	0	93	0	92	0	60.8	1206.9		10
		2	1	26	6	1	77	4	88	1	91	1	31.5	1205.6	-8837.5	9
			2	28	6	1	82	4	62	1	67	2	29.6	1204.2		10
		3	1	26	7	1	75	11	93	3	96	4	37.3	1200.3	-1599.1	7
			2	26	9	0	79	0	81	0	87	0	51.5	1200.5		10

Tablo 19. Düşük seviyeli makine maliyetleri için deney sonuçları. (c=2)

				Ort. m.	Kul. ort. m. sayısı		Ort. m. kullan. (%)		İşlenen işler (%)		İşlenen ağırlık (%)		Süre (sn.)			CPLEX
n	*b-r*	*w*	*p*	UB	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Fark (%)	ZERO*
20	1	1	1	4	1	1	49	49	79	79	79	79	0.1	3.1	1.1	
			2	5	1	1	57	57	68	67	68	69	0.1	219.6	1.6	
		2	1	4	1	1	44	44	80	81	83	83	0.1	2.1	1.3	
			2	5	1	1	56	58	65	65	69	70	0.1	11.5	8.0	
		3	1	4	1	1	45	45	80	80	85	85	0.1	3.6	0.1	
			2	4	1	1	53	54	63	63	68	68	0.1	71.3	0.1	
	2	1	1	5	1	1	51	51	90	91	90	91	0.1	3.0	3.7	
			2	5	1	1	69	69	74	75	74	75	0.1	377.0	0.7	
		2	1	5	1	1	50	51	89	90	91	92	0.0	4.3	1.8	
			2	5	1	1	62	63	71	71	75	75	0.1	91.6	1.2	
		3	1	5	1	1	51	52	90	90	92	92	0.0	4.5	0.7	
			2	6	1	1	61	62	70	71	76	76	0.1	335.0	0.7	
50	1	1	1	8	2	2	62	61	85	84	85	84	0.6	1200.1	-3.2	
			2	9	3	3	65	58	83	66	83	67	1.0	1200.1	-58.3	
		2	1	7	2	2	60	61	85	81	87	84	0.6	1200.1	-4.1	
			2	9	2	1	68	60	66	46	69	50	0.5	1200.1	-24.8	1
		3	1	8	2	2	59	58	82	83	88	89	0.7	1200.1	-0.6	
			2	10	3	3	62	54	78	62	83	68	1.0	1200.1	-39.2	
	2	1	1	9	2	2	65	66	91	73	90	74	0.7	1200.1	-29.4	
			2	10	3	1	68	47	91	26	91	27	1.2	1200.1	-181.6	3
		2	1	8	2	2	65	64	91	72	93	76	0.7	1200.1	-20.6	
			2	11	2	1	77	36	62	22	65	24	0.5	1200.1	-166.1	4
		3	1	8	2	2	67	63	91	86	94	90	0.7	1200.1	-8.2	
			2	10	3	1	71	42	85	33	89	37	1.1	1200.2	-223.8	2
100	1	1	1	12	4	3	66	52	90	61	90	61	3.9	1200.2	-73.8	1
			2	15	7	1	68	17	92	7	91	8	6.9	1200.3	-1662.8	6
		2	1	12	4	3	69	59	86	59	89	62	3.3	1200.1	-81.7	
			2	15	4	0	74	17	66	5	70	6	3.0	1200.2	-1745.1	7
		3	1	12	4	4	64	49	91	64	94	68	4.4	1200.1	-291.7	
			2	15	6	3	69	39	84	22	89	26	6.3	1200.3	-587.3	1
	2	1	1	16	4	1	71	37	95	26	94	25	4.9	1199.1	-365.6	3
			2	19	7	1	70	13	94	7	93	8	8.3	1200.3	-1853.9	7
		2	1	15	3	2	75	49	88	43	90	46	3.8	1200.2	-162.2	
			2	18	3	0	79	9	63	4	67	4	3.3	1200.4	-1646.4	8
		3	1	15	4	3	68	42	95	47	97	52	5.3	1200.2	-288.5	1
			2	18	6	2	74	28	87	16	91	19	7.2	1200.2	-1206.2	2
200	1	1	1	21	8	1	67	35	94	11	94	12	29.6	1200.2	-4359.8	2
			2	27	13	0	71	0	95	0	95	0	55.0	1200.3		10
		2	1	19	7	3	73	49	90	32	92	35	26.2	1200.3	-1176.0	
			2	27	7	1	76	2	67	1	71	1	25.6	1200.3		10
		3	1	21	8	2	68	37	94	18	96	21	31.6	1200.2	-896.5	1
			2	28	11	0	72	0	86	0	90	0	50.5	1200.3		10
	2	1	1	27	7	0	75	0	93	0	93	0	34.4	1201.8		10
			2	31	13	0	73	0	96	0	95	0	68.5	1200.5		10
		2	1	28	7	0	76	6	93	1	95	1	38.6	1201.2	-20333.3	8
			2	31	7	0	81	2	70	1	74	1	38.5	1200.4		10
		3	1	27	8	0	73	0	96	0	98	0	42.2	1200.3		10
			2	32	10	0	78	0	86	0	91	0	62.8	1200.5		10

Tablo 20. Yüksek seviyeli makine maliyetleri için deney sonuçları. (c=3)

				Ort. m.	Kul. ort. m. sayısı		Ort. m. kullan. (%)		İşlenen işler (%)		İşlenen ağırlık (%)		Süre (sn.)		Fark	CPLEX
n	*b-r*	*w*	*p*	UB	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	Algo.	CPLEX	(%)	ZERO*
20	1	1	1	4	0	0	0	0	0	0	0	0	0.1	0.1	0.0	
			2	5	0	0	0	7	0	7	0	6	0.2	5.0	10.0	
		2	1	4	0	0	0	0	0	0	0	0	0.1	0.1	0.0	
			2	4	0	0	0	0	0	0	0	0	0.2	0.1	0.0	
		3	1	4	1	1	44	44	78	78	81	81	0.1	2.8	5.8	
			2	5	0	0	15	16	19	19	20	20	0.2	8.0	4.0	
	2	1	1	5	0	0	0	0	0	0	0	0	0.1	0.1	0.0	
			2	6	1	1	58	58	55	55	55	56	0.1	161.2	9.5	
		2	1	4	0	0	0	0	0	0	0	0	0.1	0.1	0.0	
			2	6	0	0	0	0	0	0	0	0	0.2	0.2	0.0	
		3	1	4	1	1	54	54	92	92	95	95	0.0	3.3	0.4	
			2	6	1	1	50	50	58	57	61	62	0.1	71.7	6.0	
50	1	1	1	7	1	1	77	78	53	52	53	54	0.2	1200.1	5.6	
			2	9	1	1	79	81	46	33	43	32	0.3	1200.1	-20.9	
		2	1	7	1	1	70	72	55	55	59	60	0.2	1200.0	0.5	
			2	9	0	0	22	8	13	5	14	5	2.3	1116.3	34.8	9
		3	1	7	2	2	61	64	78	69	84	76	0.6	1200.1	0.1	
			2	9	1	1	72	69	45	42	52	48	0.3	1200.1	-47.6	
	2	1	1	9	1	1	82	41	57	28	56	28	0.3	1200.1	-13.5	5
			2	10	2	0	79	0	65	0	64	0	0.5	1200.1		10
		2	1	10	1	1	79	75	59	56	64	62	0.3	1200.1	-36.8	
			2	11	1	0	39	0	23	0	25	0	1.6	1200.1		10
		3	1	8	2	1	72	68	75	67	82	75	0.5	1200.1	-10.6	
			2	11	2	1	79	46	58	25	65	29	0.4	1200.1	-116.1	3
100	1	1	1	13	2	1	80	49	56	17	55	17	1.2	1200.1	-96.7	4
			2	15	4	0	80	8	59	5	56	4	2.1	1200.2	-50.7	9
		2	1	12	2	1	75	53	59	23	65	26	1.2	1200.1	-377.2	2
			2	15	1	0	79	0	26	0	28	0	0.4	1200.3		10
		3	1	12	3	2	71	57	78	55	84	62	2.3	1200.1	-36.5	1
			2	15	3	1	75	10	56	7	63	8	2.0	1200.3	-396.8	8
	2	1	1	15	3	0	81	0	73	0	71	0	2.2	1200.2		10
			2	18	5	0	81	0	75	0	72	0	3.8	1200.2		10
		2	1	16	2	1	81	40	62	17	67	19	1.6	1200.2	-693.7	4
			2	19	1	0	84	0	28	0	32	0	0.7	1200.3		10
		3	1	14	3	2	77	49	84	36	89	42	3.1	1200.2	-100.4	2
			2	18	3	0	80	4	59	2	66	2	2.5	1200.3	-9850.0	9
200	1	1	1	21	5	0	80	7	69	1	67	1	10.9	1200.2	-1316.7	9
			2	27	8	0	81	0	69	0	65	0	21.7	1200.3		10
		2	1	20	4	1	79	25	59	11	64	13	8.4	1200.3	-374.1	6
			2	26	2	1	81	4	24	2	26	2	2.2	1201.3		10
		3	1	20	6	2	75	36	81	24	87	27	19.0	1200.3	-917.3	3
			2	26	5	0	78	0	55	0	62	0	15.1	1200.3		10
	2	1	1	26	6	0	82	0	79	0	78	0	20.1	1200.2		10
			2	31	9	0	83	0	80	0	76	0	37.7	1200.4		10
		2	1	26	4	0	83	0	66	0	71	0	14.8	1200.3		10
			2	32	2	0	87	0	30	0	33	0	4.2	1200.4		10
		3	1	26	6	0	79	0	86	0	91	0	27.2	1201.2		10
			2	33	6	0	83	0	59	0	67	0	22.7	1200.5		10

Tablolar incelendiğinde öncelikle göze çarpan nokta günümüzde en gelişmiş solver olan CPLEX 12.1 kullanıldığında bile 50 iş ve daha büyük problemlerin 1200 saniye limiti içerisinde optimal olarak çözülememesidir. Geliştirilen büyük bir süre avantajına sahiptir. Algoritma tüm problemlerde çok kısa sürelerde çözüme ulaşmaktadır; en yüksek çözüm süresi yaklaşık 1 dakikadır. Dolayısıyla geliştirilen algoritma süre açısından çok etkin bir algoritmadır ve daha büyük problemlere de rahatlıkla uygulanabileceği görülmektedir.

