ANKARA ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

STAJ RAPORU

Araçlar İçin Buluşsal (Heuristik) Rota Hesaplama

Betül Öğmen
19290263

20 Ağustos 2022

ÖZET

Bu projede bir şirketin sahip olduğu araçların hangi rotada ve ne şekilde gideceğini hesaplayabilecek bir program yazıldı .

Gidilmesi gereken birçok konum, varılması gereken saatler ve yük kapasitesi gibi birçok problem bir arada olduğunda, en uygun rotayı belirlemek el hesabıyla bazen mümkün olamıyor. Rota hesaplama ile ilgili heuristik çözümler üretebilen fonksiyonları içeren OR-TOOLS kütüphanesi yardımıyla en verimli veya verimliye yakın çözümler üretilmeye çalışıldı. Görselleştirme ve matematiksel işlemler için matplotlib, numpy ve ortools kütüphaneleri kullanılmıştır.

Programda araçların varması gereken zaman aralıkları, yüklenip indirilmesi gereken her yük başına o yükte harcanacak olan süre gibi birçok kısıt bulunuyor. Bu kısıtlar, gerçekleşmesi mümkün ise eğer sorun olmadan uygulanıyor ancak olması imkansız kısıtlarda rota oluşmuyor. Örneğin araçların hızının, mesafe ve zamana göre çok yetersiz olması veya yük başına tanınan sürenin, bir sonraki lokasyonda bulunulması gereken saat aralıklarında varılması için fazla uzun olması gibi. Kısıtları doğru bir şekilde değiştirdiğimizde rota outputlarına ulaşırız.

KURUM BİLGİLERİ VE TANITIMI

Staj yapılan kurumun;

Adı : DS BİYOİNFORMATİK ROBOTİK YAZILIM ELEKTRONİK ve MEKANİK ANONİM ŞİRKETİ

Çalışma Yapılan Birim : Yazılım

Adresi : BAHÇELİEVLER MAH. 319 CAD. E BLOK (TEKNOKENT) APT. NO:35 E/B02 GÖLBAŞI - GÖLBAŞI / ANKARA / TÜRKİYE

Telefon : 03124283120

E-posta : info@dsbionano.com

İnternet Sayfası (varsa) : https://www.dsbionano.com

BİLGİ TEKNOLOJİLERİ Meslek grubunda, GÖLBAŞI / ANKARA Bölgesinde Bilgisayar programlama faaliyetleri (sistem, veri tabanı, network, web sayfası vb. yazılımları ile müşteriye özel yazılımların kodlanması vb) konularında hizmet vermektedir.

1. GİRİŞ

Raporun Bölümleri

- **Or-Tools Kütüphanesi ile araçlar için optimum/heuristik çözümü hesaplama 2**
- **Kısıtlar (Constraints) 2**
- **Kodda Kullanılan Fonksiyonların Anlamları 3**
    1. create_data_model()
    2. RoutingIndexManager()
    3. RoutingModel()
    4. distance_evaluator_index
    5. RegisterTransitCallback
    6. create_distance_evaluator()
    7. SetArcCostEvaluatorOfAllVehicles
    8. demand_evaluator_index
    9. RegisterUnaryTransitCallback
    10. create_demand_evaluator()
    11. add_capacity_constraints()
    12. time_evaluator_index
    13. create_time_evaluator()
    14. add_time_window_constraints()
    15. GetDimensionOrDie()
    16. FixedDurationIntervalVar()
    17. SetBreakIntervalsOfVehicle()
    18. DefaultRoutingSearchParameters()
    19. first_solution_strategy
    20. SolveWithParameters()
    21. print_solution()
- **Kod Diyagramı 8**

- **Giriş Çıkış Değerleri** **11**

- **Kaynaklar** **15**

ORTOOLS KÜTÜPHANESİ İLE ARAÇLAR İÇİN OPTİMUM/HEURİSTİK ÇÖZÜMÜ HESAPLAMA

Nakliye şirketleri ve ürünlerini birçok lokasyona tedarik eden şirketler, bir kamyon filosu kullanarak ürünlerini gerekli lokasyonlara ulaştırırlar. Her gün şirket bu ürünleri kamyonlara yüklemeli ve ardından her kamyona özel olacak şekilde ürünleri teslim etmesi için bir rota seçmelidir. Seçilen her rota ve ürün , kamyonların toplam seyahat mesafesini , süresini ve muhtemelen diğer faktörlere bağlı olarak maliyetini etkilemektedir. Buradaki çözülmesi gereken problem en düşük maliyet ve verimli rotaları seçmektir. Bu problem bir eniyileme (optimizasyon) problemidir.

