Altın Toplama Oyunu

Tunahan ÇELİK , Harun Aydın

180201130 180201004

Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi
celiktnhn@gmail.com, harunaydin99@gmail.com

Özet

Altın Toplama Oyunu; mxn boyutlu bir dikdörtgen tahta üzerinde farklı özelliklere sahip olan oyuncuların altın toplama yarışına dayanır. Oyuncular belli bir sıra ile hareket ederek altınları toplar her oyuncunun kendine has özellikleri algoritmaları ile tanımlanmıştır. Sistem öncelikle varsayılan puanlamalar ile çalışır ancak bu değerleri kullanıcı istediği şekilde değiştirebilir. Oyun bitiminde her oyuncunun izlemiş olduğu yol oyuncu isimlerinde .txt uzantılı dosyalara yazdırılır. Oyunun oynaşının izlendiği harita x ve y koordinat sistemine göre çalışmaktadır. Bu projede amaçlarımız farklı kısıtlara sahip arama algoritmalarının birbirlerine karşı etkinliklerini gözlemlemek, arama algoritmalarını bir uygulama içerisinde kullanma ve kodlama becerisini geliştirmek, dinamik özelliklere sahip bir program geliştirmek.

1. Giriş

JavaFx ile projenin isterleri doğrultusunda görseliteyi ; m x m boyut bilgileriyle mevcut oyun tahtası ve bunu ayırt edici iki renk belirleyip renklendirilerek oluşturulmuştur. Oyun tahtasında kullanıcının isteği üzerine belirlenmiş oranda altınlar serpiştirilir ve yine kullanıcının belirlediği oranda bu altınların bazıları gizli altındır .Bu gizli altınlar diğerlerinden farklı olarak turuncu renkle belirtilmiştir. Gizli altınlar kullanıcıya görünür iken, oyuncular gizli altınların sayı ve konumundan habersizdir. Gizli altının üstünden geçen oyuncu bu gizli altını artık normal altın olarak değiştirir. Oyun dört oyuncu , belirli dört ayrı köşeden oyun tahtasına yerleştirilmiş şekilde başlatılır . Her oyuncunun , sırayla koşullu hedefler doğrultusunda altınlara ulaşma istikametleri belirlenip bu yolu izlemeleri ile devam eder. Bu süreçte oyuncunun altın puanı ve var olan altın sayıları kontrol edilerek bu kontrolde ; oyuncuların tümünün altın puanlarının 0 a eşit olması veyahut oyun tahtasındaki tüm altınların toplanmış olması ve hiç altın kalmamış olması koşulları gözetilerek oyunun bitişi belirlenir.

Bu oyunu kodlarken temelde bir arama algoritması geliştirilmiştir oyuncular temel oalrak bu arma algoritmasını çalıştırmakta kendi özelliklerine göre işlemektedir. Oyun tahtası x-y koordinatlarına göre belirlenmiş 2 boyutlu maktiste arka planda kareler tutulmakta ve kare tipinde bir sınıfımız bulunmaktadır.Bu tahta;

Soldan sağa doğru

X=0,1,2,3,4..... m

Yukarıdan Aşağıya doğru

Y=0,1,2,3,4.....n

Oalrak kodlanmıştır bu sayede bir koordinat sistemi varmış gibi hareket edilmektedir. Oyunculara ait Boyama işlemleri JPanele ait paintcomponent özelliği ile yapılmıştır.

1. 1 Temel Bilgiler

Program JAVA programlama dilinde geliştirilmiş olup, tümleşik geliştirme ortamı olarak "JetBrains Intellij Idea 2020.2 " kullanılmıştır.

2. Yöntem

Oyun 4 oyuncu ile oynanmaktadır. Oyun özdeş karelerden oluşan ve dinamik olarak boyutu belirlenebilen dikdörtgen bir tahta üzerinde gerçekleştirilir (varsayılan boyutlar: 20x20 kare).

Her bir oyuncu tahtanın ayrı bir köşesinden oyuna başlar.
Tahtadaki karelerin bir kısmında altın bulunurr. Altınlar Karelere rastgele dağıtılır ve altın bulunan kare sayısı toplam kare sayısının belli bir oranında olur. (Varsayılan olarak %20. Örneğin 400 kare için rastgele 80 kare). Altınların bulunduğu karelerin de varsayılan %10'unda gizli altın bulunur bu altınları başlangıçta oyuncular göremeyecektir.
Altın bulunan her bir karedeki altın miktarı 5'in katlarından (5 ile 20 arasında) herhangi biri olabilir.
Her kullanıcı sırayla hamle yaparak oyunu oynayacaktır.
Bir oyuncu altın bir karenin üzerinden geçtiğinde altını alacaktır.
Her kullanıcının başlangıçta eşit ve belli bir miktar altını bulunmaktadır (varsayılan 200 altın).Altını biten oyuncu elenmektedir.