Düşük makine maliyet seviyelerinde daha fazla çözüm alternatifi olduğundan algoritmanın biraz daha fazla zaman aldığı gözlenmektedir. Aynı şekilde standby zamanı veya işlem süresi arttığında da çözüm süresinin çözüm alternatif sayısıyla doğru orantılı olarak arttığı görülmektedir. İş ağırlık seviyelerinin problem çözüm süresini kayda değer nitelikte etkilemediği, çözüm süresinin iş sayısıyla arttığı gözlenmektedir.

Tablolarda göze çarpan bir başka önemli nokta CPLEX'in zaman limiti sebebiyle optimali bulamadan çıktığı durumlarda algoritmanın bulduğu çözümlerin CPLEX'in bulduğu en iyi çözümlerden çok daha iyi olmasıdır. Bu farklar eksi yüzde değerlerin büyüklüğüne bakarak anlaşılabilir. Özellikle büyük problemlerde pek çok problem için CPLEX hiçbir olurlu çözüm bulamadan durmuş, bu durumlar için fark hesaplanmamıştır. CPLEX'in çözüm üretebildiği durumlarda ise algoritmanın çok daha üstün kaliteli çözümleri çok daha kısa sürelerde ürettiği görülmektedir.

Algoritmanın ürettiği çözümler incelendiğinde işlenen iş yüzdesi, ağırlık yüzdesi ve makine kullanım oranları açısından çok daha iyi olduğu söylenebilir. Algoritma daha çok makineyi daha fazla oranlarda kullanarak daha iyi çözümler üretmiştir.

Makine maliyetlerinin artması beklenen şekilde kapasite artırımı kararını olumsuz yönde etkilemektedir. Bu durum tablolarda kullanılan ortalama makine sayılarının karşılaştırılmasıyla anlaşılabilir. Makine maliyetlerinin artmasının diğer bir sonucu da kullanım oranlarının yükselmesi şeklinde izlenmektedir. Maliyet arttıkça aynı makinede işlenen iş sayısı artmakta, makine kullanım oranları da yükselmektedir.

Algoritma ile üretilen çözümlerdeki makine kullanım oranları küçük problemlerde yaklaşık %50 seviyelerinde iken büyük problemlerde %85'lere kadar çıkmaktadır. Bu durum büyük problemlerde çakışma yapan çok sayıda iş olması ve bunların farklı makinelerde işlenmesi, dolayısıyla planlama periyodunda işlerin makineleri daha çok dolduracak şekilde planlanabilmesinden kaynaklanmaktadır. Aynı sebeple işlenen işlerin ve işlenen ağırlığın yüzdesi de büyük problemlerde daha fazla olmuştur.

İşlenen işlerin ve işlenen ağırlığın yüzdeleri karşılaştırıldığında işlenen ağırlığın daha fazla olduğu görülmektedir. Yeni eklenen makineler sayı olarak işlerin belli bir yüzdesini işleyebilmekte, fakat potansiyel getiri seviyesine bakıldığında daha yüksek bir yüzdeye erişmektedir. Bu durum $w = 1$ seviyesi için çok belirgin değildir çünkü bu seviyede işlerin ağırlıkları işlem sürelerine eşittir. Ancak diğer seviyelerde fark belirginleşmektedir; işlenebilecek iş alternatifleri arasından w_j/p_j oranı daha yüksek işler seçilmekte ve yüzdeler arasındaki fark bu sayede artmaktadır.

Sonuç olarak Algoritma(BDİÇ) 100 iterasyon ve tüm iyileştirme algoritmalarıyla birlikte uygulandığında küçük (20 işlik) problemlerde optimale çok yakın sonuçları bir saniyenin altındaki sürelerde üretmektedir. Daha büyük problemler için ise CPLEX'in 1200 saniye limitli çözümleriyle karşılaştırıldığında hem süre hem de çözüm kalitesi olarak çok üstün bir performans göstermektedir.

Algoritma(BDİÇ) aynı şekliyle 500 işlik problemler için de çalıştırılmıştır. Algoritma ve iyileştirme algoritmalarının performanslarını ölçümlemek için $c = 1$ seviyesindeki tüm problemler kullanılmıştır. Tablo 21'de bu problemler için iyileştirme algoritmalarının performans ölçümü için yapılan değerlendirmenin sonuçları yer almaktadır. Tablodaki *"Eniyi0"* - *"Eniyi3"* arası sütunlar Algoritma(BDİÇ)'nin verdiği çözümün algoritmanın hangi adımı sonucunda üretildiğini göstermektedir. Örneğin tablonun ilk satırı okunduğunda 10 problem örneğinden birinin hiçbir iyileştirme algoritması çalıştırılmadan en iyi çözüme ulaştığı, birinin çözümünün *Algoritmaİyileştirme1* sonucunda elde edildiği, sekizinin çözümününse *Algoritmaİyileştirme3* sonucunda üretildiği görülmektedir. Bu sütunlardan iyileştirme algoritmalarının etkinliği rahatlıkla gözlenmektedir. En çok sayıda çözümün *Algoritmaİyileştirme3* sonucunda üretildiği görülmektedir, ancak değerlendirme yapılırken tüm iyileştirme algoritmalarının sırasıyla uygulandığına dikkat edilmelidir. Yani *Algoritmaİyileştirme3* kendi başına değil *Algoritmaİyileştirme2*'den sonra etkilidir. Bu konuya az sonra tekrar değinilecektir.

Tablo 21. İyileştirme algoritmaları performans ölçüm sonuçları.

