Kısıtlar İçerecek Şekilde Revize Edilmiş Atama Algoritmasına Ait Bir Uygulama

Mustafa Can Kurnaz1, Recai Oktaş2

1 Ondokuz Mayıs Üniversitesi, Akıllı Sistemler Mühendisliği Bölümü, Samsun

2 Ondokuz Mayıs Üniversitesi, Bilgisayar Mühendisliği Bölümü, Samsun

[can.kurnaz@bil.omu.edu.tr](mailto:can.kurnaz@bil.omu.edu.tr), [roktas@bil.omu.edu.tr](mailto:roktas@bil.omu.edu.tr)

Özet: Bu çalışmada, istenilen kısıtlar altında uygun grupların ve bununla birlikte ilgili eşleştirmelerin oluşturulmasına olanak sağlayan “Kısıtlar İçerecek Şekilde Revize Edilmiş Atama Algoritması” önerilmiş ve bilgisayar ortamında (Python programlama dilinde) uygulaması gerçeklenmiştir. Uygulama sonuçlarına göre, standart atama algoritmasının uygun çözüm vermediği durumlarda verimli bir şekilde (makul sürede sonuç veren) uygun çözüme ulaşılmaktadır. Algoritmanın çeşitli eşleştirme problemlerine verimli bir şekilde çözüm verecek hale getirilmesinin seçim aşamasında belirtilen kısıtların ağırlıklandırılması değiştirilerek elde edilebileceği gösterilmiştir. Bu uygulamanın kullanılabilirliğini göstermek amacıyla, Ondokuz Mayıs Üniversitesi Bilgisayar Mühendisliği Bölümü bitirme projeleri seçiminde/dağıtımında kullanılacak şekilde özelleştirilme yapılmış ve proje gruplarının oluşturulması konusunda çeşitli değerlendirmeler yapılmıştır.

Anahtar Sözcükler: Eşleştirme Problemleri, Bitirme Projesi Dağıtımı, Revize Edilmiş Atama Algoritması, Kısıtlar İçeren Atama Algoritması

An Application of Revised Assignment Algorithm Involving Constraints

Abstract: In this study, a revised assignment algorithm is proposed for making suitable groups under the desired constraints. Based on the results, the application can reach an appropriate solution within reasonable timing, where a standard assignment algorithm is not able to find a solution for a problem. The algorithm is implemented with Python programming language in computer environment.

While building the algorithm, finding an effective solution for various assignment problems is obtained by changing the weighting of constraints given during the selection. For showing the availability of this application; a customization have been applied for using in selection and allocation of the Ondokuz Mayıs University Computer Engineering Department graduation projects, and some assessments have been made for choosing the graduation project groups.

Keywords: Matching Problems, Graduation Project Distribution, Revised Assignment Algorithm, Assignment Algorithm Involving Constraints

1. Giriş

Eşleştirme problemi, eşleştirilecek öğelerin var olan maliyetlerinin minimuma indirgenmesi ile birlikte öğeler arası uyumluluğun yine minimum maliyetler ile kontrol edilmesi, karşılaştırılması ve karar verilmesi problemidir. Eşleştirme, var olan elemanların özellikleri ile tanımlanmış hedefler arasında kurulan en doğru yoldur. Doğru yola erişilmesi ise doğru matrislerin tanımlanması ve elemanların doğru yerleştirilmesi ile mümkündür. Öğeler arasında tanımlanan matrislerin kurgulanması çeşitli parametrelerin varlığı ile doğrudan ve dolaylı olarak ilişkilidir.

Bu çalışmada var olan elemanların, özelliklerine bağlı olarak en uygun şekilde gruplandırılması ve oluşturulan grupların, yine bu elemanların özelliklerine bağlı olarak hedefler ile arasındaki en uygun bağlantının sağlanması amaçlanmıştır. Elemanların gruplanması ve hedefler ile arasında bağlantı kurulma işleminin aynı adımda yapılması en uygun eşleşmenin oluşturulması için önemli bir adımdır. Bu çalışmanın ikinci bölümünde, amaçlananları gerçekleştirebilir bir modifiye edilmiş eşleştirme algoritması tasarımı üzerinde durulmuş, uygulama detaylandırılmıştır. Çalışmanın üçüncü bölümünde, gerçek veriler üzerinden hazırlanan eşleştirme algoritması kullanılarak sonuçlar elde edilmiş ve sonuçların doğruluğu tartışılmıştır. Son olarak dördüncü bölümde uygulamanın sonuçlarına yer verilmiştir.

