

**T.C.**

**ESKİŞEHİR OSMANGAZİ ÜNİVERSİTESİ**

**MÜHENDİSLİK-MİMARLIK FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ**

**INTRODUCTION TO ARTIFICIAL INTELLIGENCE**

**DAMA PROJESİ**

**PROJE EKİBİ:**

**121620081030 İLKNUR DUMAN**

**151720081031 İMREN GEÇER**

**152120091011 ABDULLAH AZARKAN**

**152120091013 ENES BATUR**

**PROJE DANIŞMANI:**

**Yrd.Doç AHMET YAZICI**

**İÇİNDEKİLER**

**Giriş……………………………………………………………………….2**

**Peas………………………………………………………………………..2**

**Oyunun Oynanışı…………………………………………………………2**

**Minimax Algoritması…………………………………………………….3**

**Minimax Algoritmasının Uygulanması…………………………………4**

**Oyunun Zorluk Seviyeleri……………………………………………….4**

**Arayüz Taslakları………………………………………………………..5**

**GİRİŞ**

Dama, iki kişinin karşı karşıya oynadığı bir oyundur. Genellikle 10x10 ya da 8x8'lik karelerden oluşan bir tahta üzerinde oynanır. 12x12'lik dama tahtaları da bulunmaktadır. Satrancın aksine bütün taşlar aynı biçimde hareket eder. Yaptığımız dama 8x8'lik karelerden oluşmaktadır.

Projede öncelikli olarak algoritmaların nasıl çalıştığı üzerine bir çalışma yürütüldü. Bazı algoritmalar incelendikten sonra bir takım kararlar verilerek oyun için kod yazmaya başladık. Burada daha önce yapılan dama oyunu projelerini inceleyip bu projelerden faydalandık

**PEAS**

**Performance measure** : En az sayıda taş kaybederek ve en az sayıda hamle yaparak oyunu kazanmaya çalışmak

**Environment** : Bilgisayar ve Oyuncular

**Actuators** : Minimax Algoritması

**Sensors** : Mouse girdileri

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **TASK ENVIRONMENT** | **OBSERVABLE** | **DETERMINISTIC** | **EPISODIC** | **STATIC** | **DISCRETE** |
| DAMA | FULLY OBSERVABLE | DETERMINISTIC | SEQUENTIAL | STATIC | DISCRETE |

**OYUNUN OYNANIŞI**

**Başlangıç Pozisyonu**

Tüm oyuncular tahtanın iki ucunda 12 adet taşla başlarlar. Taşlar birer aralıklı olarak ve bir üst satırlar çapraz gelecek şekilde dizilirler. Tahtanın her iki ucundaki ilk satırlar dama satırıdır. Oyuna ilk olarak koyu renkli olan taş sahibi başlar.

**Nasıl Hareket Eder**

Temelde iki çeşit hareket vardır:

Basit bir hareket, çapraz şekilde yapılır. Normal taşlar hareketlerini sadece ileri komşularına giderek yapabilirler. Taşlar sadece koyu karelerde ilerleyebilir. Böylece tüm oyun çapraz hareketlerle devam eder.

Bir atlama hareketi çapraz karelere ileri olarak eğer boş ise yapılır. Dama olan taş bu kuraldan farklı olarak ileri ve geri olarak birer kare atlar. Normal bir taş rakip taşı almak için eğer arkasında atlayabileceği bir boş kare varsa hamle yapabilir, bu hamlenin sonrasında bir hamle şansı daha varsa taş almak için tekrar hamle yapması gerekir. Üzerinden atlanan rakip taş tahta üzerinden alınır. Böylelikle rakip bir taş azalmış olur. Dama bir taş almak için hem ileri hemde geli olarak hamle yapabilir. Hamle sonrasında normal taşlar gibi boş olan ilk kareye atlaması gerekmez. Boş olan herhangi bir kareye yerleştirilebilir. Bu sayede dama olan taşın aleyhinde hamle yapılması gerekliliği ortadan kaldırılmış olur. Hamle sırasında alınacak bir taş varsa başka bir taşla hamle yapılamaz, alınacak konumdaki taş alınmalıdır. Bir kere dama olmuş olan taş tekrar dama pozisyonuna geldiğinde yeniden ek bir özellik kazanması söz konusu değildir.