Oyun tüm oyuncuların altını bittiğinde veya karelerde altın tamamen tükendiğinde bitecektir.
Her oyuncu bir seferde ileri geri ya da sağ sol şeklinde hareket edebilir. Her hamlede belli bir adım sayısı kadar hareket edebilecektir (varsayılan 3 adım).
Sırası gelen oyuncular ilk başta gideceği altını tespit etmeleri gerekmektedir. Her hedef belirlemenin bir maliyeti vardır ve her oyuncu için bu maliyet değişmektedir. Oyuncular hedefi olmadan hamle yapmamalıdır. Oyunda yeniden hedef belirmeyi gerektiren üç durum bulunmaktadır. Birinci durum: Oyuncu hedeflediği altına ulaşmıştır ve bir sonraki alacağı altını hedeflemesi gerekir. İkinci durum: Başka bir oyuncu hedeflenen altını kapmıştır bu yüzden başka bir altın hedefi belirlenmelidir. Üçüncü durum: Oyunun başında oyuncuların herhangi bir hedefi olmadığı için adım atmadan önce bir altını hedeflemelidir.

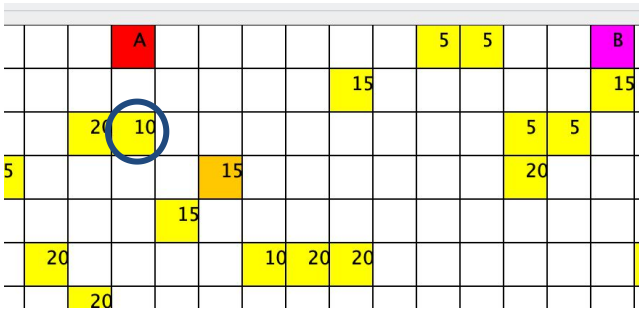
3.1 Algoritma A Oyuncusu

(Herbir oyuncunun kendine ait algoritması bulunmaktadır. Temelde bir arama algoritamsına sahiptirler ve temeli baz alarak arama yaparlar ancak kendi özelliklerine göre bu arama algoritması değişir 3.5 te Temel Algoritma anlatılmıştır.)

Program kullanıcının isteğine bağlı altın miktarı, puanlama, hamle puanları ve hareket puanları, adım sayıları gibi değişkenleri tanımlamakla başlar. A komşu kareleri gezmeye başlar komşu kareleri gezdikçe içerisine o kareye kaç adımda geldiğini tutan Kare tipinde sınıfımından oluşturduğumuz bilgileri kaydeder.

Bu araştırma esnasında eğer tipi 2(Altın) olan bir kareye denk gelirse arama işlemi sonlandırılır ve o bulduğu altından geriye doğru başlangıç noktasına kadar geriye giderek her kareyi bir diziye atar sonra bu diziyi ters çevirir yolun düzgün halini elde etmiş oluruz. Timer nesnesi ile her 3 saniyede bir bu yolun verilen adım sayısına göre boyama işlemlerini geçrçekleştiririz.

Örneğin:



Permütasyon A oyuncusu etrafındaki kareleri keşfederken beyaz kareleri +1 adım uzaklıkta ve hepsi boş kare olduuğu için aramaya devam edecek sonra bu beyaz karelere +1 uzakıkta diğer kareleri keşfetmeye başlayacak bu işelemi bir öncelik dizisine atarak gerçekleştiriyoruz. Tüm keşfettiği komşuları bir diziye atıyor onları iceliyor sonra temizliyor bu komşuların komşularını yeniden bir diziye atıyor inceliyor ve temizliyoruz bu işlem sarı renkli yani tipi 2 olan kareyeyi bulana kadar devam ediyor şekilde A nın gitmesi gereken en yakın altın 10 olarak işaretlenmiştir bu şekilde A oyuncusu Altını yada oyundaki altın bitene kadar aramaya devam ediyor. Yolda ilerleme durumunu bir timer yardımıyla gerçekleşiyor.

Şimdi elimizde gidilebilecek ihtimallerin olduğu tek yönlü bağlı listemiz vardır.