2. Algoritma

Algoritma temel olarak, eşleşmeyi bekleyen tüm elemanların eşleşecekleri hedef için kombinasyonlarının oluşturulması, bu kombinasyonların aynı hedef için puanlanması ve en yüksek puanı alan kombinasyonun bu hedef ile eşleştirilmesi şeklinde tasarlanmıştır. Her eşleştirmeden önce yapılan bir kontrol ile dağıtım işlemi tamamlandığında geriye hiç bir hedef ile eşleştirilmemiş eleman/elemanlar kalması durumu engellenmektedir.

Eşleştirme sonuçları, belirlenmiş kriterlerle kullanıcı tarafından verilecek ağırlıklar ile yönlendirilebilmektedir. Böylece sonuçtan memnun kalınmadığı durumlarda başka sonuçlar üretilebilmektedir.

Bu işlem sürecinde en büyük sorun kombinasyonların oluşturulma süresidir. n adet eleman bulunan bir veri kümesinde, bir hedefin m elemanla eşleştirileceğini düşünürsek bu n! *⁄* (n-m)! × n! adet kombinasyon oluşturulması ve bu kombinasyonların puanlanması anlamında gelir. Bu durumun önüne geçmek için elemanların tercih bilgilerine kullanılarak, öncelikle bu hedefe öncelik vermiş elemanların kombinasyonları oluşturulur. Kombinasyon oluşturmaya yetecek kadar eleman yoksa hedef işaretlenip işlemden çıkarılır. Herhangi bir atama yapıldığında ise tüm hedeflerin işaretleri temizlenir. Akış bu şekilde ilerledikten sonra işaretli olmayan hedef kalmadığında tüm elemanların tüm hedefler için kombinasyonları oluşturur.

Algoritmanın en önemli özelliklerinden birisi de eşleştirme sonuçlarının ayarlanabilir olmasında yardımcı olan kısıtlardır. Algoritmanın şu andaki versiyonunda kısıtlar, elemanların birbirleri ile olan uyumluluğu (grup uyumluğu), elemanların hedefler ile olan uyumluluğu (grup-hedef uyumluluğu) ve elemanların eşleşebilme öncelikleridir. Bu kısıtlar, her birine verilen ağırlıklar ile çarpılıp ortalaması alınarak kombinasyonların puanları üretilmektedir.

**def score(groups, objective, datas, relation\_matrix):**

**scores = list()**

***# kriterlerin ağırlıkları***

**weights = [0.80, 0.1, 0.1]**

***# kriterleri ve ağırlıkları kullanarak puanı hesapla***

**for group in groups:**

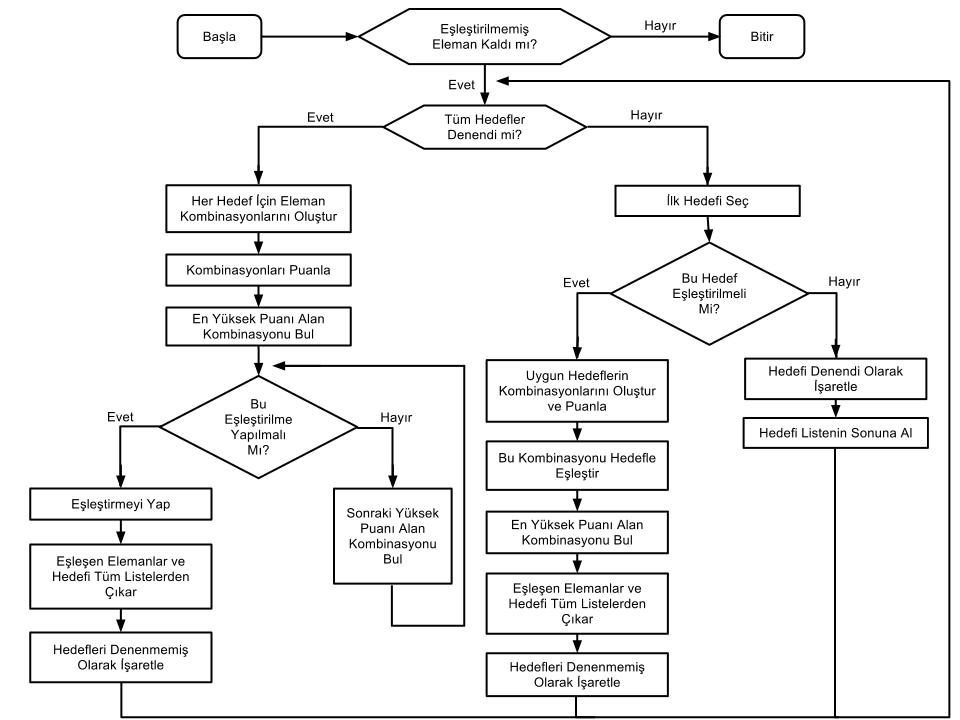
**score = priority(group, datas) \* weights[0]**

