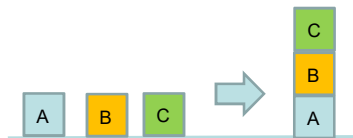


Arama

Arama Her Yerde

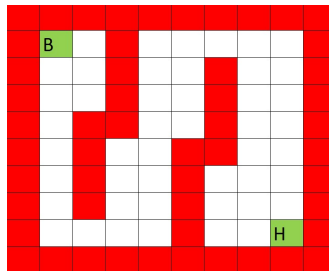
Planlama

- Problem: Masadaki küpleri mevcut pozisyondan istenilen pozisyona getirmek
- Olası hareketler: küp tutma, küpü yere koyma, küpü bir başka küpün üzerine koyma
- Çözüm: Hareketler sırası



Robot Yol Planlama

- Başlangıç noktasından (B) hedef noktasına (H) gitmek.
- Olası hareketler: Sola, Sağa, Aşağıya, Yukarıya
- Olası hareketlerin maliyetleri birbirlerinden farklı olabilir.
- En az maliyetli hareketler dizisini bulmak.



Mehmet Fatih AMASYALI Yapay Zeka Ders Notları

YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Cin Ali Nehirde

http://josquin.cs.depaul.edu/~rburke/courses/f08/comp30030/notes/lec_0911.pdf

Hedef:

Ali küçük bir sandalla, kurtunu, keçisini ve kabağını nehrin karşıısına geçirmek istiyor.

Kısıtlar:

Sandal Ali'yle birlikte en fazla bir tane nesneyi taşıyabiliyor.

Keçi - kabak, kurt - keçi ikilileri Ali yanlarında değilken, nehrin aynı tarafında olmamalı.



Mehmet Fatih AMASYALI Yapay Zeka Ders Notları

YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Problemin Gösterimi

- İfade edilmesi gerekenler
 - 4 nesnenin pozisyonu
 - Sandalı ifade etmeye gerek yok. Çünkü yeri her zaman Ali ile aynı.
 - Nesneler ya kuzeyde (N), ya da güneyde (S)
- Her bir nesneyi pozisyonunu gösteren bir harfle ifade:

farmer
wolf
goat
cabbage
 (A? , B? , C? , D?)

Mehmet Fatih AMASYALI Yapay Zeka Ders Notları

YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Problemin Gösterimi

- İlk durum
 - (S, S, S, S) (A? , B? , C? , D?)
- Yasak durumlar
 - Kurt keçiyi yer (Ali olay yerinde değilse)
 - (N, S, S, _)
 - (S, N, N, _)
 - Keçi kabağı yer (Ali olay yerinde değilse)
 - (N, _, S, S)
 - (S, _, N, N)
- Hedef durum
 - (N, N, N, N)

Mehmet Fatih AMASYALI Yapay Zeka Ders Notları

YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

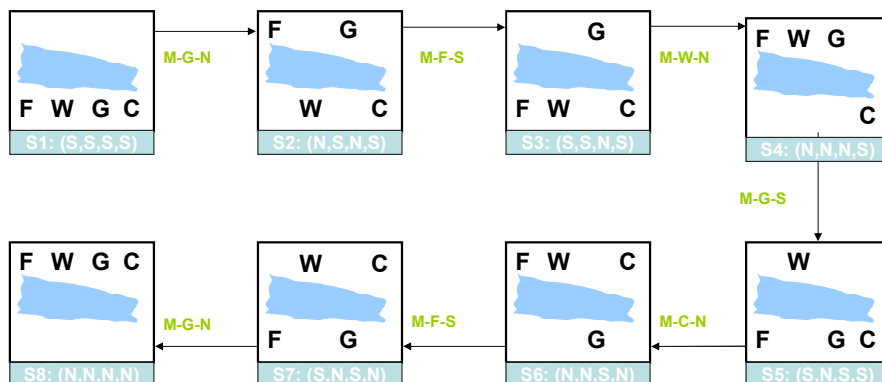
Operatörler

- Aksiyonlar / hareketler / operatörler durumları birbirine dönüştürür.
- Keçiyi kuzeye taşımak
 - Başlangıç durumu: keçi güneyde
 - Bitiş durumu: keçi kuzeyde
 - Diğer nesnelerin yerinin önemi yok.
- Gösterim
 - Move(Goat, North)
 - (S, ?wolf, S, ?cabbage) => (N, ?wolf, N, ?cabbage)
 - ?wolf kurtun yerini gösteren bir değişken
 - Değişken kullanımı bizi çok sayıda kural yazmaktan kurtarır. Burada 4 kural yerine 1 kural yetti.
 - Herbir nesne iki yerde olabildiğine göre toplam $2^4 = 16$ durum (state) var.