n	*b-r*	*w*	*p*	Eniyi0	Eniyi1	Eniyi2	Eniyi3	Gel1 %	Gel2 %	Gel3 %	Süre0 %	Süre1 %	Süre2 %	Süre3 %	Süre (sn.)
20	1	1	1	1	1	0	8	6.2	0.0	8.1	12.0	0.0	69.7	18.3	0.1
			2	1	0	0	9	0.5	1.8	74.2	0.8	0.0	82.7	16.5	0.1
		2	1	3	0	0	7	1.5	0.0	21.2	13.2	0.0	64.5	12.3	0.1
			2	7	0	0	3	0.0	0.0	52.9	11.6	5.0	76.4	7.0	0.1
		3	1	1	1	0	8	0.0	0.0	0.5	9.5	2.5	59.3	18.7	0.1
			2	3	1	1	5	0.0	3.1	0.2	7.3	0.0	92.7	0.0	0.1
	2	1	1	0	0	0	10	3.5	0.0	18.0	1.3	3.9	57.6	27.3	0.1
			2	0	0	0	10	0.8	0.3	11.7	0.0	5.3	82.3	12.3	0.1
		2	1	0	0	0	10	8.9	0.0	8.5	8.3	0.0	46.7	25.0	0.0
			2	2	0	0	8	40.0	0.0	81.8	11.5	0.8	71.4	16.3	0.1
		3	1	0	0	0	10	1.4	0.0	3.4	8.3	2.5	52.5	16.7	0.1
			2	0	1	0	9	3.8	0.0	7.0	8.5	0.0	68.5	23.0	0.1
50	1	1	1	0	0	1	9	1.3	10.4	10.9	1.9	0.0	83.3	14.9	0.4
			2	0	0	2	8	0.0	14.4	3.4	5.3	2.0	86.9	5.8	0.8
		2	1	0	0	0	10	1.0	13.8	6.8	1.8	0.6	87.0	10.6	0.6
			2	1	0	1	8	1.9	12.0	13.5	8.3	3.2	71.2	17.3	0.3
		3	1	0	0	3	7	0.0	9.1	3.3	2.5	0.4	89.0	8.1	0.6
			2	0	0	2	8	0.0	8.8	3.2	4.3	1.4	87.9	6.3	0.6
	2	1	1	0	0	0	10	0.8	31.1	5.9	3.3	0.6	84.4	11.7	0.7
			2	0	0	0	10	0.3	19.2	6.8	4.3	2.5	86.9	6.4	1.1
		2	1	0	0	1	9	0.1	13.0	7.2	4.0	2.4	78.8	14.9	0.6
			2	0	0	0	10	2.0	12.6	9.6	4.2	2.5	80.3	12.9	0.5
		3	1	0	0	0	10	0.0	9.2	2.0	3.1	0.4	86.2	10.3	0.8
			2	0	0	1	9	0.4	10.0	3.2	6.6	2.9	81.9	8.6	0.8
100	1	1	1	0	0	0	10	0.1	28.6	3.9	2.7	0.8	87.8	8.6	3.2
			2	0	0	0	10	0.3	18.1	3.7	2.6	1.0	89.9	6.5	5.2
		2	1	0	0	0	10	0.0	14.0	2.9	2.1	0.7	88.9	8.3	2.9
			2	0	0	0	10	0.1	14.9	5.2	4.0	2.3	85.1	8.6	1.9
		3	1	0	0	2	8	0.2	9.5	1.4	2.0	0.5	91.6	5.9	4.0
			2	0	0	0	10	0.0	10.9	1.8	2.8	1.3	88.7	7.2	4.0
	2	1	1	0	0	0	10	0.0	27.3	3.8	1.9	1.2	88.6	8.4	4.7
			2	0	0	0	10	0.1	23.7	3.3	3.4	1.3	87.9	7.4	7.1
		2	1	0	0	0	10	0.5	16.9	3.5	2.4	0.6	86.9	10.0	3.4
			2	0	0	0	10	0.1	26.6	3.4	4.2	2.4	83.6	9.8	3.7
		3	1	0	0	1	9	0.0	9.0	1.1	1.8	0.7	90.7	6.8	5.0
			2	0	0	1	9	0.0	11.9	2.0	2.4	0.7	89.7	7.2	6.4
200	1	1	1	0	0	0	10	0.2	23.5	2.0	1.3	0.4	90.1	8.2	24.1
			2	0	0	0	10	0.2	16.9	1.1	1.7	0.4	90.8	7.1	39.9
		2	1	0	0	0	10	0.0	16.2	1.4	1.2	0.4	90.8	7.7	20.2
			2	0	0	1	9	0.1	19.6	2.8	1.3	0.7	90.5	7.5	21.2
		3	1	0	0	0	10	0.0	8.7	0.7	1.0	0.4	92.5	6.1	27.1
			2	0	0	2	8	0.0	9.7	0.9	1.7	0.5	91.6	6.2	36.0
	2	1	1	0	0	0	10	0.1	26.0	2.5	1.1	0.4	90.5	8.0	33.1
			2	0	0	0	10	0.0	20.7	1.9	1.8	0.5	90.4	7.2	60.8
		2	1	0	0	0	10	0.0	14.1	1.8	1.2	0.4	90.4	8.1	31.5
			2	0	0	2	8	0.1	22.4	1.5	1.7	0.7	89.2	8.4	29.6
		3	1	0	0	0	10	0.0	7.8	0.7	1.1	0.3	91.7	6.9	37.3
			2	0	0	0	10	0.0	9.7	1.1	1.5	0.5	91.7	6.4	51.5
500	1	1	1	0	0	0	10	0.0	21.4	1.1	0.6	0.2	90.1	9.1	408.5
			2	0	0	0	10	0.0	12.1	1.1	1.0	0.3	88.3	10.4	786.2
		2	1	0	0	1	9	0.0	14.4	0.6	0.5	0.2	91.5	7.8	414.8
			2	0	0	0	10	0.0	17.9	1.0	0.7	0.3	90.2	8.8	359.4
		3	1	0	0	0	10	0.0	7.1	0.3	0.6	0.2	91.9	7.4	462.2
			2	0	0	6	4	0.0	9.2	0.1	0.7	0.2	91.2	7.9	775.1
	2	1	1	0	0	0	10	0.0	21.8	1.3	0.6	0.2	91.6	7.7	687.5
			2	0	0	0	10	0.0	16.6	1.2	1.0	0.2	89.7	9.1	1143.8
		2	1	0	0	0	10	0.0	13.8	0.9	0.5	0.1	91.7	7.6	695.7
			2	0	0	2	8	0.0	17.8	0.8	0.7	0.2	92.0	7.1	729.8
		3	1	0	0	0	10	0.0	6.0	0.4	0.5	0.1	92.5	6.9	803.1
			2	0	0	1	9	0.0	9.1	0.4	0.7	0.2	92.6	6.5	1302.8

Tablodaki *"Gel1%"* - *"Gel3%"* arası sütunlar iyileştirme algoritmalarının iyileştirme yapılmamış çözümü 10 problem örneğinde ortalama yüzde kaç geliştirdiğini göstermektedir. Örneğin tablonun son satırı okunduğunda *Algoritmaİyileştirme1* sonucunda amaç fonksiyonunda hiç iyileşme olmadığı, buna karşılık sonrasında çalıştırılan *Algoritmaİyileştirme2* sonucunda %9.1'lik bir iyileşme gerçekleştiği, son olarak çalıştırılan *Algoritmaİyileştirme3* ile %0.4'lük bir iyileşme daha gözlendiği anlaşılmaktadır. Bu sütunlardan en fazla iyileşmenin *Algoritmaİyileştirme2* yardımıyla elde edildiği görülmektedir, bunu sırasıyla *Algoritmaİyileştirme3* ve *Algoritmaİyileştirme1* izlemektedir.

Tablodaki *"Süre0%"* - *"Süre3%"* arası sütunlar algoritmanın iyileştirme olmadan ve iyileştirmelerle birlikte çalıştığı toplam sürenin yüzde bazında ortalama dağılımını göstermektedir. Örneğin tablonun son satırı okunduğunda *Algoritmaİyileştirme2*'nin çalışma süresi Algoritma(BDİÇ)'nin toplam çalışma süresinin %92.6'sını oluşturmuştur. Buna karşılık hiçbir iyileştirme algoritması olmadan sadece rassallaştırılmış iterasyonların çalışması (Süre0) toplam zamanın %0.7'sini almıştır. *Algoritmaİyileştirme3* ise Algoritma(BDİÇ)'nin toplam çalışma süresinin %6.5'luk kısmını oluşturmuştur. Bu sütunlardan en fazla iyileşmeyi sağlayan *Algoritmaİyileştirme2*'nin aynı zamanda en fazla zamanı aldığı gözlemlenmektedir. Özellikle büyük problemlerde toplam çalışma süresinin %90'lara varan kısmını bu iyileştirme algoritması harcamaktadır.

Algoritma(BDİÇ)'nin 100 iterasyon ve tüm iyileştirme algoritmalarıyla birlikte uygulandığında toplam çalışma süresi saniye cinsinden *"Süre (sn.)"* sütununda gösterilmiştir. Algoritmanın en büyük problemler için bile 1200 saniye sınırını sadece tek bir problem grubu için aştığı gözlemlenmiştir.

Yapılan deneyler sonrasında mevcut durumda *Algoritmaİyileştirme2*'nin zamanının yüksek olması göz önüne alınmış ve $c = 1$ seviyesindeki tüm problemler kullanılarak Algoritma(BDİÇ) değişik iterasyon sayısı ve iyileştirme kombinasyonları ile denenmiştir. Bu deneyin sonuçları Tablo 22'de gösterilmektedir.

Algoritma(BDİÇ) öncelikle 100 iterasyon ve sadece *Algoritmaİyileştirme1* ve *Algoritmaİyileştirme3* ile denenmiştir (*Algoritmaİyileştirme2* uygulanmamıştır).Tüm problemler için tekrar çözüm alınmış ve *Algoritmaİyileştirme2*'nin izole performansı bu şekilde incelenmiştir. Tablo 22'deki *"İyi2yok %"* sütunu *Algoritmaİyileştirme2*'nin olmadığı durumda Algoritma(BDİÇ)'nin amaç fonksiyonundaki kötüleşmeyi göstermekte, *"Süre İyi2yok"* sütunu ise bu durumda algoritmanın çözüm süresini vermektedir. Özellikle büyük problemlerde *Algoritmaİyileştirme2*'nin %5'e yaklaşan değerlerde iyileşme sağladığı böylece gözlemlenmiştir. Ancak bu iyileşme yüksek bir çalışma süresi maliyetini beraberinde getirmektedir. Süre farkının daha iyi gözlemlenebilmesi için Tablo 22'in son sütununda *Algoritmaİyileştirme2*'siz sürelerle *"Süre (sn.)"* sütunundaki süreler karşılaştırılmalıdır. *Algoritmaİyileştirme2* olmadan Algoritma(BDİÇ) çok daha kısa sürmektedir.

Algoritma(BDİÇ)'nın performansını *Algoritmaİyileştirme2* olmadan geliştirmenin mümkün olup olmadığı bu sefer iterasyon sayısı artırılarak denenmiş, algoritma 500 iterasyon ve sadece *Algoritmaİyileştirme1* ve *Algoritmaİyileştirme3* ile denenmiştir. Tablo 22'deki *"500İyi2yok %"* sütunu bu durumda amaç fonksiyonundaki kötüleşmeyi (orijinal durumla karşılaştırmalı olarak) göstermekte, *"Süre 500İyi2yok"* sütunu ise bu durumda algoritmanın çözüm süresini vermektedir. Yeni durumda algoritma ilk durumuna göre yaklaşık yarı sürelerde çözüm üretmektedir, dolayısıyla *Algoritmaİyileştirme2*'nin çözüm süresine iterasyon sayısından çok daha fazla etki ettiği buradan da anlaşılabilir. Küçük problemler için alınan eksi yüzde değerler 500 iterasyonlu durumda daha iyi çözümler elde edildiğini göstermektedir. Büyük problemlerde ise orijinal durum (100 iterasyon ve tüm iyileştirme algoritmaları varken) %4'e varan iyileştirmeler sağlamıştır.