**score = score + group\_compatibility(group, relation\_matrix) \* weights[1]**

**score = score + group\_objective\_compatibility(group, objective, datas)**

**scores.append(score)**

**return scores**



**Şekil 1**. Kısıtlar İçerecek Şekilde Revize Edilmiş Atama Algoritması Akış Şeması

2.1 Algoritma Parametreleri

Algoritma her elemanın öncelik sırasına göre uyumlu olduğu üyeler, öncelik sırasına göre uyumsuz olduğu üyeler, öncelik sırasına eşleştirilmesine uygun olan hedefler; her hedefin kaç adet üye ile eşleşebileceği (her hedef için farklı rakamlar kullanılabilir) bilgilerini kullanmaktadır.

2.2 Grup Uyumluluğu

Veri kümesindeki eleman sayısı n olarak düşünülürse, algoritma işleyişinde n×n’lik bir matris barındırmaktadır. Bu matris ilişki matrisidir. İlişki matrisindeki i. satır j. sütun, i numaralı elemanın j numaralı eleman ile olan uyumluluğunu göstermektedir. Tersi durum da geçerlidir. Uyumluluk durumu 1, -1 ve 0 durumları ile ifade edilmektedir. 1 durumu uyumlu, -1 durumu uyumsuzluğu, 0 durumu ise nötr ilişkiyi ifade etmektedir.

Grup elemanların birbiri ile olan uyumluluğu ilişki matrisi kullanılarak hesaplanmaktadır. Bir hedef m eleman ile eşleşeceği durumda, oluşturulan tüm kombinasyonlardaki m adet elemanın ilişki matrisindeki karşılığına bakılmaktadır. m elemanlık gruptaki tüm ikili ilişkilerin puanı toplanmaktadır. m×m adet uyumluluğun yani 1 durumunun en uyumlu grup olduğu ve bu kriterden 100 puan alacağı kabul edilerek, basit bir orantı ile grup uyumluluğu puanı hesaplanır.

M ilişki matrisi, Z grup üyelerinin sıra numaraları kümesi olmak üzere; 100 puan üzerinden grup uyumluluğu puanı hesabı aşağıdaki denkleme göre yapılmaktadır.

*Grup Uyumluluğu Puanı =*

*(Σ (Mij) | i,j ∈ Z)* × *100  ⁄ n2*

**def group\_compatibility(group, relation\_matrix):**

**score = 0**

**n = len(group)**

***# gruptaki tüm ikililerin uyumuluğunu kontrol et***

**for i in group:**

**for j in group:**

***# 1 (uyumlu) ise 1 ekle***

**if relation\_matrix[i][j]==1:**

**score = score + 1**

***# -1 (uyumsuz) ise 1 çıkar***

**elif relation\_matrix[i][j]==-1:**

**score = score - 1**

***# 100 üzerinden puanını döndür***

***# n\*n adet uyumluluk 100 puana denk gelir***

**return ((score \* 100) / n) / n**

**2.3 Grup-Hedef Uyumluluğu**

Bir gruptaki elemanların bir hedefle olan uyumluluğunu ifade etmektedir. Her elemanın hedeflerin önceliğini belirten bir listesi olduğu kabul edilmektedir. Bu listenin sıralaması eleman-hedef uyumluluğu en çok olan çiftten en az olan çifte doğru oluşturulmuştur.