Eniyileme probleminde amaç maliyeti minimize etmektir. Olası bir çözüm için hedefin değerini hesaplayan bir amaç (objective) fonksiyonu tanımlanır. Amaç fonksiyonu, herhangi bir ürün, kapasite veya rota atamasında toplam maliyeti hesaplamaktadır. Optimum çözüm, amaç fonksiyonunun değerinin en iyi olduğu çözümdür. En iyi değer, çözülen probleme bağlı olarak maksimum veya minimum değer olabilir.

Kısıtlar (Constraints)

Kısıtlar bir sorunun özel gereksinimlerine dayalı olarak belirlenen sınırlardır. Bu raporda ise, rota hesaplama sırasında karşımıza çıkacak olan kısıtlar, gerçek hayattaki somut engelleri temsil etmektedir. Hesaplamalar sırasında sınırları doğru belirlemek çok önemlidir çünkü or-tools kütüphanesi fonksiyonları çok verimli, en iyi şekilde (optimum) bir çözüm bulsa dahi eğer şirket ,bulunan çözümdeki gerekliliklere yeterince sahip değil ise çözüm gerçek hayata uygulanamaz. Örneğin nakliye şirketi kamyonlara belirli bir ağırlığın üzerinde ürün yükleyemiyor ise çözüm belirlenirken bu kısıt göz önüne alınmalıdır, gerekli fonksiyonlar ve değişkenler kullanılmalıdır. Herhangi bir kamyona o değerin üzerinde yükün yüklendiği bir çözüm gerçek hayat ile uyuşmadığı için üretilmemelidir. Gerçek hayatta uygulanabilir çözümler için tasarlanan problem, verilen bütün kısıtlamaları karşılamalıdır. Araçların en verimli rota problemini çözmenin ilk adımı kısıtlamaları belirlemektir.

Google or-tools kütüphanesi en iyi (optimum) rotayı belirlemek için kullanılır. Staj sırasında ortools kütüphanesi python dili üzerinde kullanıldı.

- Araç sayısı,
- Araçların kapasiteleri,
- Başlangıç ve bitiş konumları,
- Gidilmesi gereken yerlerde bulunmaları gereken saat aralıkları,
- Lokasyonlarda verilebilecek maksimum ve minimum ara süreleri
- Araçların gidecekleri hızlar,
- Taşıdıkları yük başına tanınan süreler,
- Lokasyonlarda yüklenmesi, indirilmesi talep edilen yük miktarı

gibi kısıtlar çalışılan programa eklendi. Görselleştirme için matplotlib ve numpy kütüphanelerinden de yararlanıldı.

Koddaki Fonksiyonların Anlamları

create_data_model() : Problem için kısıtlanacak verileri oluşturur. ‘data’ sözlüğünde(dictionary):

-‘distance_matrix’ anahtarı (key)’i konumlar arasındaki metre cinsinden mesafeyi,

-‘locations’ anahtarı gidilmesi istenen lokasyonların (x,y) düzleminde karşılıklarını,

-‘numlocations_’ gidilecek lokasyon kısıtını,

-‘time_windows’ lokasyonlarda olunması istenen zaman aralıklarını,

-‘demands’ lokasyonlarda kamyonda yüklü olması gereken yük miktarını,

-‘time_per_demand_unit’ yük başına tanınan süreyi,

-‘num_vehicles’ araç sayısını,

-’vehicle_capacity’ araçların yük kapasitelerini,

-’vehicle_speed’ araçların gidebileceği hızı,

-’starts’ araçların başlangıç konumunu, ‘ends’ bitiş konumunu (konum indislerini)

-’depot’ eğer başlangıç ve bitiş konumları aynı ise iki konumu,

-‘breaks’ araçların maksimum oyalanma süresini ve bunun kesin uygulanıp uygulanmayacağını kontrol eden değişkeni tutar.

Eğer tuple’ın ikinci elemanı False olursa ilk elemanın yani aracın maximum oyalanma süresinin uygulanacağı kesin olur, True olursa bu kısıt uygulanabilir de uygulanmayabilir de.

pywrapcp.RoutingIndexManager(): Yönetici (manager) oluşturmak için ve (Routing Solver) Rota çözücü dahili (lokasyon verileri içindeki) indisleri ve API (Application Programming Interface) aracılığıyla değişken dizinlerini kullanır. Lokasyon sayısı, araç sayısı ve başlangıç bitiş kısıtları ile manager oluşmasını sağlar.

pywrapcp.RoutingModel(): manager değişkenini alır. Rota Modelini oluşturur.(SK. LP. 2013 optimization/reference/python/constraint_solver/pywrapcp Google Or-Tools Reference Routing)

distance_evaluater_index: Uzunluk kısıtını gerçekleştirmek için oluşturulan bir değişkendir.