Tablo 22. Algoritma(BDİÇ)'nin değişik versiyonları için karşılaştırma sonuçları.

n	*b-r*	*w*	*p*	Süre (sn.)	Imp2 yok %	Süre Imp2yok	500Imp2 yok %	Süre 500Imp2yok	500Imp yok %	Süre 500Impyok	500Imptek %	Süre 500Imptek
20	1	1	1	0.1	0.0	0.0	0.0	0.1	1.3	0.1	1.3	0.1
			2	0.1	0.0	0.0	-0.5	0.1	6.0	0.1	5.5	0.1
		2	1	0.1	0.0	0.0	0.0	0.2	2.0	0.1	2.0	0.1
			2	0.1	0.0	0.0	0.0	0.3	0.0	0.1	0.0	0.1
		3	1	0.1	0.0	0.0	-0.2	0.1	0.1	0.1	-0.2	0.1
			2	0.1	0.0	0.0	-2.2	0.2	-1.5	0.1	-2.0	0.1
	2	1	1	0.1	0.0	0.0	-0.9	0.1	4.4	0.1	2.2	0.1
			2	0.1	-0.7	0.0	-0.9	0.3	5.5	0.1	4.8	0.1
		2	1	0.0	0.0	0.0	-0.2	0.2	10.5	0.1	6.0	0.1
			2	0.1	0.0	0.0	-13.1	0.3	10.2	0.1	10.2	0.1
		3	1	0.1	0.0	0.0	-0.4	0.2	1.7	0.1	1.4	0.1
			2	0.1	0.0	0.0	-0.7	0.2	4.9	0.1	3.1	0.1
50	1	1	1	0.4	2.0	0.1	-0.2	0.6	15.2	0.2	8.3	0.2
			2	0.8	1.6	0.2	-0.5	0.9	12.1	0.5	5.3	0.5
		2	1	0.6	0.1	0.1	-1.1	0.5	11.2	0.3	6.4	0.3
			2	0.3	1.8	0.1	-1.5	0.6	12.0	0.2	11.2	0.3
		3	1	0.6	0.3	0.1	-0.1	0.6	8.8	0.3	3.1	0.4
			2	0.6	2.2	0.2	0.5	0.7	8.2	0.3	2.8	0.4
	2	1	1	0.7	1.1	0.2	-2.3	0.8	17.6	0.3	5.5	0.4
			2	1.1	1.6	0.3	-2.5	1.1	17.2	0.6	8.2	0.9
		2	1	0.6	2.0	0.2	0.9	0.8	13.8	0.2	7.9	0.3
			2	0.5	1.6	0.2	-1.1	0.7	12.5	0.3	6.9	0.6
		3	1	0.8	0.8	0.2	0.0	0.7	8.9	0.3	2.8	0.4
			2	0.8	0.2	0.2	-1.8	0.9	9.6	0.4	5.4	0.7
100	1	1	1	3.2	1.3	0.6	-1.5	2.7	20.6	0.7	4.6	0.9
			2	5.2	3.0	0.8	0.9	4.1	14.9	1.3	6.0	1.6
		2	1	2.9	2.4	0.5	1.0	2.3	13.1	0.7	4.0	0.9
			2	1.9	3.0	0.4	-0.2	1.8	13.8	0.7	4.4	0.8
		3	1	4.0	1.5	0.6	0.7	2.5	8.8	0.9	3.1	1.1
			2	4.0	3.0	0.7	2.0	2.9	10.0	1.1	2.5	1.4
	2	1	1	4.7	3.3	0.7	1.9	3.6	21.8	0.9	5.4	1.1
			2	7.1	2.1	1.2	0.7	5.6	18.5	1.7	7.1	2.0
		2	1	3.4	2.0	0.6	1.0	3.0	15.6	0.8	4.1	1.0
			2	3.7	4.0	0.7	0.9	3.3	18.1	0.9	7.0	1.2
		3	1	5.0	1.3	0.7	0.6	3.5	8.1	1.0	3.7	1.2
			2	6.4	2.0	0.9	0.2	4.3	10.7	1.5	5.0	1.8
200	1	1	1	24.1	2.5	3.2	1.9	16.7	18.5	3.1	2.6	3.5
			2	39.9	2.0	5.0	1.8	24.8	13.4	5.6	2.8	6.5
		2	1	20.2	2.0	2.6	1.3	13.2	13.6	2.6	1.9	3.1
			2	21.2	4.7	2.2	3.0	11.8	15.6	2.9	4.9	3.4
		3	1	27.1	1.6	2.9	1.4	14.9	7.6	3.3	1.9	3.9
			2	36.0	2.7	4.3	1.8	19.5	8.8	4.7	2.2	5.6
	2	1	1	33.1	4.1	4.1	2.4	21.4	20.1	4.0	4.2	4.3
			2	60.8	3.1	7.4	1.5	39.2	16.6	8.2	3.2	8.9
		2	1	31.5	2.3	3.8	1.3	19.1	12.5	3.5	2.2	3.9
			2	29.6	2.5	3.5	1.1	18.4	16.2	3.6	3.6	4.4
		3	1	37.3	0.8	4.0	0.4	20.4	6.8	4.0	1.5	4.5
			2	51.5	2.2	5.4	1.3	28.0	9.3	6.0	1.9	7.2
500	1	1	1	408.5	4.8	48.8	4.2	248.3	17.1	24.1	2.9	27.9
			2	786.2	5.0	94.2	3.9	486.5	11.8	52.7	3.9	62.5
		2	1	414.8	3.6	41.7	3.0	203.2	12.1	21.8	2.3	26.2
			2	359.4	5.5	31.9	4.3	157.0	14.6	19.9	3.5	24.0
		3	1	462.2	1.5	44.3	1.5	226.5	6.1	25.4	1.3	30.8
			2	775.1	3.1	72.0	2.6	355.5	8.0	41.7	1.5	49.6
	2	1	1	687.5	4.4	63.9	3.7	328.5	17.8	30.5	3.4	37.0
			2	1143.8	2.7	134.2	1.9	667.0	14.7	68.4	3.5	80.8
		2	1	695.7	2.5	56.4	2.0	284.5	12.0	26.8	1.6	32.6
			2	729.8	4.2	55.5	3.3	292.9	14.7	30.0	4.2	35.9
		3	1	803.1	0.9	60.4	0.8	311.2	5.3	32.6	1.2	39.8
			2	1302.8	1.9	98.9	1.5	491.1	8.4	52.8	2.2	62.9

Diğer iki iyileştirmenin işe yararlığını tespit etmek amacıyla algoritma 500 iterasyon ve hiçbir iyileştirme algoritması olmadan denenmiştir. Tablo 22'deki *"500İyiyok %"* sütunu bu durumda Algoritma(BDİÇ)'nin amaç fonksiyonundaki kötüleşmeyi (orijinal durumla karşılaştırmalı olarak) göstermekte, *"Süre 500İyiyok"* sütunu ise çözüm süresini vermektedir. Yeni durumda algoritma en büyük problemler için bile bir dakikanın altında çözüm üretmektedir, fakat çözüm kalitesinde %20'lere varan bir kötüleşme söz konusudur.

Son olarak Algoritma(BDİÇ) 500 iterasyon ve tüm iyileştirme algoritmalarıyla tekrar denenmiştir. Ancak bu sefer iyileştirme algoritmaları her iterasyonda değil tüm iterasyonlar tamamlandıktan sonra tek bir kez uygulanmıştır. Tablo 22'deki *"500İyitek %"* sütunu bu durumda amaç fonksiyonundaki kötüleşmeyi (orijinal durumla karşılaştırmalı olarak) göstermekte, *"Süre 500İyitek"* sütunu ise çözüm süresini vermektedir. Büyük problemlerde tüm kombinasyonlar içinde orijinal duruma en yakın sonuçları bu kombinasyon üretmiştir ve sürelerin yine çok kısa olduğu göze çarpmaktadır.

BDİÇ probleminin çözümü sonrası verilen kararın taktik bir karar olduğu (makine alma, kapasite artımı) göz önüne alındığında karar vericinin algoritmanın daha uzun sürelerde daha iyi çözüm üretmesine razı olması beklenen bir durumdur. Yapılan deneyler sonucunda ortaya çıkan durum şuna işaret etmektedir: Eğer çok kısa sürede çözüm isteniyorsa veya küçük problemler için çalıştırılacaksa Algoritma(BDİÇ)'nin 500 (veya daha fazla) iterasyon ve tüm iyileştirme algoritmalarının tek bir kez kullanımı ile (son durum) çalıştırılması, diğer durumlar için ise 100 iterasyon ve tüm iyileştirme algoritmalarıyla birlikte uygulanması daha uygun olacaktır. Daha uzun sürelere razı olunduğu durumda daha fazla iterasyon yapmaya karar vermek ise karar vericiye bağlıdır.

4.6. KARAR DESTEK SİSTEMİ

Projenin tüm dönemlerinde üzerinde çalışılan problem ve çözüm yöntemleri projenin son döneminde geliştirilen karar destek sisteminin ana model yapısını oluşturmaktadır. Bu amaçla algoritma için yazılan kodlar ve derlenen programlar karar destek sisteminin (KDS) içine entegre edilmiştir. Modern bir karar destek sisteminde bulunması gereken diğer öğeler olan arayüz ve veri tabanı çalışmaları projenin son döneminde tamamlanmıştır. Arayüzün ve modellere ait çözüm kütüphanelerinin geliştirilmesi ve birbirine bağlanmasında MS Visual Studio 2008 ortamında C# programlama dili ve IBM ILOG CPLEX'in ilgili C# kütüphaneleri kullanılmıştır.

Projenin ikinci ve üçüncü dönemlerinde yazılım geliştirme çalışmaları olarak BTSİÇ, KTSİÇ ve ÇZB problemlerine ait modellerin optimal çözümü ve yine bu modellere ait kesin veya yaklaşık çözüm algoritmalarının kodlanması gerçekleştirilmiştir. Bu programlar CPLEX solver kütüphanesiyle bütünleşik olarak C# ve C++ programlama dilleriyle geliştirilmiştir. Matematiksel modellerin kodlanması ve çözümünde ise okunma ve güncelleme kolaylığı açısından IBM ILOG OPL 6.3 yüksek seviye model programlama dili tercih edilmiştir. Böylelikle sabit iş çizelgeleme problem sınıflarına ait model tabanı tamamlanmıştır.