**Dama Olma ve Dama Olan Taşın Hareket Kabiliyeti**

Eğer oyuncu dama karşı taraftaki dama pozisyonuna gelirse dama olmuş olur. Böylelikle hem ileri hemde geri gitme özelliği kazanmış olur. Dama olan taşı belli etmek amacıyla oyunda farklı bir işaretleme yapılır(oyunda taşın üzerine “D” harfi ekleniyor). Bir dama normal boş bir kareye giderken tek bir kareden daha fazla çapraz hareket edemez. Yani,dama olan taş hem ileri hemde geri olarak çapraz komşusuna gidebilir. Eğer çapraz komşularının herhangi birisinde rakip taş varsa ve rakip taşın bir sonraki çaprazı da boş ise dama olan taş rakip taşı yiyebilir. Dama normal anlatılan kurallara göre bir kez taş yedikten sonra, aynı çapraz istikamette ilerlemek kaydıyla, karşısına farklı bir taş çıkmadığı sürece istediği kadar ilerleyebilir.

**Oyun Nasıl Biter?**

Bir oyuncunun oyunu kazanabilmesi için rakibinin tüm taşlarını hile yapmadan alması gerekir. Ya da rakibinin hamle yapacağı herhangi bir alanın kalmaması gerekir.

**MINIMAX ALGORİTMASI**

**Minimax Tree Nedir?**

Bilgisayar mühendisliğinde, yapay zeka konusunda kullanılan bir karar ağacı türüdür. Aslında minimax ağaçları bilgisayar bilimlerine işletme bilimindeki oyun teorisinden (game theory) girmiştir.

Temel olarak sıfır toplamlı bir oyunda (zero sum game), yani birisinin kaybının başka birisinin kazancı olduğu (veya tam tersi) oyunlarda karar vermek için kullanılışlıdırlar. Örneğin çoğu masa oyunu (satranç, othello, tictactoe gibi) veya çoğu finansal oyunlar (borsa gibi) veya çoğu kumar oyunları sıfır toplamlı oyunlar arasında sayılabilir (yani birisinin kaybı başka birisinin kazancıdır ve sonuçta toplam sıfır olur).

Yukarıda bahsedilen bu oyunlarda doğru karar verilmesini sağlayan minimax ağacı basitçe kaybı asgariye indirmeye (mimize etmeye) ve dolayısyıla kazancı azamiye çıkarmaya (maximize etmeye) çalışır.

Ağaç basitçe her düğümde (node) farklı alternatiflerin değerlerini hesaplar. Son düğümden (yapraklardan ,leaf) yukarıya doğru değerleri seçerek gelir ve en sonunda bütün ağaçtaki en doğru seçenek seçilmiş olur.

**MINIMAX ALGORİTMASININ UYGULANMASI**

Dama oyunu projemizde biz ağaçlardaki dallanmaları şu anki mevcut durumdaki oyuncunun bulunduğu durumu ve yapacağı hareketleri bir array içerisinde tutuyoruz. Oyunculara sıra geldikçe daha başarılı her iki oyucu da oyunu kazanacak dallanmalarının yapraklarının muhtemel koordinatlarını belirliyor.Bu ağaç yapısındaki dallanmalar derecelerine göre ve sıradaki hamlenin hangi oyuncuya ait olduğuna dayanarak sırasıyla max ve ya min (alt-üst) şeklinde adlandırılırlar.

Projemizde minimax’a dayanan oyun stratejimizde, oyuncunun hareket edeceği koordinat, kazanma ihtimali yüksek olan dallanmalar içerisinde en yüksek orana sahip olana göre seçilir. Bilgisayar max node ları arasından seçim yaparken, sonraki min hamlelerini de göz önünde bulundurarak bir seçim yapar. Bütün yapraklardaki değerler, minimax algoritmasındaki fonksiyonu kullanarak geriye bir değer döndürür ve bu değer bulunduğu durumun iyi ya da kötü olduğu hakkında bize bilgi verir.

### Evaluation Fonksiyonu

Min-max algoritmasının oyunlara uyarlanması aşamasında gerekli olan fonksiyonlardan birisi de evaluation fonksiyonudur. Bu fonksiyon basit olarak katkısı, programı tekrar ederek farklı sezgisel ve ağırlıkları oyuna göre hesaplar.