Bir grubun grup-hedef uyumluluğu, her elemanın o hedef ile uyumluğu puanının ortalaması alınarak hesaplanmaktadır. Bu uyumluluk puanı ise; hedefin elemanın öncelik listesindeki yeri ve eşleşebilme önceliği puanının yardımıyla hesaplanır.

Öncelik puanı 100 olan ve olası bağlanacağı hedefe ilk sırada öncelik verilmiş bir eleman-hedef çiftine 100 puan verileceği kabul edilerek, basit bir orantı ile eleman-hedef uyumluluk puanı hesaplanmaktadır. Eleman sayısına göre ortalama alınarak da grup-hedef uyumluluğu hesaplanmaktadır.

m hedef sayısı, n*i* i numaralı elemanın öncelik listesinde hedefin sırası, S*i* i numaralı elemanın eşleşebilme önceliği puanı, Z grup üyelerinin sıra numaraları kümesi, k gruptaki eleman sayısı olmak üzere, 100 puan üzerinden grup-hedef uyumluluğu puanı hesabı aşağıdaki denkleme göre yapılmaktadır.

*Grup-Hedef Uyumluluğu Puanı =*

*(Σ ((m - ni)* × *Si) |  i ∈ Z)  ⁄ m* × *k*

**def group\_objective\_compatibility(group, objective, datas):**

**score = 0.00**

***# tercih listesindeki proje sayısı***

**m = len(datas[0][4])**

**for id in group:**

***# proje listede kaçıncı sırada tercih edilmiş (tersten)***

**n = m - datas[id][4].index(objective[0])**

***# en yüksek not ortalaması olan öğrencinin istediği projeye öncelik verilmeli***

**score = score + (n \* datas[id][2])**

***# grubun puanlarının ortalamasını döndür***

**return score / (m \* objective[2])**

**2.4 Eşleşebilme Önceliği**

Her elemanın kısıtlara bağlı olmayan kendine ait bir puanı bulunduğu kabul edilmektedir. Bu puan 100 üzerinden verilmiştir. Eşleşebilme önceliği puanına da karşılık gelmektedir.

S*i* i numaralı elemanın eşleşebilme önceliği puanı, Z grup üyelerinin sıra numaralar kümesi, k gruptaki eleman sayısı olmak üzere, 100 puan üzerinden eşleşebilme önceliği puanı hesabı aşağıdaki denkleme göre yapılmaktadır.

*Eşleşebilme Önceliği Puanı =*

*(Σ (Si) |  i ∈ Z)   ⁄  k*

**def priority(group, datas):**

**score = 0**

**for id in group:**

**score = score + datas[id][2]**

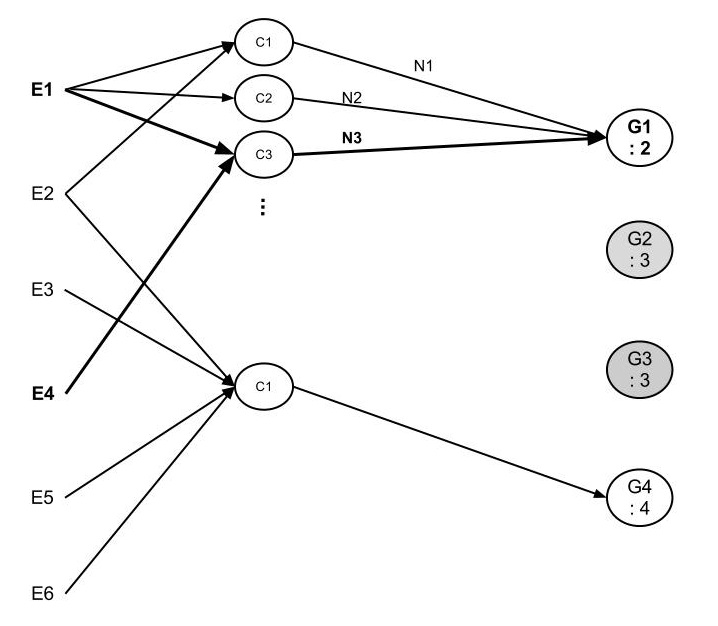
**score = score / len(group)**

**return score**

3. Uygulama Sonuçları

3.1 Basit Bir Eşleştirme Örneği

Şekil 2’de gösterilen gibi 6 eleman ile 4 hedefin eşleştirilmesi ele alınmıştır. Hedeflerden P1, 2; P2, 3; P3, 3 ve P4 4 eleman ile eşleştirilmeye uygundur.