RegisterTransitCallback: Rota modeli ile kullanılır (pywrapcp.RoutingModel()). Kısıtların kayda geçmesini, uygulanabilmesini sağlar. Kısıtların callback fonksiyonlarını eleman olarak alır. (SK. LP. 2013 optimization/reference/python/constraint_solver/pywrapcp Google Or-Tools Reference Routing)

create_distance_evaluator(): Noktalar arasındaki mesafeyi döndüren callback’i oluşturur, mesafeyi döndürür.

SetArcCostEvaluatorOfAllVehicles: Rota modeli ile kullanılır. (pywrapcp.RoutingModel()) .Tüm araçlar için ark maliyet değerini hesaplar.(SK. LP.

2013 optimization/reference/python/constraint_solver/pywrapcp Google Or-Tools Reference Routing)

demand_evaluator_index: Kapasite kısıtını gerçekleştirmek için oluşturulan bir değişkendir.

RegisterUnaryTransitCallback: Rota modeli ile kullanılır (pywrapcp.RoutingModel()). Ortools kütüphanesinin ortools /constraint_solver /routing.h kısmında bulunur çünkü bir çeşit kısıt çözümü fonksiyonudur. Kısıtların kayda geçmesini, uygulanabilmesini sağlar. Kısıtların callback fonksiyonlarını eleman olarak alır. RegisterTransitCallback fonksiyonundan farkı, oluşturulan rotaların döngü (cycle) oluşturacak şekilde olmasını sağlamaktır.(1)

create_demand_evaluator(): Lokasyonlardaki talep edilen yük miktarını döndüren callback'i oluşturur.

AddDimensionWithVehicleCapacity(): Kapasite kısıtını oluşturmak için yeni boyut (dimension) oluşturur. Boyutların 0. indisleri gerçekleşmesini istediğimiz niteliği , (demand_evaluator_index) , 1. indis kısıt lokasyonlardaki fazladan bekleme süresini , 2. indis nicelik kısıtını (bu boyut için araçların kapasite kısıtı),3. indis çözümün sıfırdan (cumul to zera) çözülüp çözülmemesini, 4. indis ise boyutun ismini tutar.

AddDisjunction(): Ayırma kısıtlaması ekler ,indislerin aktif olduğu maximum kardinalite ile. Bu kardinalite en fazla penalty değeri kadar karışık olabilir.

time_evaluator_index: Zaman kısıtını gerçekleştirmek için oluşturulan bir değişkendir.

create_time_evaluator(): İki konum arasındaki zamanı döndüren callback'i oluşturur.

AddDimension(): Zaman kısıtını eklemek için kullanılan boyut fonksiyonudur. 0. indis zaman gerçekleştirme niteliğini (time_evaluator_index), 1. indis izin verilen bekleme zamanını, 2. indis araç başına düşen maksimum zamanı, 3. indis çözümün 0 dan başlayıp başlamayacağını , 4. indis kısıtın ismini tutar.

GetDimensionOrDie(): Fonksiyon aldığı elemandan büyük bir boyut döndürür, boyut yoksa herhangi bir değer döndürmez ve ölür (1).Kodun 295. ve 299 satırları arasında routing' in kapladığı yer (Size) içinde dönülür. IndexToNode fonksiyonunun yardımıyla indexler node formatına dönüştürülür. Devamında araç sayısı kadar dönen for loop'un içinde aracın, 'breaks' kısıtı vehicle_break değişkeninde tutulur.

FixedDurationIntervalVar(): Başlangıç noktasında araca yük yüklenmesi ya da yüklenmemesi durumlarına göre zaman penceresi (time_windows) eklenir. Eğer yüklenmemiş ise bir zaman penceresi daha açılır. Bu pencereler FixedDurationIntervalVar tarafından oluşturulur. İlk sayı aracın yüklenme zamanını, ikinci sayı aracın boşaltılma zamanını tutar. 3. argüman aracın verebileceği maximum ara süresini, 4. argüman bu sürenin uygulanıp uygulanamayacağını tutar. Eğer 4. argüman True ise bu süre uygulanabilir ya da uygulanmayabilir, False ise bu süre her zaman uygulanır.(1)

SetBreakIntervalsOfVehicle(): Belirli bir araç için molaları ayarlar. 'Interval' ler ile aracı temsil eder.. Boş(slack) node 'ları doğrulayan değerlerin artmasından dolayı node'lar arasındaki transitler kesintiye uğrayabilir. Seyahat sırasında bir aracın çalıştırılmasından önce veya bitiminden sonra mola verdirebilir.

pywrapcp.DefaultRoutingSearchParameters(): Fonksiyon RoutingSearchParameters metadata özellikleri ile varsayılan değer parametrelerini döndürür.(1)

first_solution_strategy : İlk çözümü bulmak için çözücünün kullandığı metottur. Çeşitli parametreleri vardır. PATH_CHEAPEST_ARC parametresi en düşük maliyetli yolu bulur. (1)

SolveWithParameters(): Aldığı parametre ile problemi çözer.(1)

print_solution(): Bulunan çözümü bastırır.

