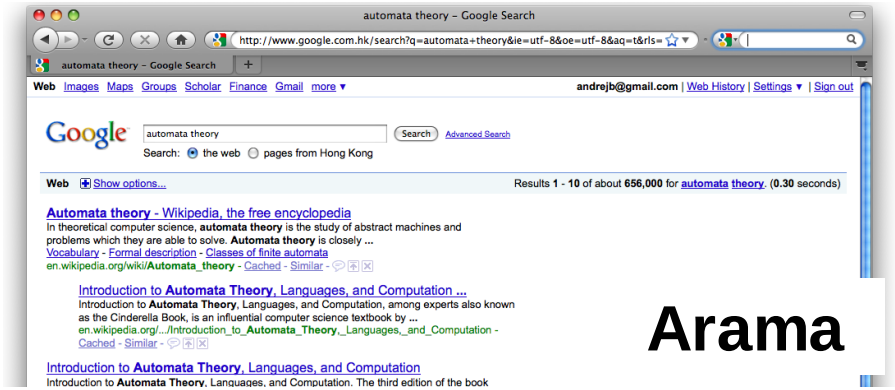
PAMUKKALE ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
2021 BAHAR

Biçimsel Diller ve Otomata Teorisi

Formal languages and automata theory

Bilgisayarların Yetenekleri

Başka Neler?

