X. Előadás (2020. Ápr. 30.)

(A) A Turing gép működése

Tekintsük az előző előadásban példaként megadott Turing gépet és vizsgáljuk meg néhány lépését.

$Q = \{q_0, q_1, q_2, q_3, \mathbf{q}_4\}$, $F = \{\mathbf{q}_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, X, Y, B\}$,

δ	0	1	X	Y	B
q_0	(q_1, X, R)	–	–	(q_3, Y, R)	–
q_1	$(q_1, 0, R)$	(q_2, Y, L)	–	(q_1, Y, R)	–
q_2	$(q_2, 0, L)$	–	(q_0, X, R)	(q_2, Y, L)	–
q_3	–	–	–	(q_3, Y, R)	$(\mathbf{q}_4, B, R)$
$\mathbf{q}_4$	–	–	–	–	–

Kezdetben a fej q_0 állapotban van és tegyük fel hogy a szallagon a 01 szó van, a fej pedig a 0-t olvassa be. Akkor a fej q_1 állapotba kerül, 0 helyére X-et ír és egyet jobbra lép a szallagon, így az 1 jelhez kerül (lásd 1 Ábra (a), (b)) Most az 1-est beolvasva q_2 állapotba kerül, az 1-helyére Y-ont ír és egyet balra lép a szallagon - ezáltal az X fölé kerül (lásd 1. Ábra (c)).

Hogy a Turing gép működését áttekinthetőbbé tegyük, ún. *folyamatos leírást* fogunk alkalmazni: Ez azt jelenti hogy az input szallag teljes tartalmát ábrázoljuk és persze a fej aktuális állapotát is. Tegyük fel, hogy az input szallag a $w = x_1x_2...x_{i-1}x_i...x_n$ szót tartalmazza, a fej pedig q állapotban az x_i jel fölött áll (Lásd a 2. Ábrát). Tegyük fel hogy δ-t megadó. táblázatban ezt látjuk:

δ	...	x_i	...
		$\vdots$	
q	...	(p, Y, L)	...

Akkor az ott megadott tranzíció teljes hatását így rögzítjük:

$x_1x_2...x_{i-1}qx_i...x_n \vdash x_1x_2...x_{i-2}px_{i-1}Yx_{i+1}...x_n$

A fej ugyanis balra mozdul egyet és p állapotba kerül, miközben x_i helyére $Y \in \Gamma$ szimbólumot ír. Használni fogjuk majd a $\overset{*}{\vdash}$ jelölést aminek a jelentése itt is: 0 vagy több (de véges) számú $\vdash$ lépés után.

Elfogadott nyelv. Azt mondjuk hogy az $\mathcal{M}$ Turing gép *elfogadja (akceptálja)* a $w \in \Sigma^*$ szót, ha w-vel a szalagján egy q_0 kezdőállapotból indulva egy $q_f \in F$ végállapotba kerül és ott megáll, miközben a fej is egy „jobbszélső cellába" kerül.
-„jobbszélső cella": olyan cellát jelent az input szallagon amitől jóbbra már csak blank (B) (azaz üres cella) van.
Folyamatos leírást használva az elfogadott szavak összeségét vagyis az $\mathcal{M}$ Turing $\mathcal{L}(\mathcal{M})$ nyelvét így adhatjuk meg:

$$\mathcal{L}(\mathcal{M}) := \{w \in \Sigma^* \mid q_0w \overset{*}{\vdash} \alpha q_f B...B, \text{ és } \mathcal{M} \text{ megáll}\}$$

Pl. az előző példában megadott Turing gép nyelve: $\mathcal{L}(\mathcal{M}) = \{0^n 1^n \mid n \geq 1\}$. Fontos megjegyezni, hogy a TM gép amikor beolvas egy akceptált szót meg kell álljon!
- Ha a Turing gép beolvas egy olyan w szót amit nem akceptál akkor vagy egy nem-elfogadó állapotban (helyzetben) áll meg, vagy végtelen ciklusba kerül (azaz nem áll meg egyáltalán, bizonyos lépések ciklikusan ismétlődnek)