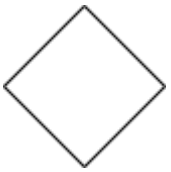
(1)(SK. LP. 2013 optimization/reference/python/constraint_solver/pywrapcp Google Or-Tools Reference Routing)

Kod Diyagramı

Araçlar için rota hesaplama kütüphanesi olan or-tools ile yazılan kodun diyagramda gösterimi:

Diyagramda;

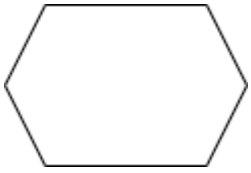
Sembolünün içindeki nesneler şeklin temsil ettiği fonksiyonun nasıl çalıştığını gösterirler.



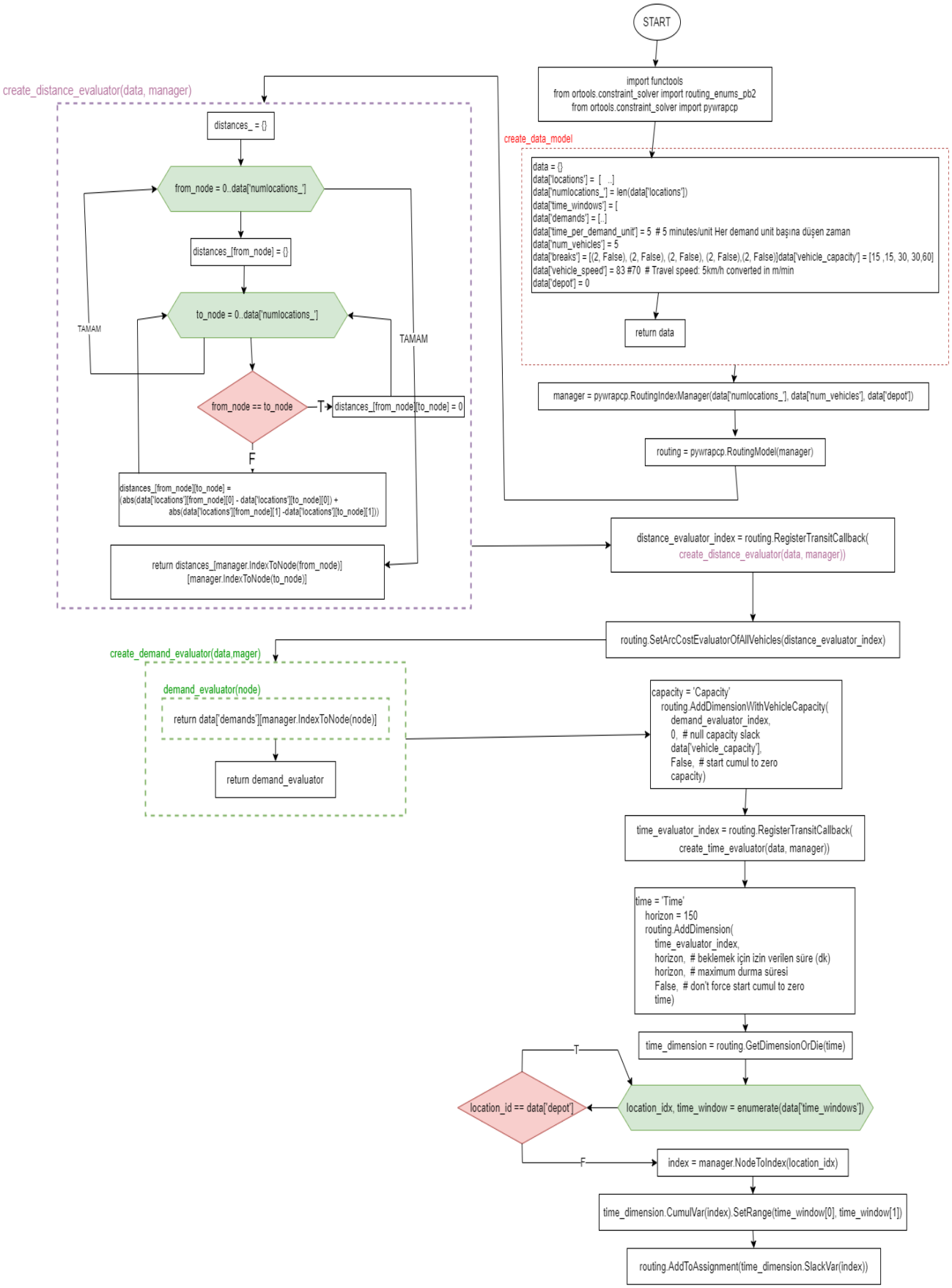
Karar işlem sembolü, koşul yapısını temsil eder. Şeklin içindeki koşul doğru ise kod true yönünde çalışır, doğru değilse false yönünden okunmaya devam eder.