Bu fonksiyonun fazla karmaşık olmasına gerek yoktur. Büyük hesaplama gerektiren oyunlar, bir anlamda büyük ağaç dallanmaları gerektiren oyunlar, fazla node lara sahip olacağından, evaluation fonksiyonunu daha fazla kullanması gerekecek. Eğer evaluation fonksiyonu da karmaşık olursa, program daha derin ağaçları da hesaplayarak, bir bakıma gerekli olmayan hesapları yaparak program karmaşık hale gelecektir. Bu yüzden evaluation fonksiyonu ile bilgisayarın ne kadar derinlikte bir arama yapacağı arasında sağlam bir ilişkisi olması lazım.

### Alpha-Beta Algoritması

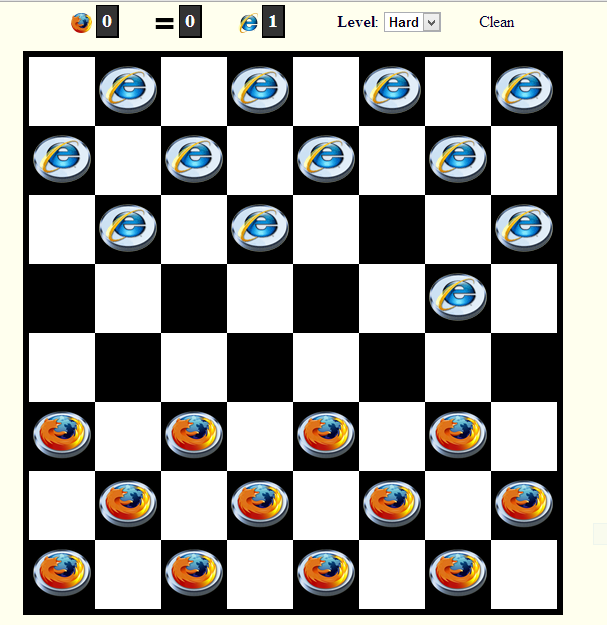
Alfa-Beta metodu bize minimax stratejisinde ortaya çıkan maximum muhtemel hamleleri (Ağaç dallanmaları) kullanıp, tek tek değerlendirme yapmak yerine, dallanmada iyi olmayan hamleleri eleme yöntemi sunuyor. Alfa-Beta metodu arama yapıp muhtemel durumları bulan 8 derinlikteki ağacı birkaç saniyede tarayabilir ve eleme yapabilir. Bu eleme işlemi Alfa-Beta ile değil de sadece min-max ile yapılsaydı, aynı süre zarfında ancak 5 derinlikteki bir ağacı tarayabilir.

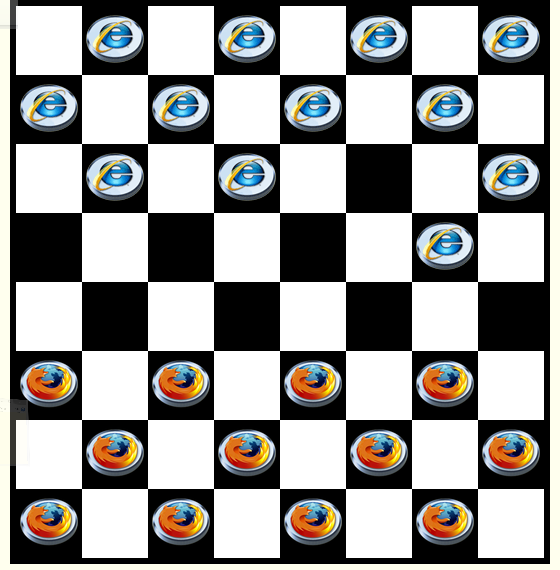
**OYUN ZORLUK SEVİYELERİ**

Yapacağımız dama oyununda zorluk seviyeleri bulunacaktı.Bunu Computer isimli bir sınıfımızın altındaki fonksiyonlarla minimax algoritmasını kullanarak 3,6 ve 9 derinliklerde arama yaptırarak gerçekleştirmeyi düşünüyoruz.

**ARAYÜZ TASLAKLARI**

Henüz arayüzlerimiz tam hazır olmadığı için ,altta örnek alacağımız arayüz bulunmaktadır.

****

****