Példa 1. Megmutatjuk hogy fenti példában szereplő Turing gép elfogadja a $w = 0011$ szót. Valóban:

$q_0 0011 \vdash X q_1 011 \vdash X 0 q_1 11 \vdash X q_2 0 Y 1 \vdash q_2 X 0 Y 1 \vdash X q_0 0 Y 1 \vdash X X q_1 Y 1 \vdash$
$\vdash X X Y q_1 1 \vdash X X q_2 Y Y \vdash X q_2 X Y Y \vdash X X q_0 Y Y \vdash X X Y q_3 Y \vdash X X Y Y q_3 \vdash$
$\vdash X X Y Y B q_4 \vdash$ STOP.

A fej végállapotba (q_4) és egy jobbszélő cellába kerül (elfogadó állapot) és ott megáll - tehát a TM gép a szót elfogadja.

(B) A Turing gépek által elfogadott nyelvek

A Turing gépek által elfogadott nyelveket *rekurzíven felsorolható nyelvekn*ek nevezzük. A rekurzivan felsorolható nyelvek között vannak olyan $L \subseteq \Sigma^*$ nyelvek is, ahol egy $w \in \Sigma^*$ szónak az L nyelvhez tartozását nem lehet egy algoritmussal (véges lépésben) eldönteni! -ez a nyelvosztály a legáltalánosabb nyelvosztály amit vizsgálni szoktak. Természetesen, az előzöleg megismert nyelvosztály, a környezetfüggetlen nyelvek osztálya egy valódi részhalmazát képezi a rekurzívan felsorolható nyelvek osztályának. Ez utóbbinak az ún. rekurzív halmazok fogalmához is kapcsolódnak.

Rekurzív halmazok. Egy $A \subseteq U$ részhalmazt *rekurzív halmaznak* nevezünk, ha az hogy egy $a \in U$ elem az A halmazhoz tartozik mindig eldönthető egy algoritmussal (véges lépésben). Azokat az $L \subseteq \Sigma^*$ nyelveket ahol egy $w \in \Sigma^*$ szónak az L nyelvhez tartozása eldönthetőt egy algoritmussal *rekurzív nyelveknek* nevezzük. A rekurzív nyelvek tehát nem mások mint rekurzív szóhalmazok, pontosabban, Σ^* rekurzív részhalmazai.

Példák (1) A prímszámok Π halmaza az egész számok $\mathbb{Z}$ halmazának egy rekurzív részhalmazát alkotják, mert az Euklidészi algiritmussal bármely egész számról véges lépésben eldönthető hogy prímszám-e vagy sem.
(2) A racionális számok $\mathbb{Q}$ halmaza a valós számok $\mathbb{R}$ halmazának **nem** rekurzív részhalmaza mert véges lépésben egy $r \in R$ valós számnak csak véges sok számjegye ismerhető meg és ez alapján nem eldönthető hogy r racionális szám-e vagy sem.
(3) Ha egy M turing gép olyan hogy minden $w \in \Sigma^*$ szóra véges lépésben megáll, akkor az $\mathcal{L}(\mathcal{M})$ nyelve rekurzív nyelv - az algoritmus ami egy $w \in \Sigma^*$ szónak $\mathcal{L}(\mathcal{M})$-hez tartozását eldönti az maga a TM gép beolvasási folyamata, amely

végén $\mathcal{M}$ vagy akceptáló vagy nem akceptáló állapotban áll meg - ami alapján w $\mathcal{L}(\mathcal{M})$-hez tartozása eldönthető.
(4) A tanultak alapján a környezetfüggetlen nyelvek ugyancsak rekurzív nyelvek (w akkor tartozik az L környezetfüggetlen nyelvhez, ha az L-t megadó G környezetfüggetlen nyelvtan szabályai szerint elvégezhető az $S \Longrightarrow w$ deriváció. Egy ilyen deriváció mindig véges lépésből áll -így ez jelenti a nyelvhez tartozást eldöntő algoritmust.)