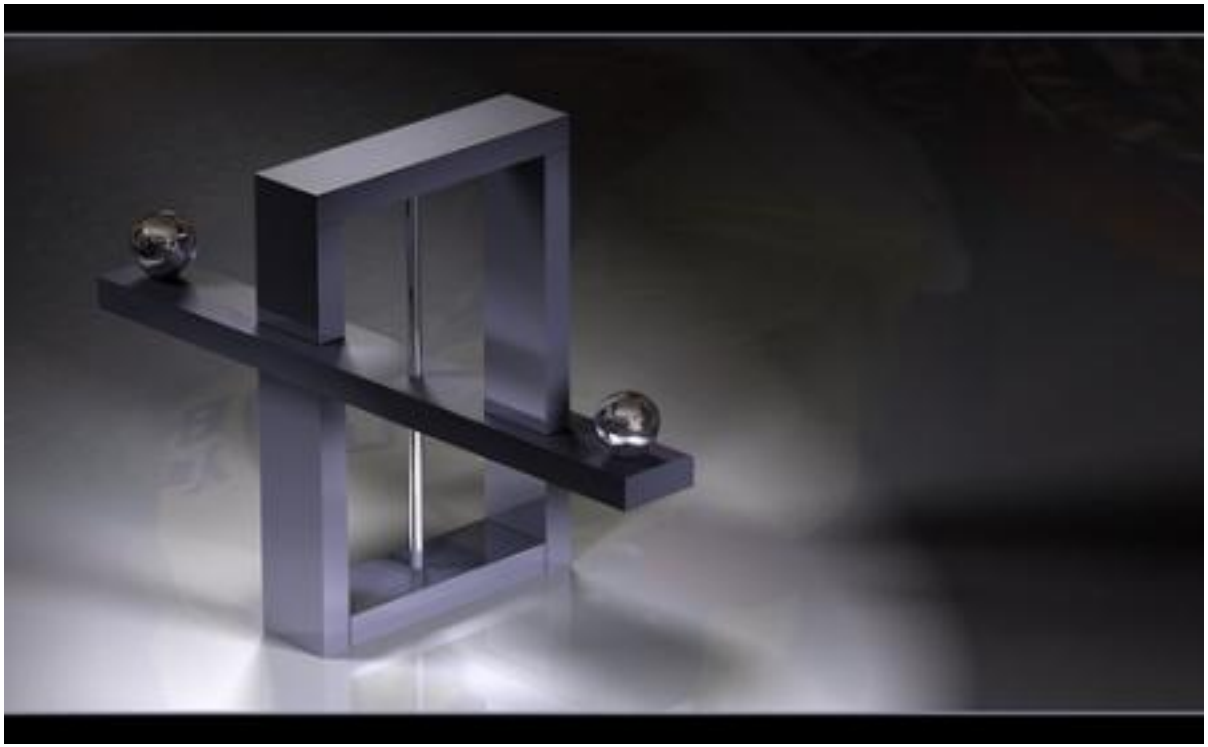


Arama

Otomatik Uçuş

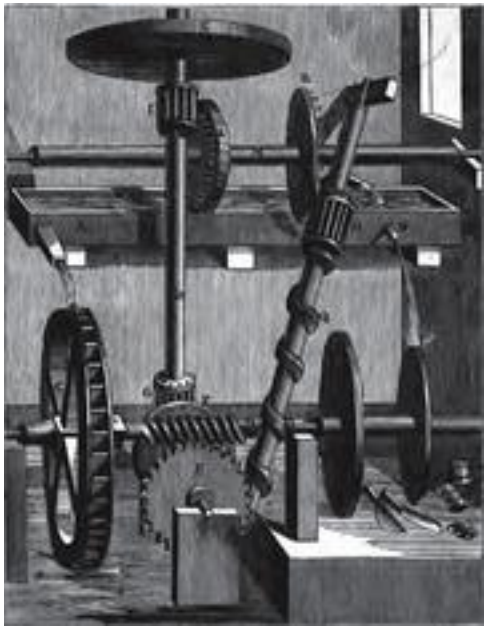
Bir bilgisayarın yapamayacağı bir şey var mı?

İmkansızlık



İmkansız olan durumları neden önemsiyoruz?

Ebedi Hareket (Perpetual motion)



Orta çağlarda insanlar hiç enerji kullanmayan bir makine istiyorlardı.

Daha sonra fizikteki keşifler, enerjinin yoktan var edilemeyeceğini gösterdi.

Perpetual motion → boş çaba

İmkansızı anlamak, enerjimizi daha yararlı olana yönlendirmemize yardımcı olur.

Hesaplama Yasaları

Tıpkı fizik yasaları bize doğada olması mümkün olan veya olmayanları söylemesi gibi ...

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon_0} \qquad \nabla \cdot \mathbf{B} = 0 \qquad \nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \qquad \nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t}$$

... hesaplama yasaları bize bilgisayarlar için neyin mümkün olduğunu veya olmadığını söyler.

Hesaplanabilirlik (Computability)

- Hesaplanabilirlik, hesaplama probleminin bir özelliğidir. Daha özel olarak, sınırlı zamanda sınırsız belleğe sahip bir bilgisayar tarafından gerçekleştirilebilen problemi çözen bir algoritma varsa, bir problem hesaplanabilirdir. "Sonlu zamanın" üst sınırının ne olduğu önemli değil; bu sorunu çözen bir algoritmayı çalıştırmak için üssel olarak çok sayıda adım veya keyfi olarak büyük miktarda bellek gerekebilir, ancak hesaplanabilirse, algoritmanın sınırlı bir süre içinde sonuçlanacağını biliyoruz. Bununla birlikte, neyin hesaplanabilir olması gerektiğine ilişkin bu sınırlandırılmamış bir sınırla bile, hala hesaplanamayan, yani bunları çözmek için böyle bir algoritmanın olmadığını kanıtlayabileceğimiz sorunlar vardır.

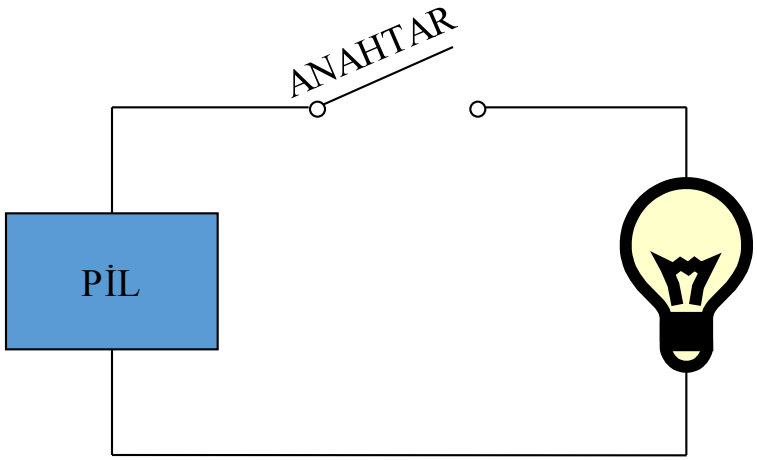
Otomata Teorisi

Otomata teorisi hesaplama yasalarını inceler.

