# Algoritmusok

Van olyan probléma, amit nem tudunk algoritmussal megoldani. Ezért van szükség az algoritmus definiálására.

Turing gép: Minden más gépmodellről kiderült, hogy pont annyit tud, mint a Turing gép.

## Számítási eljárás

Kiindulunk egy halmazból: Állapotok halmaza

Ennek része a bementi állapotok halmaza

 kimeneti állapotok halmaza

 átmeneti függvény

…

 végtelen sorozat

pl.: Számítógép állapotsorozatai. (Minden órajel után)

Kimeneti állapotokat az tovább nem változtatja, a sorozat állandó lesz.

Véget ért a számítás, az eredmény.

Előfordulhat, hogy a számítás nem ér véget. Kimeneti állapotok halmazába sosem ér be.

Ez a fogalom túl általános.

## Példa: Euklideszi algoritmus

1 db egész szám a kimenet, az LNKO

2 db egész szám a bemenet

 az, hogy melyik sorában vagyunk épp az algoritmusnak.

## Szimulálás

Számítási eljárás mikor szimulál számítási eljárást

* bemeneteket át lehet kódolni
* állapotdekódolás: állapotokat vissza lehet kódolni
* lépésszám állapotból másikba eljutáshoz hányat kell lépni, vagy valami ilyesmi

Természetes dolgok teljesülnek.

Minden számítási eljárás szimulálja saját magát.

Tranzitivitás: c=c' és c'=c'' akkor c=c''

### Példa: Bővített euklideszi algoritmus

kimenet változik, 3 számot ad vissza 1 helyett.

állapotok halmaza is bővül: közbenső állapotok: 7 db egész szám + 1, ami megmutatja, melyik lépésben vagyunk: 2 vagy 3.1/3.2-ben

Ez szimulálni fogja az euklideszi algoritmust. elhagyjuk az x-et és y-t, akkor az eredmény ugyan az. lépések is dekódolhatóak, 2=2, 3=3.1 után 3.2

Visszafele nem igaz, euklideszi algoritmusból nem jön x és y, nem is tudom kiszámolni lnko-ból közvetlenül

### Két számítási eljárás egyforma erős

Ha egyik tudja szimulálni a másikat és a másik az egyiket.

## Szimulációk sebességének összehasonlítása

Valós idejű, ha minden lépés egy lépéssel van szimulálva.

Korlátos, egy lépés szimulálására max 10 lépést tesz meg

## Ordó (rend)

 függvény egy számsorozat.

 olyan számsorozatokat jelöl, amelyekre: sorozatra , hogy minden -re

Első néhány tagját megváltoztathatom, arra nem érzékeny . Eleje nem érdekes.

 sorozat nem nő gyorsabban, mint sorozat, konstanstól eltekintve.

 alsó becslés a kettő között. A másik felső volt.

A kettő metszete: (Theta)

Polinomiális függvény:

## Algoritmus

Számítógépen szimulálható dolgok érdekelnek bennünket.

Például maradékos osztás valós számok között nem megy. Két valós szám összehasonlítása sem megy.

Turing gépen szimulálható

## Turing gép

 szalagból és egy vezérlőegységből áll.

A szalagok mindkét irányban végtelen sok mezőre vannak osztva.

Véges sok mező kivételével minden mezőn üres jel áll. Ezt az egyszerűség kedvéért nem is jelöljük.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| … |  | a | I |  | 0 | 1 | … |

Vezérlőegység belső állapota

Szalagábécé:

Belső állapotok:

Minden szalagon be van jelölve, hol áll épp a feje.

Kezdőállapot: START s

Végállapot: HALT h

1 ütem: Vezérlőegység beolvassa a fej alatt álló jeleket. Mindenhova visszaír valamit, lehet ugyan azt. Elmozdul a szalagon 1-et balra, vagy 1-et jobbra, vagy semennyit. És állapotot vált

Ez akkor ér véget, mikro elér h állapotba, ekkor ami a szalagokon áll, az a számítás eredménye.

### Matematikailag

: belső állapotok

: szalagállapotok

: balra/jobbra jelek

Extra jelölések: hány szalag van, mi az üres jel, startállapot, halt állapot.

Halt állapotban nem csinál semmit:

## Turing gép, mint számítási eljárás: bemenet és kimenet

Turing gép egy aktuális állapota

 szó

Bementi állapot:

Kimeneti állapot:

Nem tudunk sok mindent csinálni, ha nem tudjuk a szalagok hosszát. Ezért szóközt nem tartalmazhatnak a bemenetek.

Kimenet mi lesz: Turing gép elér a halt állapotba, fejtől balra van a kimenet az első szóközig.

Fejtől jobbra levő dolgok neve "szemét". Ettől meg kell szabadulni, hogy összekapcsolhassunk több gépet akadály nélkül.

Ha bemeneti szó van, akkor az első m szalagra vannak felírva. Kimenetek pedig az utolsó m-re.

Kimenet hossza: Kimeneti szavak hosszának összege.