Mehmet Fatih AMASYALI Yapay Zeka Ders Notları

YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Bir çözüm örneği

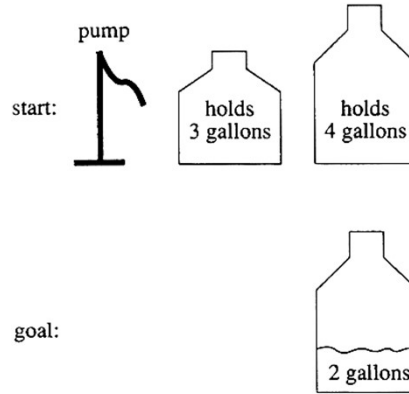


Mehmet Fatih AMASYALI Yapay Zeka Ders Notları

YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Su bidonları (Jugs problem)

- Elinde 3 ve 4 galon hacimlere sahip 2 bidon var.
- Limitsiz bir su kaynağın var. (Pompa)
- İstenen: 4 galonluk bidona tam 2 galon su koyman.



Die Hard: <https://www.youtube.com/watch?v=6cAbgAaEOVE>

Problemin Gösterimi

- Durum (State) gösterimi: (x, y)
 - x : 4 galonluk bidondaki su miktarı
 - y : 3 galonluk bidondaki su miktarı
- Başlangıç durumu: $(0, 0)$
- Bitiş / hedef durumu $(2, n)$

Operatörler

- | | |
|--|--|
| 1 $(x,y) \rightarrow (4,y)$
if $x < 4$ | 4 galonluk bidonu pompa ile doldur |
| 2 $(x,y) \rightarrow (x,3)$
if $y < 3$ | 3 galonluk bidonu pompa ile doldur |
| 3 $(x,y) \rightarrow (0,y)$
if $x > 0$ | 4 galonluk bidonu yere boşalt |
| 4 $(x,y) \rightarrow (x,0)$
if $y > 0$ | 3 galonluk bidonu yere boşalt |
| 5 $(x,y) \rightarrow (4,y - (4 - x))$
if $x + y \geq 4$ and $y > 0$ | 4 galonluk bidon dolana kadar 3 galonluk bidondan su doldur. |
| 6 $(x,y) \rightarrow (x - (3 - y),3)$
if $x + y \geq 3$ and $x > 0$ | 3 galonluk bidon dolana kadar 4 galonluk bidondan su doldur. |

Operatörler

- | | |
|---|---|
| 7 $(x,y) \rightarrow (x + y, 0)$
if $x + y \leq 4$ and $y > 0$ | 3 galonluk bidondaki suyun tamamını 4 galonluğa boşalt. |
| 8 $(x,y) \rightarrow (0, x + y)$
if $x + y \leq 3$ and $x > 0$ | 4 galonluk bidondaki suyun tamamını 3 galonluğa boşalt. |

Bir çözüm

4 galonluk bidondaki su miktarı	3 galonluk bidondaki su miktarı	Uygulanan kural no
0	0	2
0	3	7
3	0	2
3	3	5
4	2	3
0	2	7
2	0	

8 – puzzle*

4	3	6
2	1	8
7		5

 S_0

1	2	3
4	5	6
7	8	

G

Karmaşıklığı azaltmak için taş hareketleri boş karenin hareketleri olarak temsil edilebilir.