- Otomata teorisi, soyut hesaplama cihazlarının incelenmesidir
- Soyut cihazlar, gerçek hesaplamaların (basitleştirilmiş) modelleridir
- Hesaplamalar her yerde olur: Dizüstü bilgisayarınızda, cep telefonunuzda, doğanın içinde ...
- Neden soyut modellere ihtiyacımız var?

Gerçekte, hesaplama yasaları tam olarak anlaşılmamıştır, ancak otomata teorisi iyi bir başlangıçtır.

Basit bir bilgisayar



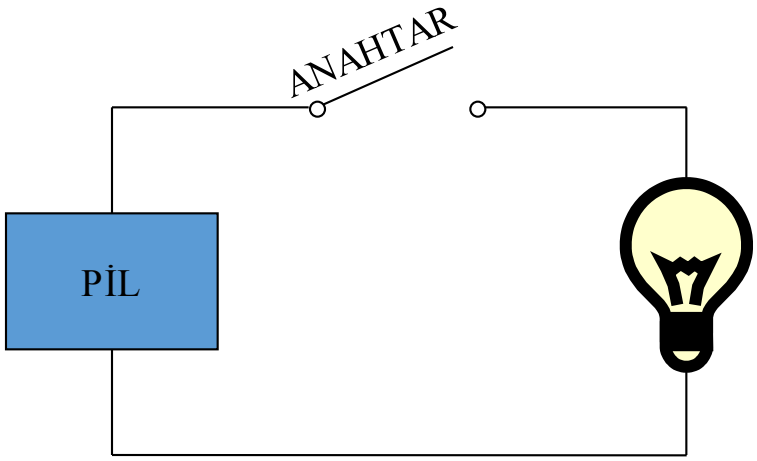
giriş: anahtar

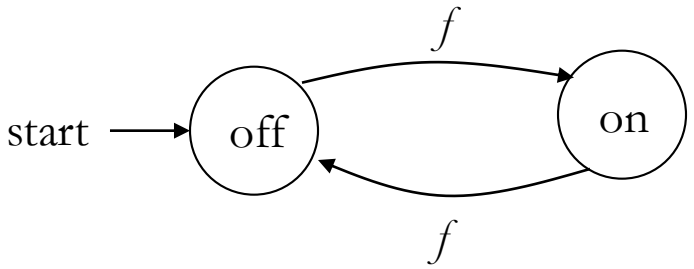
çıkış: lamba

eylem: anahtarın çevrilmesi

states: açık/kapalı on/off

Basit bir bilgisayar





ampul, ancak ve ancak tek sayıda çevirme varsa yanar (başlangıç: off)