Aritmetik işlem sembolü, kodda uygulanan aritmetik işlemleri ve zaten tanımlı fonksiyonları şemada temsil eder.

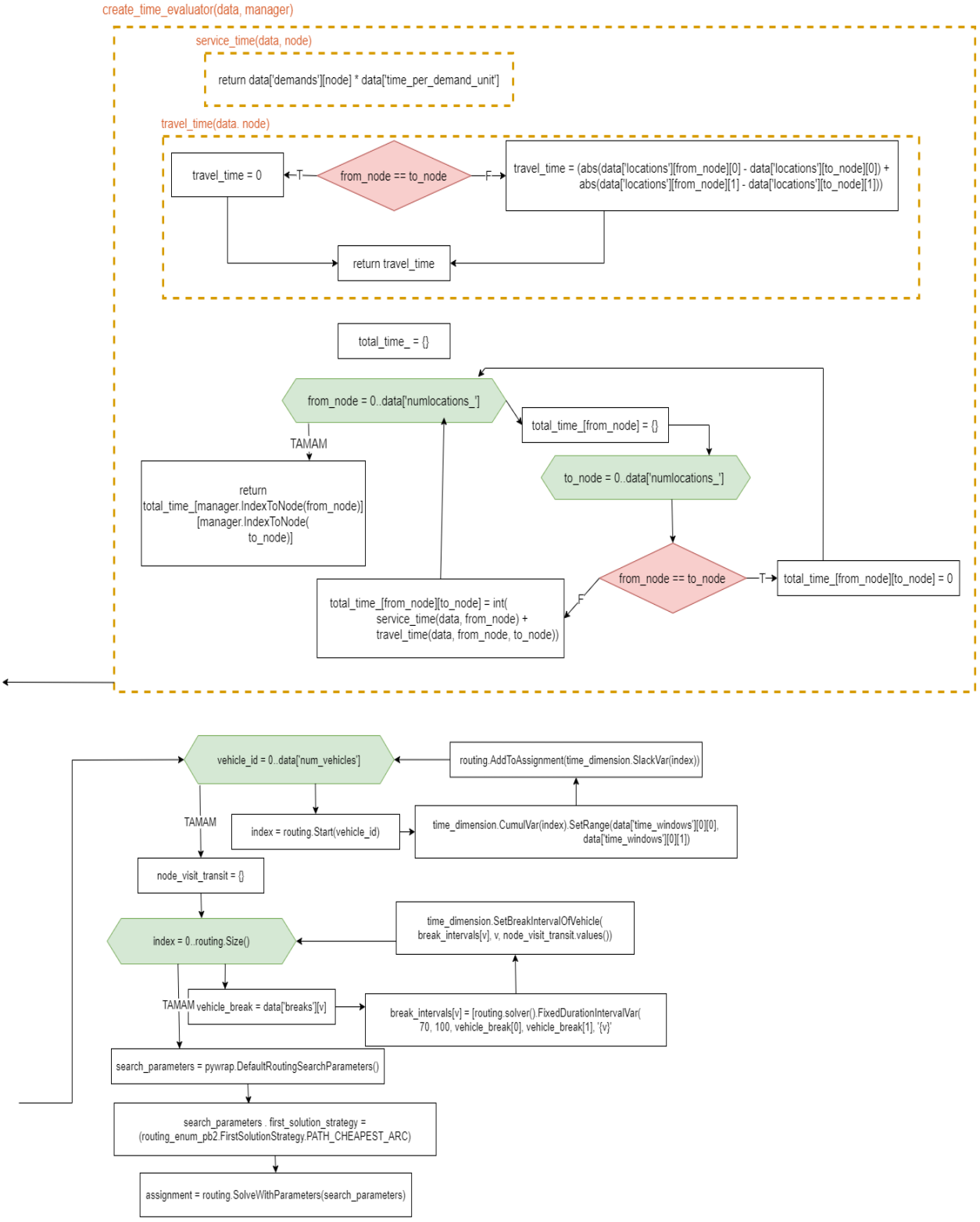
Yazıcı (Printer) sembolü, kodda ekrana basılan yerleri temsil eder. Print() fonksiyonuna yazılan nesneler bu sembol ile diyagramda gösterilir.