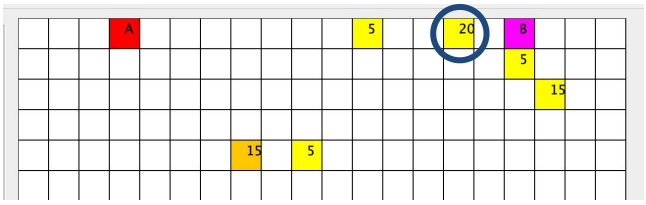
3.2 Algoritma B Oyuncusu

B Oyuncusu kendisine en yakın en kârlı kareyi hedeflemektedir. Bundan dolayı B oyuncusu algoritması sayesinde öncelikle tüm karelerde arama yapar bu bulduğu kare altına yani tipi 2 olana eşit ise içerisindeki altın miktarını kendine ait formüle tabi tutar bu formül:

Maliyet = (Bulduğum karenin altın miktarı) - (toplam adımsayım/ bir defada gidelecek adım sayısı)*bHamlePuanı

Bu işlemi tüm ziyaret ettiği kareler için yapılır. Her karenin özelliğine bu maliyet bilgisini ekler ve altın sahibi karelere ait Kare tipinde bilgi tutan maliyetb Arraylistesine atılır. Tüm altınlar ziyaret edilince bu arraylistesini maliyet değerine göre küçükten büyüğe sıralanır ve en büyük kârlı karem artık benim bitiş noktam olarak belirlenir. Byolçiz() fonksiyonu çağırılır başlangıç noktamdan başlayarak bitiş noktamı arar ve bulduğu anda en kısa şekilde geriye doğru giderek yol arraylistesine atılır Sonra liste ters çevrilerek yol ouşturuşur timer nesnesi ile oyuncu 3 saniyede bir bu yol üzerinde istenilen adımda hareket eder.

Örneğin;



Şekilde işaretlendiği üzere B oyuncusu kendisne en yakındaki 5 altına gitmek yerine solundaki 20 altına girmeyi seçecektir. Hamle puanım eğer varsayılan 5 ve Hedef belirleme puanım 5 olduğunu düşünürsek formülden yola çıkarak kazanacak olduğumuz altın miktarı 10 olacaktır o yüzden B oyuncusu 20 altına gitmiş olacaktır.

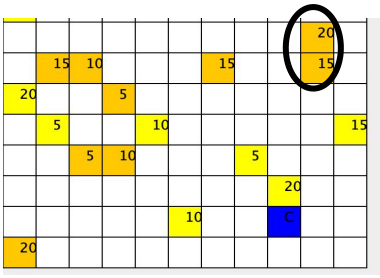
3.3 Algoritma C Oyuncusu

C Oyuncusu kendisine en yakın en kârlı kareyi hedeflemektedir. Aynı B oyuncusu gibi çalışmaktadır ancak C oyuncusu kendisine en yakın 2 tane gizli altını her hedef belirleme öncesi açığa çıkarma özelliğine sahiptir. Bundan dolayı C oyuncusu algoritması sayesinde öncelikle tüm karelerde arama yapar arama esnasında denk geldiği ilk 2 gizli altın kare kendisine en yakın olarak düşünür ve tipini 2 yani altına çevirir arama esnasında bulduğu kare altına yani tipi 2 olana eşit ise içerisindeki altın miktarını kendine ait formüle tabi tutar bu formül:

Maliyet = (Bulduğum karenin altın miktarı) - (toplam adımsayım/ bir defada gidelecek adım sayısı)*cHamlePuanı

Bu işlemi tüm ziyaret ettiği kareler için yapılır. Her karenin özelliğine bu maliyet bilgisini ekler ve altın sahibi karelere ait Kare tipinde bilgi tutan maliyetc Arraylistesine atılır. Tüm altınlar ziyaret edilince bu arraylistesini maliyet değerine göre küçükten büyüğe sıralanır ve en büyük kârlı karem artık benim bitiş noktam olarak belirlenir. Cyolçiz() fonksiyonu çağırılır başlangıç noktamdan başlayarak bitiş noktamı arar ve bulduğu anda en kısa şekilde geriye doğru giderek yol arraylistesine atılır Sonra liste ters çevrilerek yol ouşturuşur timer nesnesi ile oyuncu 3 saniyede bir bu yol üzerinde istenilen adımda hareket eder.