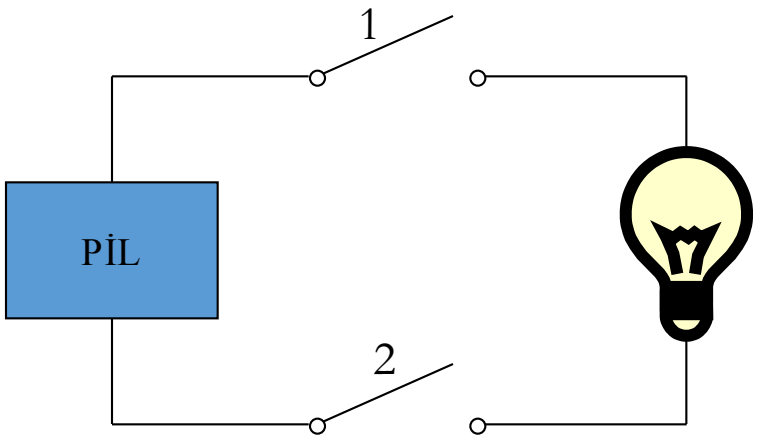
giriş: anahtar

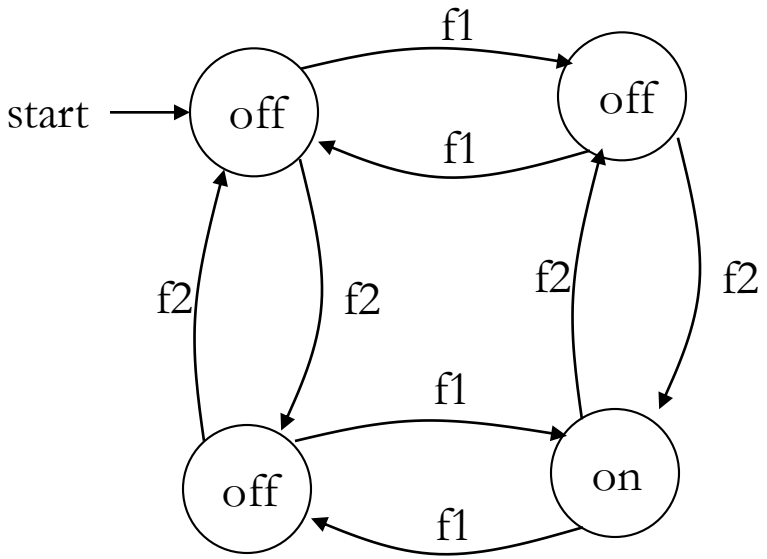
çıkış: lamba

eylem: *f* (anahtarın çevrilmesini ifade eden eylem)

states: *on/off*

Başka bir “bilgisayar”





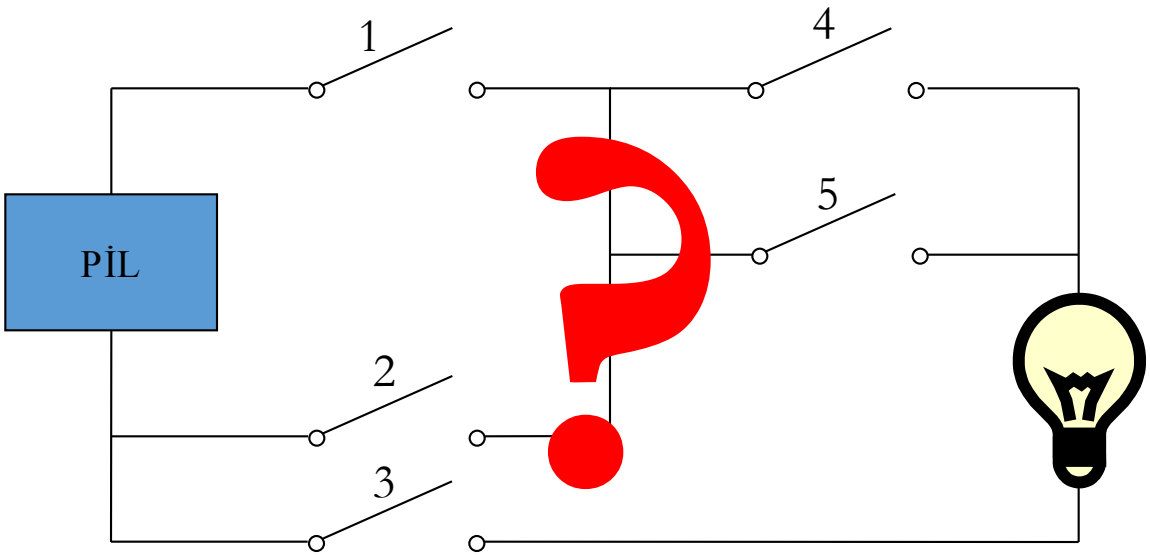
inputs: anahtar 1 ve 2

actions: f1 “anahtar 1 için”
f2 “anahtar 2 için”

states: on, off

ampul ancak ve ancak her iki düğme de tek sayıda çevrildiğinde yanar.

Tasarım Problemi



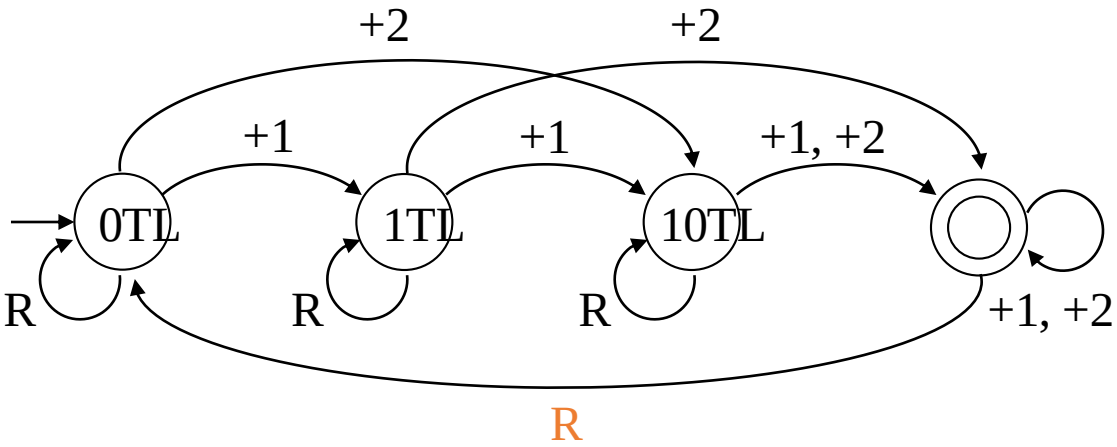
Yalnızca ve ancak tüm anahtarlar tam olarak aynı sayıda çevrildiğinde ışığın açık olduğu bir devre tasarlayabilir misiniz?