Hazırlık, değiştirme veya tekrar sembolü; döngü (loop) yapısının içinde tekrar etmesi gereken değeri diyagramda temsil eder.



Şekil 1.1 Diyagramın sol kısmı



Şekil 1.2 Diyagramın sağ kısmı

Giriş- Çıkış değerleri

Çizelge 1.1. Bazı kısıtlar ve çözümleri

Kısıt/Durum	Durum 1	Durum 2
Yük başına tanınan süre	5 (dk)	4 (dk)
Araç sayısı	5 tane	4 tane
Aracın maksimum oyalanma süresi ve bunun uygulanıp uygulanmayacağı	(13, False), (2, False), (2, True),(2, False),(32,False)	(13, False),(2,True), (2,True),(32,False)
Araçların taşıyabilecekleri yük kapasiteleri	[30, 15, 15, 30, 15]	[30,15,15,30]
Araçların hızları (int tipinde olursa tüm araçlar aynı hızda gider kabul edilir)	83 (m/dk)	77(m/dk)
Araçların başlangıç noktaları	[0, 0, 0, 0, 0]	[1,2,15,16]
Araçların bitiş noktaları	[0, 0, 0, 0, 0]	[0,0,0,0]

Kısıt (constraint) verileri programın içinde data() fonksiyonunun içinde belirtilir. Bazı kısıtlar listeler içinde , bazı kısıtlar tam sayı (integer) tipinde girilebilir. Örneğin yük başına tanınan süre, araç sayısı,araçların hızları kısıtları tam sayı tipinde girilir. Aracın maksimum oyalanma süresi ve bunun uygulanıp uygulanamayacağını tutan değişken, araçların taşıyabilecekleri yük kapasitelerini tutan değişken ve aracın başlangıç ve bitiş noktalarını tutan değişkenler liste tipinde tutulur. Listelerdeki elemanların indisleri, temsil ettikleri değeri kullanacak olan araçların da aynı zamanda indisidir. Mesela araçların başlangıç noktalarını tutan listedeki 2. indisteki

eleman 2 indisli aracın başlangıç noktasını temsil ederken 0. indisteki eleman 0 indisli aracın başlangıç noktasını temsil eder.

Çıkışlarda (outputlarda) ‘Route for vehicle …’ yazan kısım, indexi … olan aracın hesaplanan heuristik rotasını basar. Örneğin 0 indexli aracın rotası 0 noktasından 5 noktasına, 5 noktasından 9 noktasına, 9 noktasından 11 noktasına, 11 noktasından 14 noktasına, 14 noktasından 15 noktasına, 15 noktasından da tekrar 0 noktasına doğru gider.

‘Distance of the route’ kısmında rotanın toplam mesafesini gösterir. Örneğin 0 indexli araç rotasının toplam mesafesi 716 metredir.

‘Load of the route’ rotada taşınan toplam yük miktarını gösterir. Örneğin 0 indexli araç rotasında taşınan toplam yük 16 tondur.

‘Time of the route’ rotada harcanan toplam süreyi gösterir.

‘new routes’ kısmı araçlar için oluşturulan yeni rotaları bir dizide gösterir.

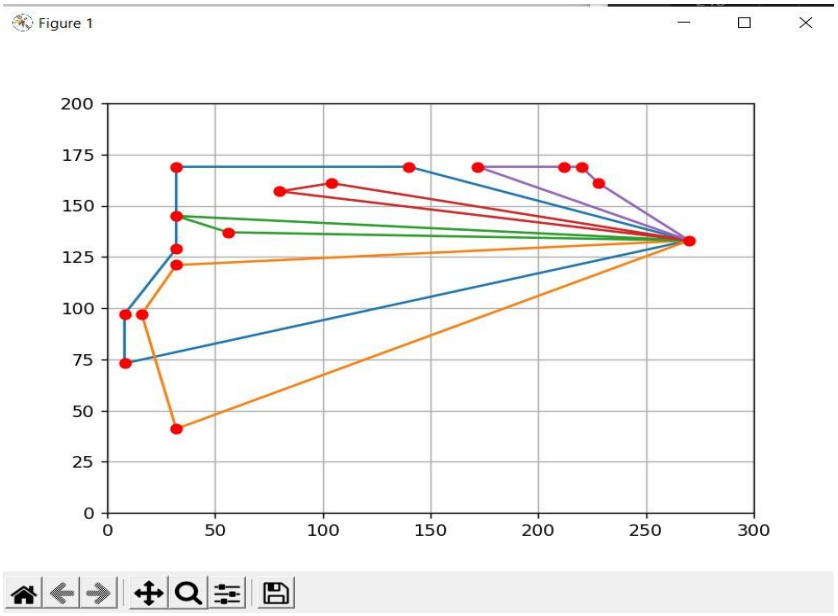
‘Total distance of all route’ rotaların toplam mesafesini gösterir.