Projenin son döneminde değişken iş çizelgeleme problemine kapasite planlama açısından yaklaşan BDİÇ problemine ait optimal çözümler için C# programlama dili kullanılarak CPLEX kütüphanesine bağlı bir program geliştirilmiştir. Yaklaşık fakat çok hızlı çözümler elde etmek amacıyla geliştirilen Algoritma(BDİÇ) için yazılan kod ticari yazılımlardan bağımsız olarak tasarlanmıştır, başka bir solver kütüphanesine bağımlılığı yoktur. Tüm problemlere ait sayısal deney sonuçlarının tablolar üzerinde excel makroları kullanılarak otomatize şekilde raporlanması sağlanmıştır.

SİÇ problemlerinin optimal ve sezgisel yöntemlerle çözümleri için projenin son üç döneminde geliştirilen problem sınıflarına ait örneklerin çözümlerinde kullanılacak tüm kodlar karar destek sisteminin beyni olarak adlandırılabilecek model tabanını oluşturmuştur. Önceki dönemlerde ve son dönemde gerçekleştirilen sayısal deneylerde bağımsız olarak kullanılan program kodları KDS içinde bütünleşik hale getirilmiştir..

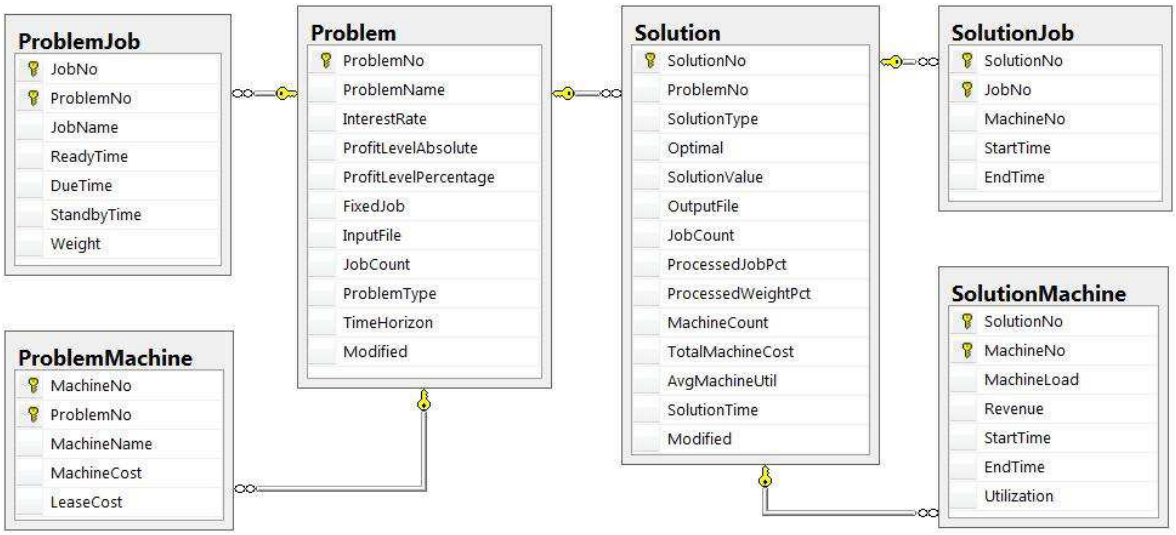
Tüm problem tiplerinin çözümünde esas alınacak çözüm kütüphaneleri IBM ILOG CPLEX solver paketine bağlı Windows işletim sistemi için derlenmiş ve C# ve C++ programlama dilleriyle geliştirilmiş algoritma yazılım kütüphanelerinden oluşmuştur. Sadece ÇZB problemi için 500 ve daha az iş içeren problemlerde geliştirilen algoritmanın yanında çok kısa sürede sonuç verdiğinden doğrudan optimal çözüme ait matematiksel modeller de çözdürülebilmektedir. Daha yüksek sayıda iş olduğunda ise sezgisel algoritmaya ait C++ modülü doğrudan aktif hale gelmektedir. Çalışılan diğer tüm problemler için KDS içinde geliştirilen algoritmalar kullanılmıştır.

Model tabanını oluşturan kütüphaneleredeki kod sınıfları içinde gömülü olarak tanımlı olan tüm problem tiplerine ait matematiksel modeller, ayrıca algoritmalarda kullanılan tüm değişken ve parametrelerle ilgili veri girişleri geliştirilen form arayüzleriyle kolaylıkla yapılabilmektedir. Bunun yanında benzer arayüzlerle problemin boyutu ve türüne bağlı olarak çözüm yöntemi ve çözüm çıktıları için değişik seçenekler belirleyen tüm parametrelerin girişi sağlanmaktadır. Böylelikle KDS arayüzü, Algoritma(BDİÇ) örneğinde olduğu gibi, yaklaşık çözüm veren bir algoritmanın optimale daha yakın veya daha hızlı sonuçlar vermesini etkileyen ayarların esnek bir şekilde yapılabilmesine olanak tanımaktadır.

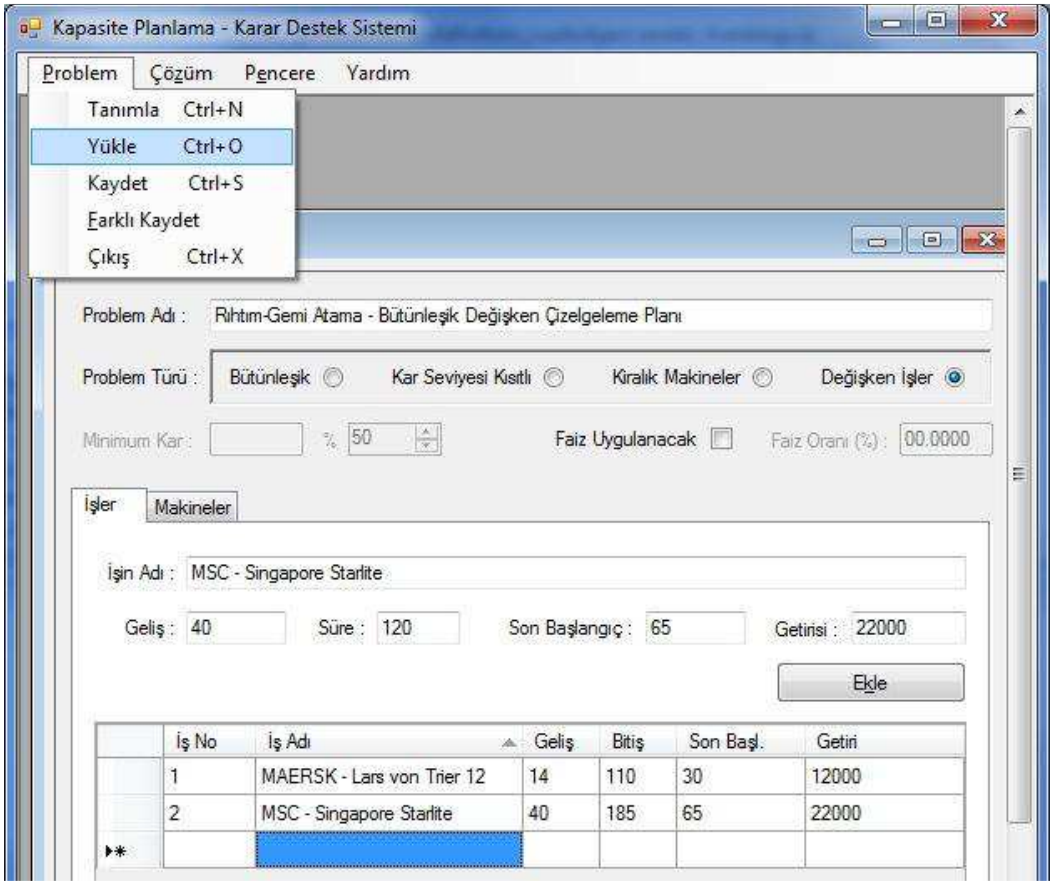
Problem girdi ve çıktılarının arayüz dışında da tanımlanabilmesi ve incelenebilmesi açısından düz metin (txt), XML ve Excel girdi ve çıktı dosya formatları kullanılmıştır. Proje süresince gerçekleştirilen sayısal deneylerde kullanılan düz metin girdi ve çıktı formatı karar destek sistemi tarafından desteklenmektedir. BTSİÇ türünde bir problem üzerinde bu dosya içeriklerinden örnekler ve problem/çözüm verilerinin XML formatlı yazımları EK 1'de verilmiştir. KDS kapsamında kullanılan XML şablonlarının geliştirilmesinde literatürdeki çizelgeleme kütüphanelerinde kullanılan örneklerden yararlanılmış (LiSA, 2012) ve proje için yeterli olacak daha spesifik bir uyarlama yapılmıştır.

Karar destek sisteminin model tabanı ve arayüzüyle bağlantılı olarak bahsedilen dosya tabanlı veri yapısına ek olarak bir ilişkisel veritabanı alternatifi de yaratılmıştır. İlişkisel veritabanı MS SQL Server 2008 üzerinde tasarlanmıştır. Bu veritabanı aynı problemin parametrelerinin güncellenmiş hallerini üretmede ve zaman içindeki farklı çözümleri daha hızlı ve etkin bir şekilde karşılaştırmada kullanılabilmektedir. Bir problem sınıfından diğerine farklılık gösteren, karar vericinin şart ve tercihlerine bağlı olarak değişebilecek genel ve spesifik program parametrelerinin daha verimli şekilde saklanıp güncellenmesi sağlanmıştır. Böylelikle benzer problem senaryolarının tanımı ve farklı modellerde çözümü için zaman kaybını önleyecek esnek bir veritabanı tasarımı ve arayüz bağlantısı geliştirilmiştir. Şekil 5'te KDS için tasarlanan veritabanı yapısında kullanılan tablolar arasındaki ilişkiler ve türleri gösterilmektedir. Veritabanının bulunmadığı durumlarda KDS yine metin ve XML veri dosyaları üzerinden çalıştırılabilecektir.