Operatörler:

L : Boş kare sola

R : Boş kare sağa

U : Boş kare yukarıya

D : Boş kare aşağıya

$$C(L) = C(R) = C(U) = C(D) = 1$$

[*] <http://aima.cs.berkeley.edu/figures.pdf>

8 puzzle'in arama ağacı

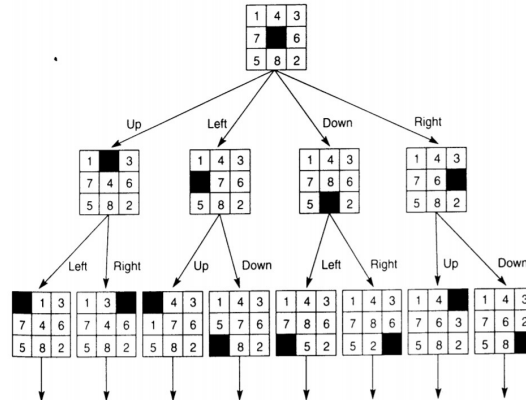


Figure 3.6 State space of the 8-puzzle generated by "move blank" operations.

Ortalama adım sayısı 22, Ortalama dallanma sayısı: 2,67
Tekil olmayan 2.7 milyar durum

Mehmet Fatih AMASYALI Yapay Zeka Ders Notları

YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Tanımlar

- Durum Uzayı, Arama Uzayı, Arama ağacı, State Space, Search Space, Search Tree : Durumları, aralarındaki geçişleri, geçiş parametrelerini ve kısıtlarını gösteren graf/ağaç
- Operatörler : Durumlar arası geçişleri sağlarlar.
- Başlangıç durumu : S_0 (Aramanın başladığı durum)
- Hedef durumu: $\{G\}$ - (Aramanın bittiği durum)
- Maliyet, Cost : Operatörü uygulamanın maliyeti
- Çözüm yolu, solution path: Başlangıç durumundan hedef duruma giden yol
- Optimum Yol, Optimal path : En düşük maliyetli

Mehmet Fatih AMASYALI Yapay Zeka Ders Notları

YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Arama neden zor olabilir?

Varsayımlar:

Bir durumdan gidilebilecek durum sayısı , dallanma sayısı : $b=10$

Saniyede işlenen durum / node sayısı: 1000

Bir durum / node için tutulan bellek miktarı: 100 bytes

Çözümün derinliği	İşlenen durum sayısı	Zaman	Bellek
0	1	1 milisaniye	100 bytes
2	111	0.1 saniye	11 kbytes
4	11,111	11 saniye	1 megabyte
8	10^8	31 saat	11 gigabytes
12	10^{12}	35 yıl	111 terabytes

Arama Algoritmaları / Stratejileri

- Arama stratejileri, bir durumdan diğer bir duruma giderken gidilecek durumun olası gidilebilecek durumlar arasından nasıl seçildiğini belirler.
- Stratejiler birkaç boyutta değerlendirilir:
 - Tamlık (completeness):** Bir çözüm varsa mutlaka bulması
 - Zaman Karmaşıklığı (time complexity):** Aramanın alacağı süre
 - Hafıza Karmaşıklığı (space complexity):** Arama için gereken hafıza miktarı
 - Optimallik (optimality):** ilk bulunan çözümün en düşük maliyetli çözüm olması
- Zaman ve Hafıza karmaşıklıklarının ölçümlemlerinde kullanılan kavramlar
 - b : Maksimum dallanma sayısı (bir düğümden çıkan maksimum düğüm sayısı)
 - d : En az maliyetli çözümün derinliği
 - m : arama uzayının maksimum derinliği (bazen ∞)

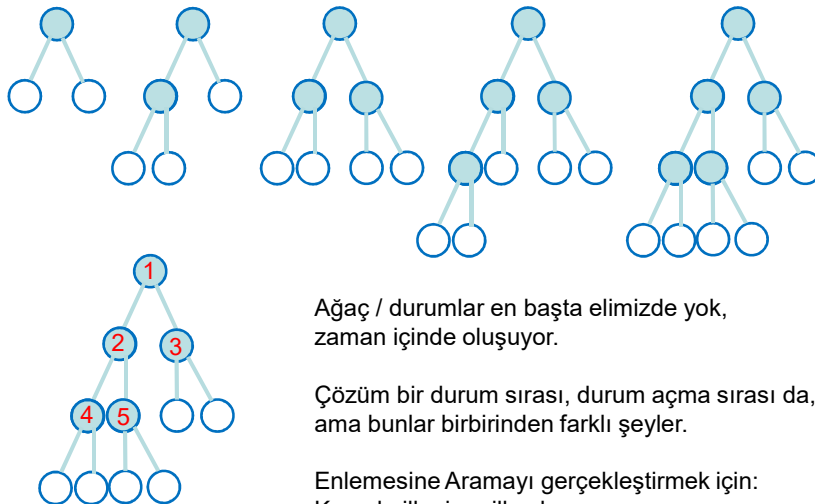
Kör / Mekanik / Bilgisiz / Blind / Uninformed Arama Stratejileri