‘Total load of all routes’ araçların yolda giderken toplam taşıdıkları yük miktarını basar.

‘Total time of all routes’ araçların rotalar boyunca harcadığı toplam zamanı basar.

```
C:\Users\ahmet\OneDrive\Desktop\STAJ>python ortools_staj.py
Route for vehicle 0:
 0 Load(0) Time(12,12) Slack(0,0) -> 5 Load(0) Time(14,14) Slack(0,0) -> 9 Load(2) Time(25,25) Slack(20,20) -
> 11 Load(3) Time(50,50) Slack(0,0) -> 14 Load(4) Time(55,55) Slack(0,0) -> 15 Load(8) Time(75,75) Slack(0,0)
 -> 0 Load(16) Time(118,118)
Distance of the route: 716m
Load of the route: 16
Time of the route: 118

Route for vehicle 1:
 0 Load(0) Time(17,17) Slack(0,0) -> 12 Load(0) Time(20,20) Slack(2,2) -> 13 Load(2) Time(32,32) Slack(0,0) -
> 16 Load(6) Time(52,52) Slack(0,0) -> 0 Load(14) Time(95,95)
Distance of the route: 692m
Load of the route: 14
Time of the route: 95

Route for vehicle 2:
 0 Load(0) Time(23,23) Slack(0,0) -> 8 Load(0) Time(25,25) Slack(0,0) -> 10 Load(8) Time(65,65) Slack(0,0) ->
 0 Load(10) Time(78,78)
Distance of the route: 500m
Load of the route: 10
Time of the route: 78

Route for vehicle 3:
 0 Load(0) Time(23,23) Slack(0,0) -> 6 Load(0) Time(25,25) Slack(2,2) -> 7 Load(4) Time(47,47) Slack(0,0) ->
0 Load(12) Time(89,89)
Distance of the route: 436m
Load of the route: 12
Time of the route: 89

Route for vehicle 4:
 0 Load(0) Time(25,25) Slack(10,20) -> 1 Load(0) Time(35,45) Slack(0,10) -> 2 Load(1) Time(50,50) Slack(0,0)
-> 3 Load(2) Time(55,55) Slack(0,0) -> 4 Load(4) Time(65,65) Slack(32,32) -> 0 Load(8) Time(118,118)
Distance of the route: 268m
Load of the route: 8
Time of the route: 118

new_routes:  [[0, 5, 9, 11, 14, 15, 0], [0, 12, 13, 16, 0], [0, 8, 10, 0], [0, 6, 7, 0], [0, 1, 2, 3, 4, 0]]
Total Distance of all routes: 2612m
Total Load of all routes: 60
Total Time of all routes: 498min
```