Bir sakız makinesi

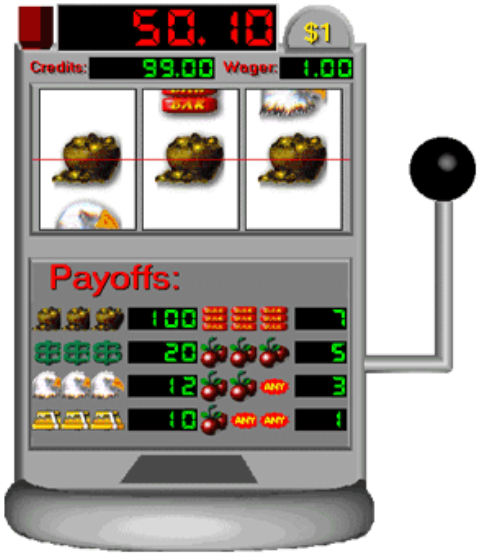
Makine 1TL ve 2TL ile bozuk ile çalışsın.

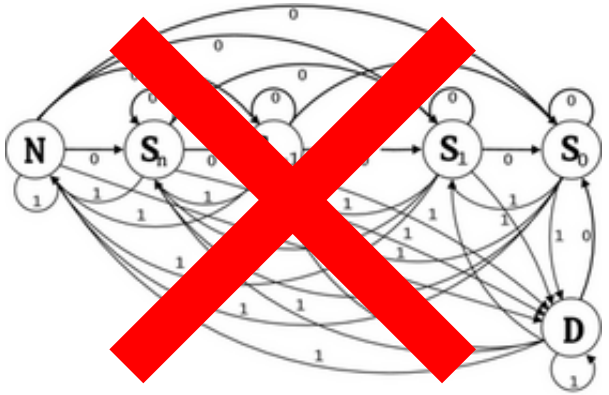
Bir sakız 3 TL olsun

eylemler: +1 at, +2 at, çevir (R)

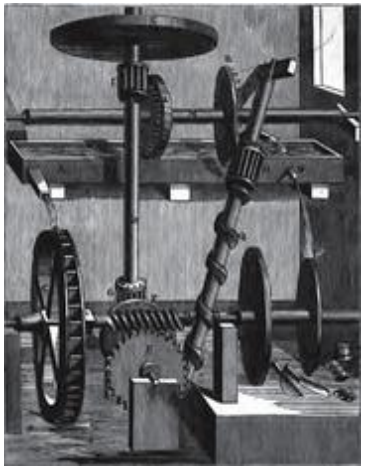


Slot makinesi

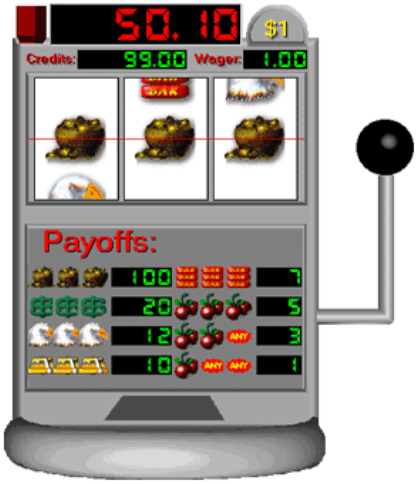




=



What is the difference?