- Enlemesine Arama / Breadth-first search
- Düşük Maliyetli Arama / Uniform cost search
- Derinlemesine Arama / Depth-first search
- Sınırlı Derinlikte Arama / Depth-limited search
- Artan Derinlikli Arama / Iterative deepening search
- Çift Yönlü Arama / Bidirectional search

Mehmet Fatih AMASYALI Yapay Zeka Ders Notları

YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Enlemesine Arama Breadth-first search



Mehmet Fatih AMASYALI Yapay Zeka Ders Notları

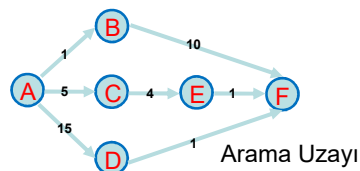
YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

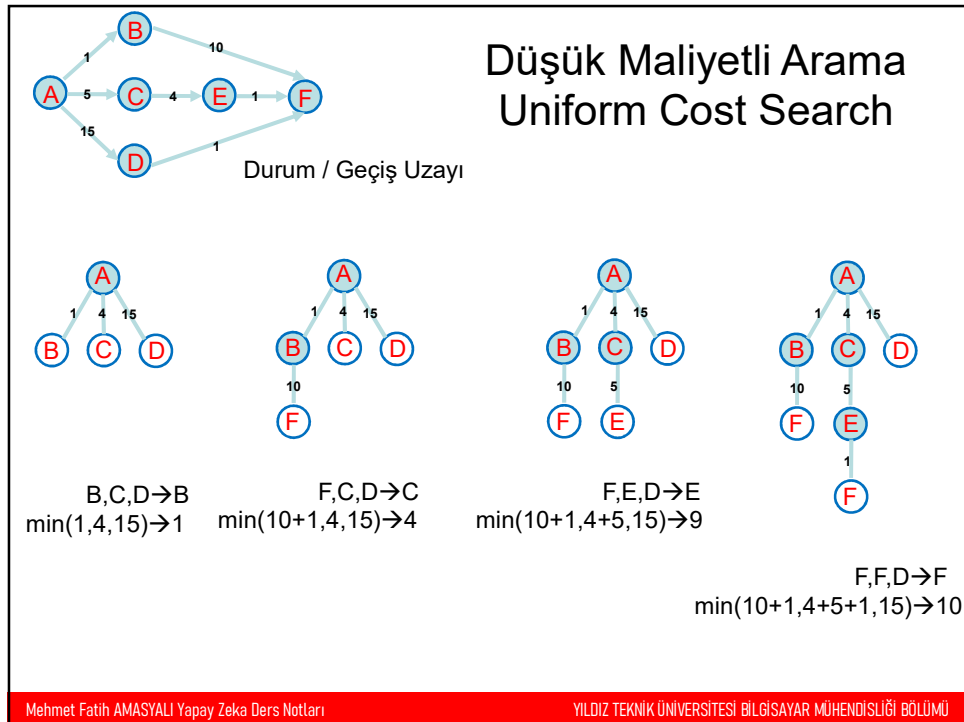
Enlemesine Aramanın Analizi

- Complete? Eğer b sonlu ise evet
- Time? $1+b+b^2+b^3+\dots +b^d = O(b^d)$
- Space? $O(b^d)$ (her node hafızada)
- Optimal? Her adımın maliyeti eşitse evet
- **Hafıza** zamandan daha büyük problem

Düşük Maliyetli Arama Uniform Cost Search

- Enlemesine aramaya benzer.
- Kökten itibaren toplam maliyeti en az düğümü seçer ve genişletir.
- Tüm maliyetler birbirine eşitse enlemesine aramanın aynısı