Örneğin;



Şekilde görüldüğü üzere C oyuncusu hedef beliremeden önce kendisine en yakında bulunanan 15 altın ve 20 altın gizli karelerini sarı renkli altın kareye çevirecektir. Sonra aynı B oyuncusu gibi kendisine en yakın kârlı kareye gidecektir.

3.4 Algoritma D Oyuncusu

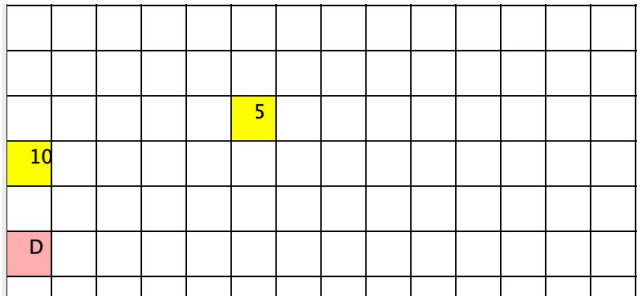
D yuncusu diğer oyunculardan farklı olarak diğer oyuncuların nereye gideceklerinden haberdardır. Öncelikle tüm kareleri araştırmaya başlar bu kareler içerisinde A, B, C oyuncusunun başlangıç ve bitiş noktarına eşit bir kare var ise;
Manhattan Distance(Uzaklığı) ile bu noktanın kensine olan uzaklığına bakmaktadır. Örneğin;

(a,b) ve (x,y) noktaları arası manhattan uzaklığı = |a-x|+|b-y| şeklinde hesaplanmıştır. Eğer bu oyuncuların varayacağı noktaya olan uzaklıkları benim başlangıç noktamdan varacakları noktama daha büyük D için gidilecek yer artık orasıdır eğer değilse D Oyuncusu kendisine en yakın en kârlı kareyi hedeflemektedir. Bundan dolayı BDoyuncusu algoritması sayesinde öncelikle tüm karelerde arama yaptığı anda bu bulduğu kare altına yani tipi 2 olana eşit ise içerisindeki altın miktarını kendine ait formüle tabi tutar bu formül:

Maliyet = (Bulduğum karenin altın miktarı) - (toplam adımsayım/ bir defada gidelecek adım sayısı)*dHamlePuanı

Bu işlemi tüm ziyaret ettiği kareler için yapılır. Her karenin özelliğine bu maliyet bilgisini ekler ve altın sahibi karelere ait Kare tipinde bilgi tutan maliyetd Arraylistesine atılır. Tüm altınlar ziyaret edilince bu arraylistesini maliyet değerine göre küçükten büyüğe sıralanır ve en büyük kârlı karem artık benim bitiş noktam olarak belirlenir. Dyolçiz() fonksiyonu çağırılır başlangıç noktamdan başlayarak bitiş noktamı arar ve bulduğu anda en kısa şekilde geriye doğru giderek yol arraylistesine atılır Sonra liste ters çevrilerek yol ouşturuşur timer nesnesi ile oyuncu 3 saniyede bir bu yol üzerinde istenilen adımda hareket eder.

Örneğin;



Şekilde görüldüğü üzere D oyuncusu hedef belirlememe esnasında 5 altınlı kareye diğer oyuncuların hedefinde olduğunu ve o kareye ise onlardan daha önce olduğunu keşfettiği için yakındaki 10 altına gitmek yerine 5 altına gitmeyi seçecektir.

3.5 Tüm oyuncuların Temelindeki Arama Algoritması

Algoritma başlangıçta bütün karelere henüz erişim olmadığını kabul ederek kare adım sayılarını -1 alınır koordinat sistemi üzerinden başlangıç ve bitiş noktası olan ilgili oyuncuya ait bilgiler alınır başlangıç noktasının adımsayısı 0 olarak kabul edilir ve kordinat sisteminde bu 0 noktasının önce sağ, sol, yukarı ve aşağı noktalarını alınarak bir diziye atılır ayrıcı bu yönlerdeki karelerin tipine bakılır eğer altın yada gizli altın iste ilgili oyuncu bu duruma göre işlem yapar bitiş noktası iste yol dizisine atar hiç biri deil ise adım sayısını kareye işler ver keşfetmiş olduğu komşularının sırasıyla komşularını keşfetmeye başlar. Oyuncular temelde bu şekilde altın arayışında bulunurlar