KDS için şu ana dek tanımlanmış problemlerin model yapılarını destekleyen veritabanı ve arayüz bileşenleri tasarlanmıştır. Problemin türünü, boyutunu, iş ve makinelere ait değerleri, çözüm yöntemi ve çıktıları için alternatifleri belirleyen tüm parametrelerin Şekil 6'da gösterildiği gibi hem en baştan girişi, hem de daha önceden belirli biçemlerde tanımlanmış düz metin, XML veya Excel formatlı dosyalardan yüklenebilmeleri tasarlanmıştır.

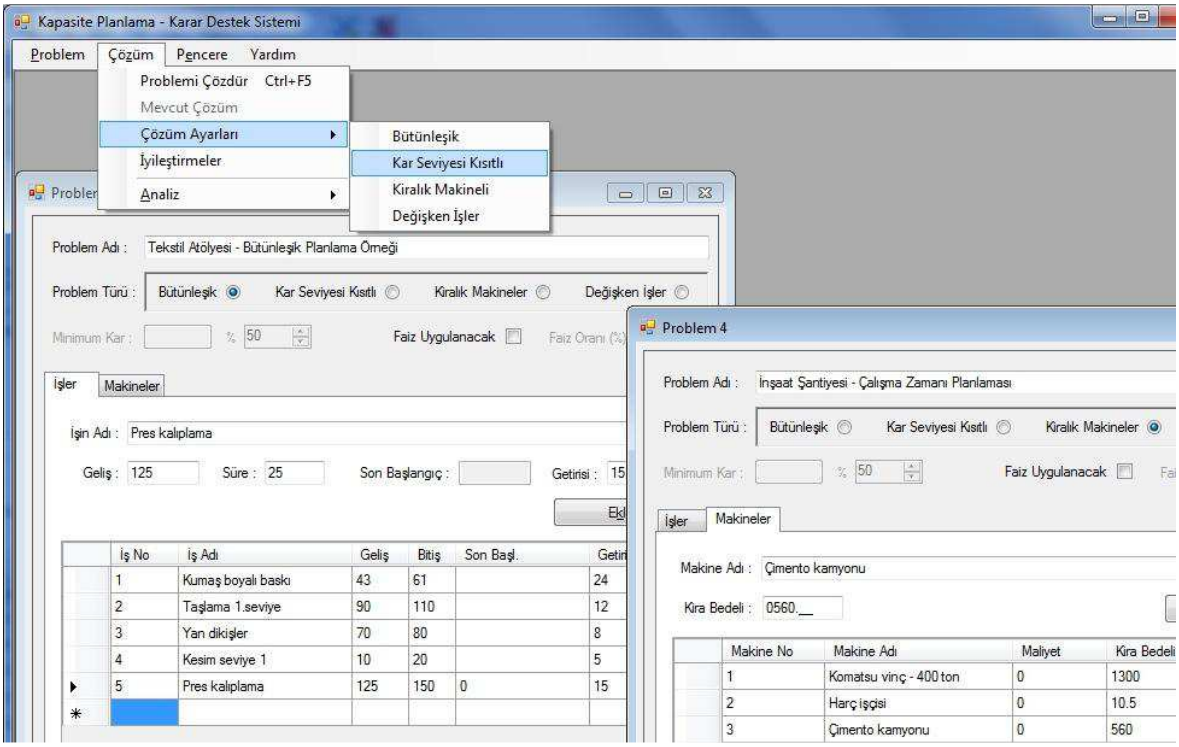


Şekil 5. Karar Destek Sistemi veritabanının tablo ilişkileri diyagramı.



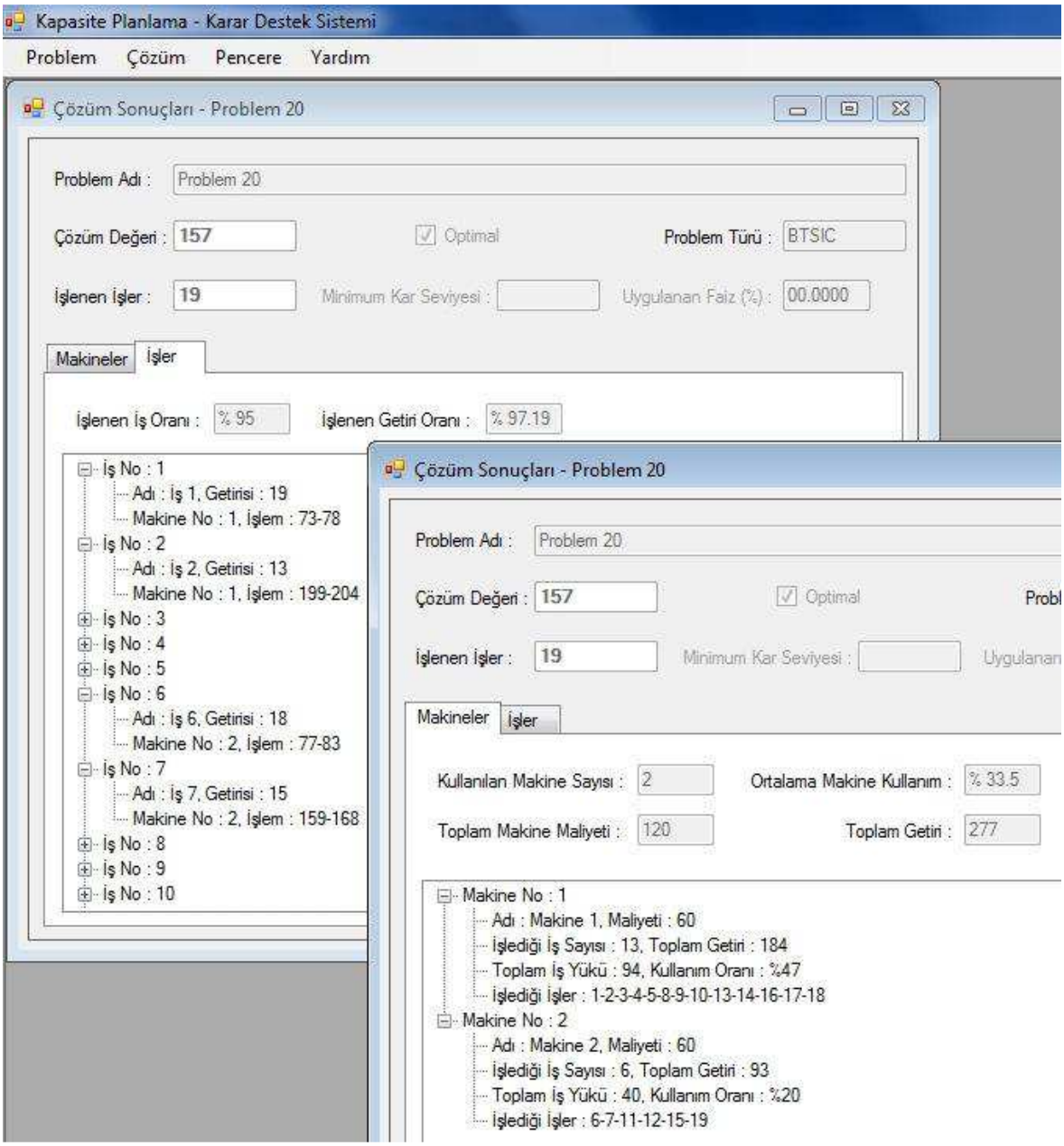
Şekil 6. Geliştirilen Karar Destek Sistemi arayüzünde problem ana menüsü.

Şekil 7'de farklı iki problem türüne ait iş ve makine veri giriş örnekleri gösterilmektedir. Şekil 8'de ise EK 1'de girdi verisi sunulan BTSİÇ problem örneği için KDS çözüm sonuçları arayüzü gösterilmektedir. Kullanıcının kolaylıkla veri girişi yapabilmesini sağlayan, aynı zamanda çözüm sonuçlarını incelemesine olanak tanıyan bu arayüzün geliştirilmesinde C# programlama dili kullanılmıştır.



Şekil 7. Geliştirilen Karar Destek Sistemi arayüzünde problem tanımları ve çözüm ana menüsü.

Arayüz tasarımında aynı anda birden fazla problemin düzenlenebilmesi, incelenebilmesi, çözümlerin karşılaştırılabilmesi ve aralardaki pencere geçişlerinin daha anlaşılır olması için ticari yazılımların çoğunda kullanılan Çoklu Döküman Arayüz (Multiple-Document Interface - MDI) formları kullanılmıştır.



Şekil 8. Karar Destek Sistemi çözüm sonuçları arayüzü.

Model tabanındaki yeniliklerin aynı arayüzde farklı menü seçenekleriyle kullanılabilmesi sağlanmıştır. Böylelikle karar destek sisteminin arayüz açısından da modülerliği ve yeniden kullanılabilirliği artırılmıştır. Geliştirilen karar destek sistemi kullanıcılara problemlerini kolaylıkla tanımlayıp çözüm alabilmeleri için gerekli ortamı sağlamaktadır.