Düşük Maliyetli Aramanın Analizi

- **Complete?** Eğer b sonlu ve maliyet >0 ise Evet
- **Time?** $O(b^d)$
- **Space?** $O(b^d)$
- **Optimal?** Tüm maliyetler pozitif ise Evet
- **Hafıza** zamandan daha büyük problem

Sınırlı Derinlikte Arama Depth-limited search

= derinlemesine aramanın derinlik sınırlanmış (l) hali
 l derinliğinde olan node'ların genişlemesine izin verme

Artan Derinlikli Arama Iterative deepening search

= for derinlik limiti (l)= 0 to X
Sınırlı derinlikte arama (l)

Artan Derinlikli Arama $l = 0$

Limit = 0



[*] <http://aima.cs.berkeley.edu/figures.pdf>

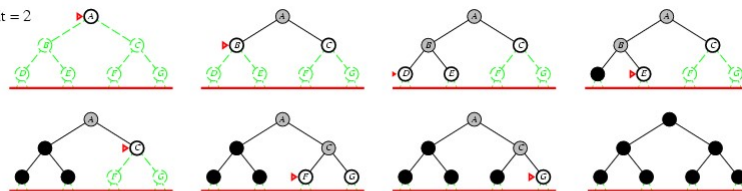
Artan Derinlikli Arama / =1

Limit = 1



Artan Derinlikli Arama / =2

Limit = 2

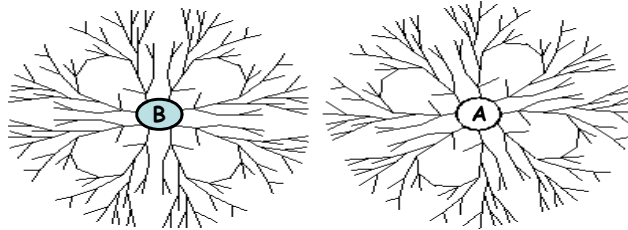


Artan Derinlikli Aramanın Analizi

- Complete? Evet
- Time? $(d+1)b^0 + d b^1 + (d-1)b^2 + \dots + b^d = O(b^d)$
- Space? $O(bd)$
- Optimal? Eğer tüm maliyetler eşitse Evet

Artan derinlikli aramada derinlemesine arama yerine enlemesine arama yapılırsa ne olur?

İki Yönlü Arama Bidirectional Search



- İleri ve geri aramaların her biri sadece yarım yol gider. Enlemesine arama yapılır.
- $b=10$, $d=6$ için her bir yön 3 derinliğinde olur ve oluşturulan düğüm sayısı 2,222 dir. Genişlik öncelikli (enlemesine) aramada bu sayı 1,111,111.
- Complete? Evet
- Time? $O(b^{d/2})$
- Space? $O(b^{d/2})$
- Optimal? Eğer tüm maliyetler eşitse Evet

Enlemesine Arama Algoritması

- kuyruk = [kök durum]
- bulundu = FALSE
- While (kuyruk <> boş) and (bulundu = FALSE)
 - Kuyruktan ilk durumu (N) çek
 - Eğer N hedef durumsa, bulundu = TRUE
 - N'den gidilebilecek tüm durumları kuyruğun sonuna ekle

Düşük maliyetli Arama Algoritması

- kuyruk = [kök durum]
- bulundu = FALSE
- While (kuyruk <> boş) and (bulundu = FALSE)
 - Kuyruktan ilk durumu (N) çek
 - Eğer N hedef durumsa, bulundu = TRUE
 - N'den gidilebilecek tüm durumları kuyruğun sonuna ekle
 - Kuyruktaki durumları kökten maliyetlere göre küçükten büyüğe sırala

Derinlemesine Arama Algoritması

- stack= [kök durum]
- bulundu = FALSE
- While (stack <> boş) and (bulundu = FALSE)
 - Stack'ten ilk durumu (N) çek
 - Eğer N hedef durumsa, bulundu = TRUE
 - N'den gidilebilecek tüm durumları stack'in başına ekle