Unáris, bináris, ternáris, stb Turing gép: szalagábécének hány betűje van.

## Megjegyzés

Sok különböző gép van, lényegében ekvivalens különbségekkel. Pl.: melyik szélről indul a fej, stb

Ha egyik gép többet tud a másiknál, akkor triviálisan tudja szimulálni a másikat.

Vajon ha kevesebbet tud, akkor is tudja szimulálni a többet tudót? Igen. Mindjárt megnézzük.

## Példa: 3 szalagos Turing gép, összeadni fog

Jelkészlete:

Egyik szalagon egyik szám, másikon másik szám, harmadikon lesz az összegük.

4 állapot: s=start, c=carry, h=halt, b=vissza

(vissza: Mikor végzett, vissza kell mennie a végére, hogy a fejtől balra legyen a megoldás.)

## Turing gép szimulálása csökkentett jelkészlettel

Legalább kételemű ábécével kell rendelkeznie.

n-esekkel kódoljuk az eredeti ábécé elemeit.

Kódolás:

Belső állapotokat elszaporítjuk. Ezek segítségével "emlékezünk" dolgokra.

 és állapotok megjegyzik, hogy mit olvastunk.

…

Ez 4 lépés

## Egyszalagos Turing gépen többszalagosat tudunk szimulálni

 db fej helyett csak fejünk lesz.

…

### Példa

Gyökvonás egyszalagos gépen.

Ha 27-ből kell gyököt vonni:

Eredmény 5:

Könyvben nézzük meg, ha érdekel. Engem érdekelne, de nincs könyvem. Na, jó, annyira nem is érdekel.

## Szemétgyűjtés

Szalagon hagyott szemetet letöröljük.

Azonos szalagszámú géppel szimuláljuk, úgy, hogy ne hagyjon szemetet. Tudunk ilyen gépet konstruálni.

Feltétel: Ez csak akkor működik, ha az eredeti gépnek legalább egyszer végig kell olvasni az egész szalagot.

Ha tartalmaz extra jelet, amit nem, akkor be tudja festeni a szalagot: helyett -ot ír. De nem tartalmaz. Megoldás? Mindenből kétbetűset csinálunk:

a→space a

b→space b

space→space space

Ekkor a szükséges \*→nemspace space lesz.

Betűk széthúzása: Eltolom 1-gyel az egészet, visszajövök, megint.

Végén meg összenyomjuk hasonló módon a kimenetet.

## Univerzális Turing gépek

Eddigi szépséghiba: Minden problémára új Turing gépet konstruáltunk. Van-e programozható Turing gép?

Többet tud, mint egy számítógép, mert a szalagok végtelen hosszúak.

-edik szalagra írom a programot.

(Fél órás magyarázatot átaludtuk.)

## Egyszalagos univerzális Turing gép

Igen, van. 4 fejjel,

Akár 6elemű ábécével és 6 belső állapottal.

## Félszalagos Turing gépek

Csak egy irányban végtelenek a szalagok.

Ha lelép a szalagról, akkor "elszállt".

Félszalagoson lehet egész szalagost szimulálni.

## Post gép

Pont annyit tud, az két tétel: Turingon Postot, és Poston Turingot tudunk szimulálni..

## Korlátozott gépmodellek

Ha például nem lehet írni/olvasni egy szalagon.

## RAM gép

Utasításkészlete: CLR, INC, DEC, GET, PUT, ADD, LOAD, STORE, JUMP

3 regiszter: Akkumulátor , Indexregiszter , Segédregiszter

Ez is annyit tud, mint egy Turing gép.

Akár párhuzamos számításokat is tud szimulálni, ha milliárd számjegyes számokkal dolgozunk.

## Tárolt programú gép

Csak Akkumulátor regisztere van, de más az utasításkészlet.

Maga a program is a memóriában van. Ez jobban hasonlít a mai számítógépekhez.

Program fölül tudja írni saját magát.

## Markov automata

Ez is ekvivalens a Turing géppel.

## Sejtautomata

Négyzetrácson véges automaták, szomszédjaikkal tudnak kommunikálni.

Ez olyan, mint Convay's Game of Life

Lehet olyat írni, ami megduplázza saját magát. A természet is így dolgozik (DNS)

## Church féle λ-kalkulus

## Kiszámíthatóság: Kiszámítható függvények

Algoritmussal megoldhatatlan probléma:

Turing gép programját és bemenetét beolvasva meg tudjuk mondani, hogy megáll-e.

Sőt, még azt sem tudjuk, hogy programot beolvasva üres bemenettel megáll-e

## Vizsga

Keddenként. Index vagy fényképes igazolvány legyen. Inkább index.

Beugró kérdések.

Tételsor: 12 tétel, előadásonként 1-1.

Végleges változat a hétvégén még át lesz igazítva a tétellista (amit kihagytunk, ki lesz véve)

Tanár úr honlapja->Oktatás->DiMat2->

Ha más szóbeliztet, más lesz a tételsor.

## Konzultáció

E-milben néhány nappal előtte egyeztessünk.