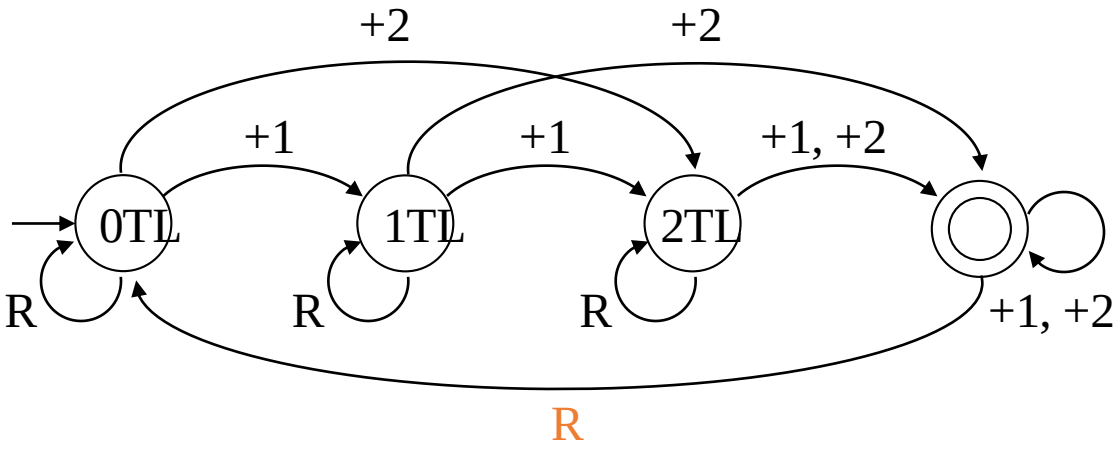
- Bu tür soruların akıl yürütmesi zordur, çünkü cihazlar sonsuz sayıda farklı şekilde tasarlanabilir.
- Otomata Teorisi bu soruların bazılarını cevaplamamızı sağlar.

Otomata ne yapabilir

- Küçük bir cihazın çalışmasını tanımlayabilirler
- (Basit) yazılımı doğrulamak için kullanılabilirler *
- Sözcük çözümleyicilerinde (lexical analyzer) programlama dillerindeki ifadeleri tanımak için kullanılırlar:

`ab1` is a legal name of a variable in Java
`5u=` is not

Farklı tür makineler



- Bu, bir makinenin yalnızca bir örneğiydi

- Farklı türdeki makinelere bakıp soracağız:
  - Bu tür bir makine ne tür sorunları çözebilir?
  - Bu tür bir makine için imkansız şeyler nelerdir?
  - Makine A, Makinesi B'den daha mı güçlü?

Bazı tür makineler

finite automata	Az miktarda belleğe sahip cihazlar. Çok basit şeyleri modellemek için kullanılır.
push-down automata	Sınırlı bir şekilde erişilebilen sonsuz belleğe sahip cihazlar. Gramer ayrıştırma (Parse Grammar) için kullanılır
Turing Machines	Sonsuz hafızalı cihazlar. Bunlar gerçek bilgisayarlar
time-bounded Turing Machines	Sonsuz bellek, ancak sınırlı çalışma süresi. Bunlar oldukça hızlı çalışan bilgisayarlardır.

Dersin bazı önemli noktaları

- **Sonlu Otomata (Finite automata)**
  - Otomatlar, metindeki kalıpları arama göreviyle yakından ilgilidir

```
find (ab)*(ab) in abracadabra
```

- **Gramer (Grammars)**
  - Gramer, konuşma dilindeki cümlelerin anlamını ve bilgisayar programlarının anlamını tanımlar.
  - Bir bilgisayar programından anlamın çıkarılmasını gramer kuralları sağlar (formal languages)

Dersin bazı önemli noktaları

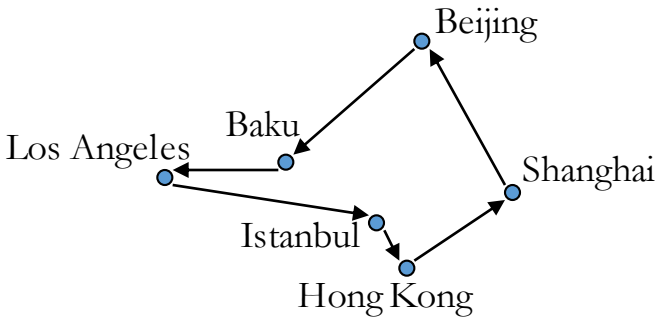
- Turing Machines
  - Bu, hesaplamayı umduğumuz her şeyi başaran genel bir bilgisayar modelidir.
  - Ancak bilgisayarların yapamayacağı birçok şey var:

The halting problem

Bir programlama dersinde özyinelemeyi ilk kez öğrenirken, kazara sonsuza kadar yinelenen bir kod yazmış ve kötü şöhretli bir "yığın taşması" hatası oluşturmuş olabilirsiniz. Bunun kodu çalıştırmadan gerçekleşeceğini bilmenin bir yolu olmasını dilemiş olabilirsiniz.