Tekrarlayan Durumlar

- $A \rightarrow D$
 - $D \rightarrow E$
 - $E \rightarrow A, F$
 - A'dan başlayıp derinlemesine arama yaparsak? F hedef olsun
- Çözüm: yeni durumları eklemeye önce kontrol, varsa ekleme
- A
 - D(AD)
 - E(ADE)
 - A(ADEA) F(ADEF)
 - D(ADEAD) F(ADEF)
 - E(ADEADE) F(ADEF)
 - A(ADEADEA) F(ADEADEF) F(ADEF)
 - D(ADEADEAD) F(ADEADEF) F(ADEF)
 - E(ADEADEADE) F(ADEADEF) F(ADEF)
 - A(ADEADEADEA) F(ADEADEDEF) F(ADEADEF) F(ADEF)
 - ...
- E den A ve F ye gidilebiliyor
A içeride var ekleme, F yok ekle

Karşılaştırma

DFS,

- + Hafıza gereksiminde lineer
- Loop içeren arama uzaylarında sonsuza kadar çalışır
- Optimum çözümü bulmayı garantilemez

BFS,

- + Optimum çözüm
- + Loop lardan kurtulabilir
- Hafıza gereksinimi derinlikle üssel olarak büyüyor

IDS,

- + Lineer hafıza gereksinimi
- + Looplardan kurtulabilir
- + optimum çözümü garantiler

Kör arama stratejilerinin analizi

Criterion	Breadth First	Uniform Cost	Depth-First	Depth-Limited	Iterative Deepening	Bidirectional
Time	b^d	b^d	b^m	b^l	b^d	$b^{\frac{d}{2}}$
Space	b^d	b^d	bm	bl	bd	$b^{\frac{d}{2}}$
Optimal	Yes	Yes	No	No	Yes	Yes
Complete	Yes	Yes	No	Yes, if $l \geq d$	Yes	Yes

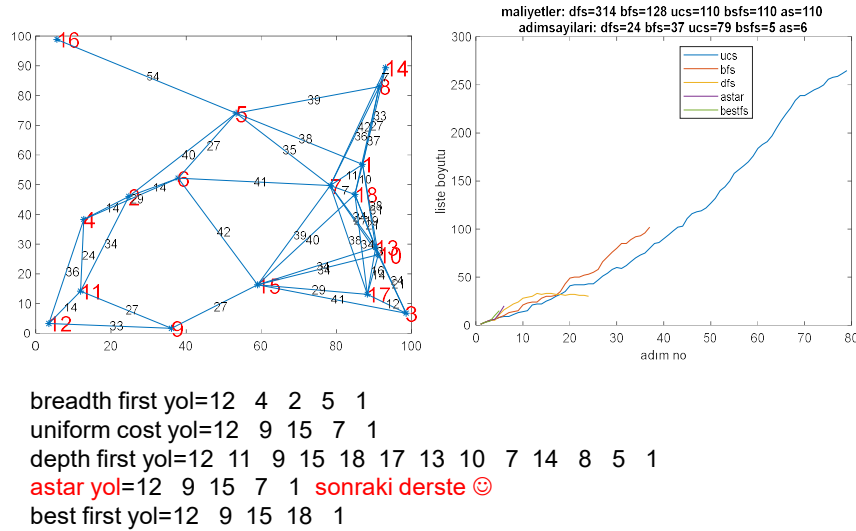
b — branching factor

d — depth of shallowest solution

m — maximum depth of tree

l — depth limit

Uygulama my_search.m



Mehmet Fatih AMASYALI Yapay Zeka Ders Notları

YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Olası dünyalar

- Hergün iki seçeneği olan bir karar verdiğimizizi düşünelim. Bu durumda kararımızın etkisine göre iki olası dünyadan birinde yaşarız.
- 32 gün sonra 2^{32} = yaklaşık 4.3 milyar olası dünyadan sadece birinde yaşıyor oluruz.
- Başka Faktörler
 - Doğduğumuzdan beri geçen zaman
 - Başkalarının kararlarının bizim dünyamıza etkisi
 - 2'den fazla seçeneği olan durumlar

Mehmet Fatih AMASYALI Yapay Zeka Ders Notları

YILDIZ TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Kaynaklar

- <http://aima.cs.berkeley.edu/>