3.6 Kullanılan Metodlar

aDeneme()
bDeneme()
cDeneme()
dDeneme()
Bu fonksiyonlar oyuncuların belli adım sayısından ilerlemeyi ve atının toplanmasını harita üzerinde işlemlerin gösterilmesini sağlar.
siralaMaliyet() - Maliyet dizisini kar oranına göre sıralar en karlı kare başa getirir.
aYolCiz();
bYolCiz();
cYolCiz();
dYolCiz();
Bitiş noktasına yol array listi oluşturmak için sırasıyla aşağıdaki fonksiyonları çağırır.ve Keşfedilen komşuları öncelik dizisine atar.
yolumuOlusturA();
yolumuOlusturB();
yolumuOlusturC();
yolumuOlusturD();
Gidilecek yol dizisini bulunduğu noktada geriye doğru arrayliste atar.
komsuTipiKesfetYolCizA();
komsuTipiKesfetYolCizB();
komsuTipiKesfetYolCizC();
komsuTipiKesfetYolCizD();
Yol çizme fonksiyonu için bulmuş olduğu kareyi tipine göre keşfeder.
komsuKesfetYolCizA();
komsuKesfetYolCizB();
komsuKesfetYolCizC();
komsuKesfetYolCizD();
Karenin yukarısı aşağısı sol ve sağ noktalarına giderek kare var ise komsitipi kesfete yollar.
aAlgoritması();
bAlgoritması();
cAlgoritması();
dAlgoritması();
Oyuncuların kendine ait özelliklerinin işlendiği ve yol oluşturumalarının sağlandığı fonksiyonlar.

3.7 Sözde Kod

Program çalıştığında sol kısımda varsayılan değerlele birlikte işlem yapma özelliği bulunur istenildiği taktir bu değerler değiştirilebilir.

Ayarla Butonuna Basıldı
Arka Planda Harita oyun bilgirileri kaydedildi
Oyuna Başla Butonuna Basıldı
Sıra A;
aAlgoritması(); çalıştı
Kendine en yakın altını bul
Buluduğun altın karesinden geriye doğru başlangıçtan geldiğin kareleri bir arrayliste
Timer nesnesi yardımı ile istenilen adım sayısında bu yolun ilgli karesini boya.
Sıra B;
bAlgoritması(); çalıştı
Tüm kareleri keşfet
En karlı kareyi maliyetb arreylistinin ilk elemanına ata
bYolçiz(); çalıştır.
En karlı kareyinin kooordinat noktalarını ara
Bulduğun anda geriye doğru geldiğin kareleri yol arreylistine at
Timer nesnesi yardımı ile istenilen adım sayısında bu yolun ilgli karesini boya.

3-cAlgoritması(); çalıştı
Tüm kareleri keşfet
Keşfetme sırasında buludğun ilk iki gizli altını görünür yap
En karlı kareyi maliyetc arreylistinin ilk elemanına ata
cYolçiz(); çalıştır.
En karlı kareyinin kooordinat noktalarını ara
Bulduğun anda geriye doğru geldiğin kareleri yol arreylistine at
Timer nesnesi yardımı ile istenilen adım sayısında bu yolun ilgli karesini boya.
SıraD;
Tüm kareleri keşfet
Keşfetme sırasında diğer oyuncuların bitiş noktalarının kendine olan uzaklığını karşılaştır
Eğer Ondan önce gideceksem bitişimi o nokta yap
Gitmeyeceksem;
En karlı kareyi maliyetc arreylistinin ilk elemanına ata
dYolçiz(); çalıştır.
En karlı kareyinin kooordinat noktalarını ara
Bulduğun anda geriye doğru geldiğin kareleri yol arreylistine at
Timer nesnesi yardımı ile istenilen adım sayısında bu yolun ilgli karesini boya.

4. Deneysel Sonuçlar

Daha öncesinde nesneye yönelik programla üzerinde çalıştığımız için algoritmalarınıda geliştirmekte zorlanmadık ancak çizim işlemlerini yaprırken bir takım hatalar ile karşılaştık bazen bu hatalar RAM den dolayı olabilmekte olduğu gördün yeniden başlatmalarda düzeldi.

(Ekran Görüntüleri 5. Madde Son Sayfada)

6. Kaynakça

1-https://www.tutorialspoint.com/how-can-we-implement-the-paintcomponent-method-of-a-jpanel-in-java

2-https://forum.donanimhaber.com/manhattan-uzakligi-nasil-olculur--92121270

3-https://www.youtube.com/watch?v=_NvD0WzKTC8

5. Ekran Görüntüleri