****

**Şekil 2**. Öğrenci - Proje eşleştirme işleminde bir adım

İlk adımda P1 hedefi seçildi ve bu hedef eşleştirilirse geriye kalacak elemanların kalacak hedeflerle uygun şekilde eşleştirilip eşleştirilemeyeceği kontrol edildi. Bu eşleştirme yapılırsa kalacak 4 eleman geri kalan hedeflerle eşleştirilebilecektir. Kombinasyonlar oluşturuldu ve en yüksek uyumluluğu yakalayan S1 ve S4 kombinasyonu, P1 hedefi ile eşleştirilmiştir.

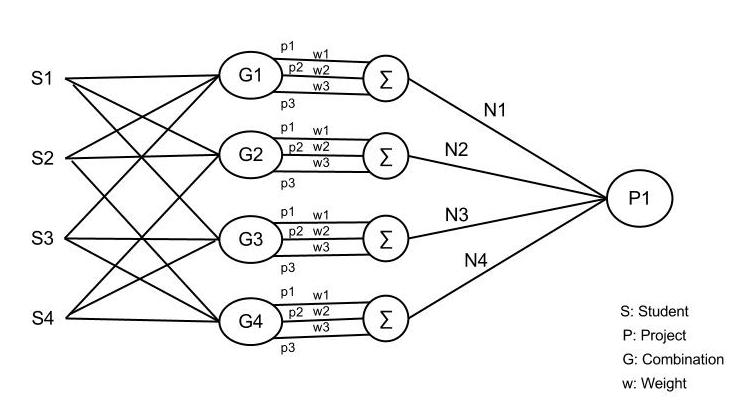
İkinci adımda P2 hedefi seçildi ve bu hedef eşleştirilirse 1 elemanın hiçbir hedef ile eşleştirilemeyeceği görüldü. Bu durumda denenecek hedefler arasından üç elemanlık hedefler (P2, P3) çıkartılır.

Üçüncü adımda P3 hedefi seçildi ve eşleştirildi mi kontrolü yapılarak eşleştirilmek için uygun olduğu görüldü. Kalan dört eleman P4 hedefi ile eşleştirildi. Atama işlemi sonlandı.

3.2 Bitirme Projesi Dağıtımı

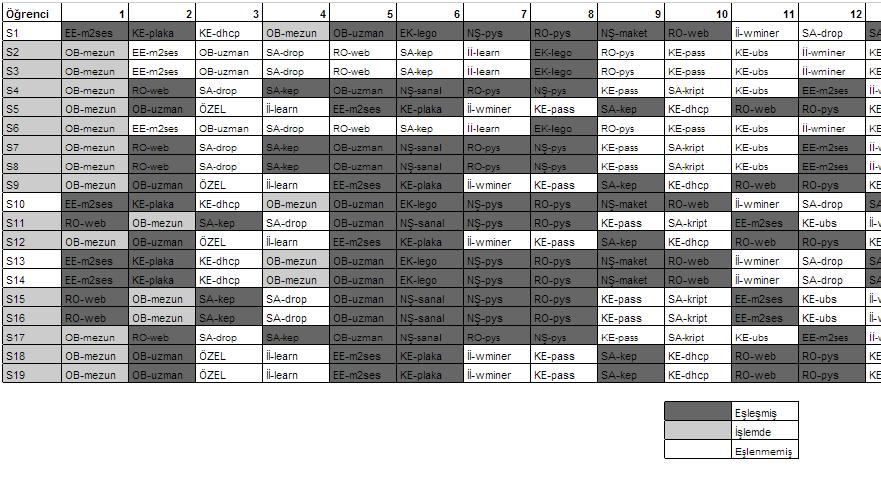
Bir proje dağıtma işlemi için uygulandığında algoritma, her proje için tüm projeler denenmiş ise herhangi bir projeye atanmamış tüm öğrencilerden; denenmemiş bir proje bile varsa bu projeyi ilk kalan listesinde ilk sırada tercih etmiş öğrencilerden bu proje için grup kombinasyonlarının oluşturulması, bu kombinasyonların tüm kısıtlar için ayrı ayrı puanlanması, her kısıt için elde edilen puanın o kısıtın ağırlığı ile çarpılması, elde edilen puanların toplanması ve en yüksek puanı alan kombinasyonun projeye atanması şeklinde ilerlemektedir.