Dersin bazı önemli noktaları

- Time-bounded Turing Machines
  - Prensipte bir bilgisayarda birçok problem çözmek mümkündür, ancak pratikte çok fazla zaman alır
  - Gezgin Satıcı Problemi (TSP): Bir şehir listesi verildiğinde, onları bir defa ziyaret etmenin ve eve geri dönmenin en kısa yolunun bulunması.



  - Pratikte zor: 100 şehir için bu, en hızlı bilgisayarda bile 100+ yıl sürecektir!

Otomata Teorisinin Temeli

- Soruyu nasıl sorarız

A makinesi B problemini çözebilir mi?

- Öncelikle, çözmek istediğimiz sorunları tanımlamanın bir yoluna ihtiyacımız var

Problemler

- Dikkate alacağımız sorun örnekleri
- Bir s kelimesi verildiğinde, alt kelime olarak «be» içeriyor mu?
- Bir n sayısı verildiğinde, 7'ye bölünebilir mi?
- İki kelime s ve t verildiğinde, bunlar aynı mı?

- Bunların hepsinin "evet / hayır" yanıtları var.
- "Bunu bul" veya "Kaç tane" gibi başka tür sorunlar da vardır, ancak bunlara bakmayacağız.

- Ayrık matematik
- Belirli bir alana gerek yoktur, ancak ders doğası gereği matematikseldir.
- Tümevarım yoluyla fonksiyonlar ve ilişkiler, kümeler veya ispatlar hakkında giriş dersinde ele alacağız.

- Ayrık Matematik ve Veri Yapıları dersleri bu ders için yapılan çalışmaları desteklemektedir.

Ders Değerlendirme Kriterleri

- Vize : %40
- Final: %60

- Ödevler: extra credit

- Derse Katılım: öğrencinin sorumluluğunda

Referenslar

- Notes will be provided for every lecture at EDS (education support system).

The following references cover some of the topics in more detail.

Can be obtained from Library Genesis libgen.is

MAIN BOOK:

- **Elements of the Theory of Computation,** by **Lewis** and **Papadimitrou, second edition, 1998.**

SUPPLEMENTARY:

- *Introduction to the Theory of Computation*, second edition, by Michael **Sipser**
- *Introduction to Automata Theory, Languages, and Computation*, 3rd edition, by John **Hopcroft**, Rajeev Motwani, and Jeffrey Ullman
- Elements of Computation Theory, **Arindrama Singh**
- *Theory of Computing. A Gentle Introduction,* Efim Kinber & Carl Smith
- Video lectures by Shai Simonson.
- www.class-central.com Stanford University Automata Theory Course, Video lectures from the great Jeffrey Ullman
- GitHub - yurifw/Visual-Automata-Simulator by Jean Bovet
- **https://www2.cs.duke.edu/csed/jflap/ form Duke University**

Ders İçeriği

- **Introduction**
  - Presentation of the Course
  - Repetition of Data Structures and Basic Discrete Mathematics Concepts
  - Computational Problems and Solvability
- **Regular Languages and Finite Automata**
  - Deterministic Finite Automata (DFA – DSO)
  - Minimization of Finite Automata
  - Nondeterministic Finite Automata (NFA -NDSO)
  - Determinization of Automata
  - Regular Expressions and Languages (RE –RL)
  - Restrictions of Regular Languages: the Pumping Lemma (PL)
- **Context-free Languages and Pushdown Automata**
  - Context-free Grammars and Languages
  - The Parsing Problem for Grammars
  - Pushdown Automata (PDA)
  - Restrictions of Context-free Languages
- **Introduction to Theory of Computation**
  - Unrestricted grammars
  - Turing Machines (TM)
  - Decidability