5. SONUÇ VE ÖNERİLER

Bu projede gerçekleştirilen çalışmayla literatürde önemli bir yer tutan ve birçok pratik uygulaması olan aralık çizelgelemede kapasite planlama problemleri daha önce yaklaşılmamış bir açıdan ele alınmış, problemler için doğruluk ve geçerlilik seviyesi yüksek, uygulamada katkı sağlayacak modeller geliştirilmiştir. Aynı derecede önemli ancak daha önce ele alınmamış bir problem olan kapasite artırım problemi de ilk problem için bağlantılı şekilde ele alınmıştır. Böylece genel anlamda herhangi bir rezervasyon sisteminde herhangi bir anda eldeki makine (veya genel anlamda kaynak) sayısının belirli bir miktar artırımıyla elde edilebilecek ek karın hesaplanması mümkündür.

Projede sabit ve değişken iş çizelgelemede yer alan değişik kapasite planlama ve kapasite artırım problemleri için yeni ve özgün matematiksel modeller ve etkin/verimli çözüm yöntemleri geliştirilmiştir. Bu bağlamda başarısı kanıtlanmış bazı yöntemler, problemin yapısal özelliklerinden faydalanacak yeni ve özgün yöntemlerle güçlendirilerek kullanılmıştır. Çalışma sonucunda ortaya çıkan modeller ve çözüm yöntemleri incelenecek problemlere özgü yeni ve güçlü yaklaşımlar içermektedir. Bu sayede hem konuyla ilgili literatüre hem de problemlerin pratikteki uygulamalarına somut katkılar sağlanması hedeflenmiştir.

Daha önceki bölümlerde belirtildiği gibi imalat sistemlerinde makine/iş atamasında çok sık karşılaşılan bir problem olan aralık çizelgeleme ve bu sistemlerin kapasite planlaması ve kapasite artırımı problemleri aynı zamanda hizmet sektöründe de oldukça geniş bir uygulama sahasına sahiptir. Önerilen proje ile geliştirilen model ve çözümlerin daha önce üzerinde çalışmış olduğu bazı gerçek hayat problemleri üzerinde doğrudan etkisi olacaktır. Bu problemler limanlarda gemi-rıhtım atanmaları, araç filosu kapasite planlaması gibi rezervasyon sistemlerinin tipik örneklerini oluşturan problemlerdir. Bu açıdan proje çıktıları gelecekteki uygulama projeleri için çok değerli olacak teorik birikimini sağlamaktadır.

Projede yaratılan özgün model ve çözüm yöntemlerinin performans açısından eniyi çözüme yakınlık ve çözüm zamanı gibi kriterlerde üstün yöntemler olduğu sayısal deneylerle ispatlanmıştır. Ayrıca analizleri kolaylaştırabilmek, verimli çözümler arasında seçim yapabilmek ve duyarlılık analizlerini gerçekleştirebilmek için tüm model ve çözüm yöntemlerini bir arayüzle entegre eden bir karar destek sistemi uygulaması yapılmıştır. Tasarımı ve uygulaması yapılan karar destek sistemiyle analizler kolay bir hale getirilmiştir, bu sistem çözümlerde etkin bir araç olarak kullanılabilecektir. Bunun yanında geliştirilen yöntemlerden gelecekte benzer problemler için de yararlanılabilecektir.

Proje boyunca proje yürütücüsü Orta Doğu Teknik Üniversitesi'nde Yüksek Lisans ve Doktora eğitimini tamamlayıp kazandığı deneyim ve bilgiyi şu anda çalışmakta olduğu İzmir Ekonomi Üniversitesi'ne aktarabilmiştir; bu yapılan bilimsel katkı açısından önemlidir. Proje kapsamındaki çalışmalarla ve projeden edinilen deneyim ve gözlemlerden esinlenecek yeni çalışmalarla yeni araştırmacılar yetişecek ve araştırma ve uygulama alanında yeni işbirlikleri doğabilecektir. Proje içeriğinden hareketle hazırlanan bildiri ve makaleler ulusal bilimsel birikime katkıda bulunmuş, ülkemizin uluslararası düzeyde bilimsel açıdan temsil edilmesine katkı sağlamış ve diğer araştırmacılarla bilgi paylaşımını mümkün kılmıştır. Tüm bu açılardan projenin hedeflerine eriştiği rahatlıkla söylenebilmektedir.

Bu projede ortaya çıkarılan teorik birikimin gelecekte özellikle sanayi uygulamaları içeren projelerde kullanılması mümkün olabilecektir. Uygulamalı projelerde bu projede ortaya konulan yeni model ve çözüm yöntemleri bir başvuru kaynağı olacak, geliştirilen teorik altyapıdan yararlanarak daha kompleks sistemler incelenebilecektir. Geliştirilen karar destek sisteminin de gelecek çalışmalarda fayda sağlaması beklenmektedir. Bu karar destek sistemi uygulama projelerinde mevcut kurumsal kaynak planlama paket programlarına entegre

edilerek kullanılabilir. Bu sistem giriş bölümünde belirtildiği gibi özellikle atölye tipi üretim yapan firmaların kullanımı için çok faydalı olacaktır.

Projede geliştirilen teorik altyapıdan hareketle incelenen problemlerin yeni kısıtlar altındaki özel halleri için (örneğin işleyebilirlik kısıtları) yüksek lisans tezleri ve doktora tezleri üretilebilecektir. Bu yeni çalışmaların yine ulusal bilimsel birikime katkıda bulunması hedeflenecektir. Çalışmaların sonuçları karar destek sistemine eklenebilecektir.

Sonuç olarak proje kapsamında aralık çizelgelemede kapasite planlama konusunda çeşitli problemler için özgün çalışmalar gerçekleştirilmiş, bu çalışmalar bilimsel konferans kitaplarında ve dergilerde yayınlanmış, ayrıca tüm bileşenleri bir araya getiren bir karar destek sistemi geliştirilmiştir.

KAYNAKÇA

ARKIN A.M., Silverberg E.L., Scheduling Jobs with Fixed Start and End Times, *Discrete Applied Mathematics*, 18, 1-8, (1987).

BENHAMOU, F., Jussien, N., O'Sullivan, B., *Trends in constraint programming*, ISTE: London, (2007).

BOUZINA K.I., Emmons H., Interval Scheduling on Identical Machines, *Journal of Global Optimization*, 9, 379-393, (1996).

ELİİYİ D.T., *Operational Fixed Job Scheduling Problem*, (Doktora Tezi), Endüstri Mühendisliği Bölümü, Orta Doğu Teknik Üniversitesi, Türkiye, (2004).

ELİİYİ D.T., Azizoğlu M., Approximation Algorithms for Operational Fixed Job Scheduling Problems, *Proceedings of 35th International Conference on Computers & Industrial Engineering*,1, İstanbul, (2005) pp:567-572.

ELİİYİ D.T., Azizoğlu M., Spread time constraints in operational fixed job scheduling, *International Journal of Production Research*, 44, 4343-4365, (2006).

ELİİYİ, D.T., Kandiller, L., A Decision Support System for the Cell Formation Problem, International *Journal of Industrial and Systems Engineering*, 3, 348-367, (2008).

ELİİYİ D.T., Örnek A., Karakütük S.S., A Vehicle Scheduling Problem with Fixed Trips and Time Limitations, *International Journal of Production Economics*, 117, 150-161, (2009a).

ELİİYİ D.T., Korkmaz A.G., Çiçek A.E., Operational Variable Job Scheduling with Eligibility Constraints: A Randomized Constraint-Graph-Based Approach, *Technological and Economic Development of Economy*, 15, 245 – 266, (2009b).

ELİİYİ D.T., Azizoğlu M., A Fixed Job Scheduling Problem with Machine-Dependent Job Weights, *International Journal of Production Research*, 47, 2231 – 2256, (2009c).

ELİİYİ, D.T., Integrated Capacity Expansion and Scheduling Decisions in a Sewing Workshop, Tekstil ve Konfeksiyon, 20, 366-372, (2010a).

ELİİYİ, D.T., Azizoğlu, M. Working Time Constraints in Operational Fixed Job Scheduling, *International Journal of Production Research*, 48, 6211-6233, (2010b).

ELİİYİ, D.T., Azizoğlu, M., Heuristics for operational fixed job scheduling problems with working and spread time constraints, *International Journal of Production Economics*, *International Journal of Production Economics*, 132, 107-121, (2011).

FAIGLE, U., Kern, W., Nawijn, W.M., A greedy online algorithm for the k-track assignment algorithm, *Journal of Algorithms*, 31, 196-210, (1999).

FISCHETTI M., Martello S., Toth P., The Fixed Job Schedule Problem with Spread-Time Constraints, *Operations Research*, 35, 849-858, (1987).

FISCHETTI M., Martello S., Toth P., The Fixed Job Schedule Problem with Working-Time Constraints, *Operations Research*, 37, 395-403, (1989).

FISCHETTI M., Martello S., Toth P., Approximation Algorithms for Fixed Job Schedule Problems, *Operations Research*, 40, S96-S108, (1992).

GABREL, V., Scheduling jobs within time windows on identical parallel machines, *European Journal of Operational Research*, 83, 320-329, (1995).

GARCIA, J.M., Lozano, S., Production and delivery scheduling problem with time windows, *Computers & Industrial Engineering*, 48, 733 – 742, (2005).

GERTSBAKH, I., Stern, H.I., Minimal resources for fixed and variable job schedules, *Operations Research*, 26, 68-85, (1978).

HASHIMOTO A., Stevens J.E., Wire Routing by Optimizing Channel Assignments within Large Apertures, *Proceedings of the 8th Design Automation Workshop*, (1971), pp: 155-169.

HUANG, Q., Lloyd, E., Cost Constrained Fixed Job Scheduling, *Lecture Notes in Computer Science* (Theoretical Computer Science, 8th Italian Conference, ICTCS 2003, Proceedings, Ed.: Carlo Blundo and Cosimo Laneve), 2841, 111 – 124, (2003).