Puanların hesaplanmasında, öğrencilerin not ortalamaları *eşleşebilme önceliği puanı*, öğrencilerin grup üyesi tercihleri *grup uyumluluğu puanı*, öğrencilerin çalışmak istedikleri proje tercihleri ise *grup-hedef uyumluluğu puanı* hesabı için kullanılmıştır.

****

**Şekil 3.** Öğrenci - Proje eşleştirme işleminde bir adım

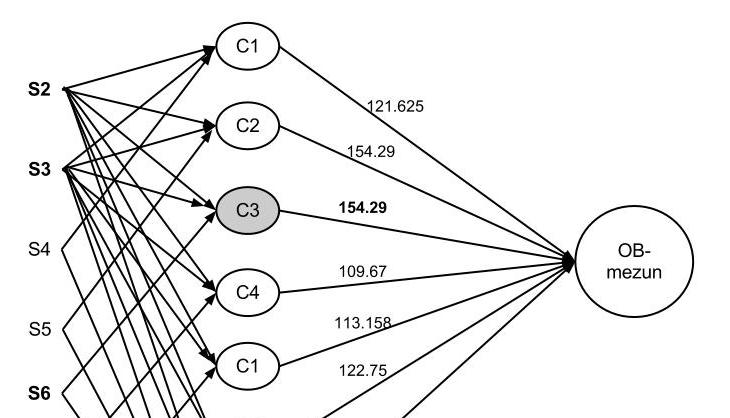
Bu çalışma kapsamında algoritmanın başarısı test edilmek istenmiştir. Bu amaçla, Ondokuz Mayıs Üniversitesi Bilgisayar Mühendisliği Bölümü 4. sınıf öğrencilerinin bitirme projeleri tercih bilgileri alınmıştır. Alınan bilgiler, önceden oluşturulmuş gruplar ile proje eşleştirmesi yapılması amacına uygun düzenlenmiş olduğundan, aynı grupta olan tüm öğrencilerin birbirlerinin grup üyesi tercihlerine eklenerek algoritma için kullanılabilecek şekilde özelleştirilmiştir.



**Tablo 1.**OMÜ Bilgisayar Mühendisliği Bitirme Projesi Dağıtımı işleminde 14. adım

Tablo 1’de görülen örnek bir adımı incelemek gerekirse; 19 öğrenci henüz hiçbir projeye kayıtlanmamıştır. Anlık seçili olan proje OB-mezun projesidir ve bu projenin 3 öğrenci ile eşleştirileceği öngörülmüştür.

Algoritma ilk olarak 19 kişiden 3 kişi bir projeye atanır ise geriye kalan öğrenciler geriye kalan projelere yerleştirildiğinde açıkta öğrenci kalıp kalmayacağını kontrol etti; 16 öğrenci 4’er kişilik 6 projeden 4’üne kolaylıkla dağıtılabilir olduğunu gördü.



**Şekil 4.**OMÜ Bilgisayar Mühendisliği Bitirme Projesi Dağıtımı 14. Adım Kombinasyon ve Puanlama İşlemi

Şekil 4’te gösterildiği gibi seçili olan projesi kalan uyumluluk listesinde ilk sırada olan öğrenciler seçildi; S2, S3, S4, S5, S6, S7, S8, S9, S11, S12, S15, S16, S17, S18 VE S19. Bu öğrencilerin 3’erli kombinasyonları oluşturuldu ve kombinasyonlar puanlandı. S2, S3 ve S6 öğrencilerinden oluşan kombinasyon 154.29 puan alarak en büyük uyumu yakaladı, OB-mezun projesi bu öğrenciler ile eşleştirildi.