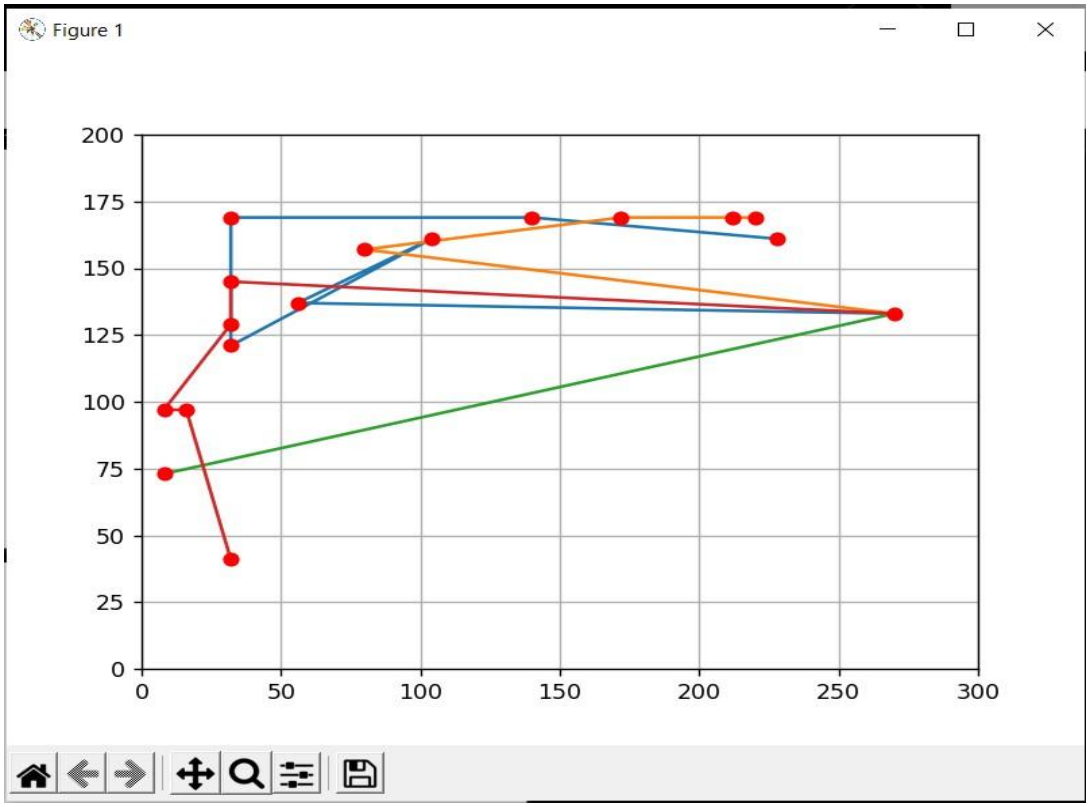
Şekil 2.1 Durum 1'in çıkışı (outputu)



Şekil 2.2 Durum 1 çıkışının (outputunun) görsel çözümü

```
C:\Users\ahmet\OneDrive\Desktop\STAJ>python ortools_staj_yedek.py
Route for vehicle 0:
 1 Load(0) Time(0,0) Slack(0,0) -> 5 Load(1) Time(5,5) Slack(0,0) -> 9 Load(3) Time(14,14) Slack(0,0) ->
 12 Load(4) Time(18,18) Slack(0,0) -> 6 Load(6) Time(27,27) Slack(0,0) -> 8 Load(10) Time(43,43) Slack(1
3,13) -> 0 Load(18) Time(90,90)
Distance of the route: 654m
Load of the route: 18
Time of the route: 90

Route for vehicle 1:
 2 Load(0) Time(21,21) Slack(0,0) -> 3 Load(1) Time(25,25) Slack(0,0) -> 4 Load(3) Time(33,33) Slack(0,0
) -> 7 Load(7) Time(50,50) Slack(0,0) -> 0 Load(15) Time(84,84)
Distance of the route: 366m
Load of the route: 15
Time of the route: 84

Route for vehicle 2:
 15 Load(0) Time(0,0) Slack(0,0) -> 0 Load(8) Time(36,36)
Distance of the route: 0m
Load of the route: 8
Time of the route: 36

Route for vehicle 3:
 16 Load(0) Time(0,0) Slack(0,0) -> 13 Load(8) Time(32,32) Slack(0,0) -> 14 Load(12) Time(48,48) Slack(0
,0) -> 11 Load(16) Time(64,64) Slack(0,0) -> 10 Load(17) Time(68,68) Slack(32,32) -> 0 Load(19) Time(111
,111)
Distance of the route: 402m
Load of the route: 19
Time of the route: 111

new_routes:  [[1, 5, 9, 12, 6, 8, 0], [2, 3, 4, 7, 0], [15, 0], [16, 13, 14, 11, 10, 0]]
Total Distance of all routes: 1422m
Total Load of all routes: 60
Total Time of all routes: 321min
```

Şekil 3.1 Durum 2'nin çıkışı (outputu)



Şekil 3.2 Durum 2 çıkışının (outputunun) görseli

Kaynaklar

https://developers.google.com/optimization

https://developers.google.com/optimization/routing

http://google.github.io/or-tools/javadoc

https://stackoverflow.com/questions/tagged/or-tools

https://github.com/google/or-tools/tree/stable/examples/python

https://groups.google.com/g/or-tools-discuss?pli=1

https://developers.google.com/optimization/reference/python/constraint_solver/pywrapcp

https://matplotlib.org/stable/tutorials/introductory/quick_start.html

https://matplotlib.org/stable/gallery/index.html

https://numpy.org/doc/stable/reference/generated/numpy.arange.html

Serge Kruk, Laurent Perron 2013 https://github.com/google/or-tools