JAIN A.S., Meeran S., Deterministic job-shop scheduling: Past, present and future, *European Journal of Operational Research*, 113, 390-434, (1999).

KOLEN A.J.W., Kroon L.G., On the Computational Complexity of (Maximum) Class Scheduling, *European Journal of Operational Research*, 54, 23-38, (1991).

KOLEN A.J.W., Kroon L.G., License Class Design: Complexity and Algorithms, *European Journal of Operational Research*, 63, 432-444, (1992).

KOLEN A.J.W., Lenstra J.K., Papadimitriou C.H., Spieksma F.C.R., Interval scheduling: A survey, *Naval Research Logistics*, 54, 530 – 543, (2007).

KOVALYOV M.Y., Ng C.T., Cheng T.C.E., Fixed interval scheduling: Models, applications, computational complexity and algorithms, *European Journal of Operational Research*, 178, 331-342, (2007).

KROON L.G., *Job Scheduling and Capacity Planning in Aircraft Maintenance*, (Doktora Tezi), Rotterdam School of Management, Erasmus University, Hollanda, (1990).

KROON L.G., Salomon M., Van Wassenhove L.N., Exact and Approximation Algorithms for the Operational Fixed Interval Scheduling Problem, *European Journal of Operational Research*, 82, 190-205, (1995).

KROON L.G., Salomon M., Van Wassenhove L.N., Exact and Approximation Algorithms for the Tactical Fixed Interval Scheduling Problem, *Operations Research*, 4, 624-638, (1997).

LiSA - A Library of Scheduling Algorithms, [http://lisa.math.uni-magdeburg.de], Son erişim tarihi 18 Mart 2012.

ROJANASOONTHON S., Bard J.F., Reddy S.D., Algorithms for parallel machine scheduling: a case study of the tracking and data relay satellite system, *Journal of the Operational Research Society*, 54, 806-821, (2003).

ROJANASOONTHON S., Bard J.F., A GRASP for parallel machine scheduling with time windows, *INFORMS Journal on Computing*, 17, 32-51, (2005).

ROSSI, A., Singh, A., Sevaux, M., A Metaheuristic for the fixed job scheduling problem under spread time constraints, *Computers and Operations Research*, 37, 1045-1054, (2010).

ROSSI, F., van Beek, P., Walsh, T. *Handbook on Constraint Programming*, Elsevier: NY, (2006).

SOLYALI, O., Ozpeynirci, O., Operational fixed job scheduling problem under spread time constraints: a branch-and-price algorithm, *International Journal of Production Research*, 47, 1877-1893, (2009).

SPIESKMA F.C.R., On the approximability of an interval scheduling problem. *Journal of Scheduling*, 2, 215-227, (1999).

WOLFE W.J., Sorensen S.E., Three Scheduling Algorithms Applied to the Earth Observing Systems Domain, *Management Science*, 46, 148-168, (2000).

EK 1. KARAR DESTEK SİSTEMİNDEN ÖRNEK VERİ DOSYALARI

Aşağıdaki örnek problem girdi verisi 20 işlik bir BTSİÇ problemi için, sırasıyla her satırda iş numaraları, geliş zamanı, son bitiş zamanı ve getirilerine karşılık gelmektedir. Satır sonundaki değerler alınması mümkün olan makinelere ait maliyetlerdir. Bu örnekte tüm makine maliyetleri özdeş ve 60'a eşittir.

```
20
1       73      78      19      60
2       199     204     13      60
3       27      31      4       60
4       0       9       19      60
5       152     162     5       60
6       77      83      18      60
7       159     168     15      60
8       83      93      17      60
9       19      25      16      60
10      164     169     17      60
11      151     155     19      60
12      17      24      17      60
13      133     141     19      60
14      188     198     20      60
15      83      90      4       60
16      9       17      20      60
17      36      43      4       60
18      44      51      11      60
19      133     140     20      60
20      163     172     8       60
```

Aşağıdaki örnek problem çıktı verisi yukarıdaki BTSİÇ probleminin optimal çözümüne ait değerleri vermektedir. İlk iki satırda sırasıyla optimal amaç fonksiyon değeri ve çözüm süresi (sn.) verilmekte, bir sonraki satırdaysa maksimum kullanılabilecek 3 makineden 2 tanesinin kullanıldığı belirtilmektedir. Sonraki iki satırda çözümdeki iki makineye ait bilgiler yer almaktadır. Her makine satırında sırasıyla kullanılan makine maliyeti, toplam getiri, işlenen iş sayısı, iş yükü, kullanım oranı ve işlenen işlerin endeksleri verilmektedir. Dosyanın son satırı, problemde verilen 20 işin 19'unun yapıldığı (yüzdesiyle beraber - %95) ve tüm işlerin getirisi olan 285'in 277'lik kısmının (%97.19) yapıldığını özetlemektedir.

```
157
0.1092
3       2
60      184     13      94      47.00   1       2       3       4       5       8       9       10
        13      14      16      17      18
60      93      6       40      20.00   6       7       11      12      15      19
19      95.00   277     285     97.19
```

Aşağıdaki XML dosyası biçemindeki örnek problem girdi verisi Şekil 7'de gösterilmiş BTSİÇ problemine aittir.

```
<?xml version="1.0" encoding="utf-8"?>
<problem>
  <instance name="Tekstil Atölyesi - Bütünleşik Planlama Örneği" type="BTSIC" m="2" n="5">
    <no>25</no>
    <interest>0</interest>
    <targetProfit>0</targetProfit>
  </instance>
  <jobs>
    <job no="1" ready="43" due="61" weight="24">Kumaş boyalı baskı</job>
    <job no="2" ready="90" due="110" weight="12">Taşlama 1.seviye</job>
```

```
    <job no="3" ready="70" due="80" weight="8">Yan dikişler</job>
    <job no="4" ready="10" due="20" weight="5">Kesim seviye 1</job>
    <job no="5" ready="125" due="150" weight="15">Pres kalıplama</job>
  </jobs>
  <machines>
    <machine no="1" cost="20">Boyama makinesi</machine>
    <machine no="2" cost="20">Kesim-Dikiş makinesi</machine>
  </machines>
</problem>
```

TÜBİTAK
PROJE ÖZET BİLGİ FORMU

Proje Yürütücüsü:	Yrd. Doç. Dr. DENİZ TÜRSEL ELİİYİ
Proje No:	109M576
Proje Başlığı:	Aralık Çizelgelemede Kapasite Planlama Ve Kapasite Artırımı Problemleri
Proje Türü:	Kariyer
Proje Süresi:	24
Araştırmacılar:	
Danışmanlar:	
Projenin Yürütüldüğü Kuruluş ve Adresi:	İZMİR EKONOMİ Ü. BİLGİSAYAR BİLİMLERİ F. ENDÜSTRİ SİSTEMLERİ MÜHENDİSLİĞİ B.
Projenin Başlangıç ve Bitiş Tarihleri:	01/04/2010 - 01/04/2012
Onaylanan Bütçe:	39560.0
Harcanan Bütçe:	30560.0
Öz:	Bu projede yöneylem araştırmasının çizelgeleme alanında önemli konulardan olan aralık çizelgeleme ele alınmıştır. Rezervasyon sistemleri dahil olmak üzere birçok uygulama alanında kullanılan pratik değeri yüksek bu problem kapsamında taktik ve operasyonel kararların birleştirilmesi hedeflenmiştir. Rezervasyon sistemlerinde kullanılacak makine veya operatör sayısının ve maliyetinin belirlenmesi projede ele alınan taktik kararları, seçilen operatörler ile hangi iş veya rezervasyon kümesinin işleneceğinin, dolayısıyla karın belirlenmesi ise operasyonel kararları oluşturmaktadır. Literatürde daha önce ayrı ayrı ele alınan ve hiyerarşik sırada incelenen bu problemlerin aynı anda çözümünü sağlayan matematiksel modeller ve çözüm yaklaşımları projenin ana kapsamını oluşturmaktadır. Özellikle mevsimsellik gösteren rezervasyon sistemlerinde yararlı olacak bu yeni problemlerin literatürde aralık çizelgelemede kapasite planlama ve kapasite artırımı alanında yer alan önemli bir boşluğu dolduracağı ve bilimsel açıdan önemli katkı sağlayacağı düşünülmektedir. Bu kapsamda öncelikle temel sabit iş çizelgeleme problemi ele alınmış, bu problem özelinde operasyonel ve taktik kararları birleştiren yeni bir bütünleşik problem tanımlanarak bu problem için bir matematiksel model geliştirilmiştir. Bu problem için polinom zamanlı bir kesin çözüm yaklaşımı geliştirilmiş ve sayısal deneylerle performans analizi yapılmıştır. Projenin daha sonraki kısımlarında daha zor problemler olan ve pratik önemi yüksek kar kısıtlı sabit iş çizelgeleme ve çalışma zamanı belirleme problemleri bütünleşik şekilde tanımlanarak analiz edilmiştir. Bu problemler için problemlerin yapısal özelliklerden yararlanan etkin ve verimli çözüm yöntemleri geliştirilmiş ve sayısal deneylerle performans ölçümü gerçekleştirilmiştir. Son olarak NP-zor bir problem olan bütünleşik değişken iş çizelgeleme ele alınmış ve problem için çok etkin bir sezgisel çözüm algoritması geliştirilmiştir. Projede geliştirilen model ve yöntemlerin ülkemizin teorik birikimine ve uygulamaya katkı sağlaması hedeflenmiştir.
Anahtar Kelimeler:	Aralık çizelgeleme, kapasite planlama ve kapasite artırımı, optimizasyon, sezgisel yöntemler.
Fikri Ürün Bildirim Formu Sunuldu Mu?:	Hayır