Bu bilgiler kullanılarak ve kısıtlara çeşitli ağrılıklar verilerek, atama deneyleri yapılmıştır. Ondokuz Mayıs Üniversitesi Bilgisayar Mühendisliği Bölümü orijinal bitirme projesi dağıtım sonuçlarının belirlenmesinde en önemli kısıtın öğrenci grup tercihleri olduğu kanısına varılmıştır. Bu bilgiyi kullanarak algoritmanın *grup uyumluluğu* kısıtına yaklaşık olarak %60 ağırlığı verildiğinde alınan dağıtım sonuçları, Ondokuz Mayıs Üniversitesi Bilgisayar Mühendisliği Bölümü orijinal bitirme projesi dağıtım sonuçları ile %100 örtüşmektedir.

**4. Sonuç**

Bu çalışmada, bilinen atama algoritması istenilen kısıtlar altında uygun grupların ve bununla birlikte ilgili eşleştirmelerin oluşturulmasına olanak sağlayacak şekilde belli kısıtlara dayandırılarak revize edilmiş ve bu algoritmanın uygulaması gerçeklenmiştir. Bu uygulama kullanılarak, Ondokuz Mayıs Üniversitesi Bilgisayar Mühendisliği Bölümü bitirme projeleri seçimi ve dağıtımı gerçekleştirilmiştir. Sonuçlar, insan emeği ile uzun uğraşlar sonucunda belirlenen dağıtımlar ile %100 örtüşmektedir.

Burada dikkate alınan algoritma; lisans bitirme projelerinin en uygun şekilde, en az insan gücü kullanılarak ve en verimli şekilde dağıtılmasında kullanılabilir.

**5. Kaynaklar**

[1] Burkard, Rainer E., Mauro Dell'Amico, and Silvano Martello. Assignment Problems, Revised Reprint. Siam, 2009.

[2] Munkres, James. "Algorithms for the assignment and transportation problems." Journal of the Society for Industrial & Applied Mathematics 5.1 (1957): 32-38.

[3] Hanan, Maurice, and Jerome M. Kurtzberg. "A review of the placement and quadratic assignment problems." Siam Review 14.2 (1972): 324-342.

[4] Kuhn, Harold W. "The Hungarian method for the assignment problem." Naval research logistics quarterly 2.1‐2 (1955): 83-97.

[5] Kuhn, Harold W. "Variants of the Hungarian method for assignment problems." Naval Research Logistics Quarterly 3.4 (1956): 253-258.

[6] Brualdi, Richard A. (2006). Combinatorial matrix classes. Encyclopedia of Mathematics and Its Applications 108. Cambridge: Cambridge University Press. ISBN 0-521-86565-4.Zbl 1106.05001.

[7] Sharma, P. R., and S. Shukla. "An Advance Technique for Solving Assignment Problems."

[8] Silver, Roland. "An algorithm for the assignment problem." Communications of the ACM 3.11 (1960): 605-606.

[9] Gale, David, and Lloyd S. Shapley. "College admissions and the stability of marriage." American mathematical monthly (1962): 9-15.

[10] McVitie, David G., and Leslie B. Wilson. "The stable marriage problem." Communications of the ACM 14.7 (1971): 486-490.

[11] Gusfield, Dan, and Robert W. Irving. The stable marriage problem: structure and algorithms. Vol. 54. Cambridge: MIT press, 1989.

[12] Irving, Robert W. "An efficient algorithm for the “stable roommates” problem." Journal of Algorithms 6.4 (1985): 577-595.

[13]Iwama, Kazuo, and Shuichi Miyazaki. "A survey of the stable marriage problem and its variants." Informatics Education and Research for Knowledge-Circulating Society, 2008. ICKS 2008. International Conference on. IEEE, 2008.

[14] Crawford, Vincent P., and Elsie Marie Knoer. "Job matching with heterogeneous firms and workers." Econometrica: Journal of the Econometric Society (1981): 437-450.