Az előző előadásban már említettük hogy a Turing gépek lényegében az algoritmus fogalmának a modellezésére szolgálnak. Ha elfogadjuk azt az „axiomát" hogy minden algoritmus származtatható egy Turing géppel, akkor azt kapjuk hogy a rekurzív nyelvek a rekurzívan felsorolható nyelvek egy valódi osztályát alkotják.

Magyarázat: Az axióma alapján minden rekurzív nyelv előállítható egy TM géppel (mégpedig egy olyan géppel ami minden szóra megáll). Mivel a TM gépek által előállított nyelvek összesége nem más mint a rekurzívan felsorolható nyelvek osztálya, a fenti tartalmazás nyilvánvaló. Később igazolni fogjuk hogy valóban létezik olyan rekurzívan felsorolható nyelv ami nem rekurzív.

(C) Turing gépek mint „egész" függvényeket kiszámító berendezések

Megmutatjuk hogy a Turing gépeket felfoghatjuk úgy is mint „egész" függvényeket kiszámító berendezéseket:
Egész függvények alatt a számítástechnikában $f\colon \mathbb{N}^k \longrightarrow \mathbb{N}$ alakú parciális függvényeket értünk. (az egész függvény itt egy hagyományos elnevezés). Például $f(n,m) = \text{ln.k.o}(n,m)$ egy $f\colon \mathbb{N}^2 \longrightarrow \mathbb{N}$ alakú egész függvény.
Szükségünk lesz még a temészetes számok ún. *unáris reprezentáció*jának a fogalmára:
- az $i \in \mathbb{N}$ szám unáris reprezentációja a 0^i szó, konkrétan:
  3 ún. reprezentációja: $0^3 = 000$
  0 ún. reprezentációja: $0^0 = \varepsilon$
- ha f k-változós, akkor az $x_1, x_2, ..., x_k$ változóknak megfelelő értékeket 1-el választjuk el. Tehát $w = 0^{i_1}1\ 0^{i_2}1...\ 0^{i_k}$ szó jelöli az $(i_1, i_2, ..., i_k) \in \mathbb{N}^k$ vektort. például 00010100 nem más mint a $(3,1,2) \in \mathbb{N}^3$ vektor unáris reprezentációja.

Tekintsük most egy olyan TM gépet ahol $\Sigma = \{0,1\}$, $\Gamma = \{0,1,B\}$. Ennek az input szallagjára felírjuk az $(i_1, i_2, ..., i_k)$ vektor unáris reprezentációját jelentő w szót, ami egy 0-ból és 1-ből álló sorozat. Ha a Turing gép ennek hatására nem kerül ciklusba, akkor attól függetlenül hogy elfogadja-e vagy sem a $w = 0^{i_1}1\ 0^{i_2}1...\ 0^{i_k}$ szót, véges lépés után megáll és a szallagra kiír egy 0-ákból és 1-sekből és B-kből (üres cellákból) álló sorozatot, tehát egy $w' \in (0+1+B)^*$ szót. Ebből a B-ket (vagyis az üres mezőket) elhagyva egy $f(w) \in (0+1)^*$ szót kapunk, amit az $n \in \mathbb{N}$ szám bináris reprezentációjának tekintünk. Pl.: $10BB11B001B \mapsto 1011001 \longrightarrow 89$ (89 a 2-es számrendszerben 1011001). Így

tehát egy a TM segítségével egy $f\colon \mathbb{N}^k \longrightarrow \mathbb{N}$ parciális függvényt állítunk elő (azért csak parciális függvény, mert azokra a $(i_1, i_2, ..., i_k)$ vektorokra amire a TM gép végtelen ciklusba kerül, nem kapunk eredményt.) Nyílvánvalóan, ha a TM gép olyan, hogy minden szó beolvasása után megáll, akkor egy valódi $f\colon \mathbb{N}^k \longrightarrow \mathbb{N}$ függvényt kapunk.

Sajnos az előző okfejtésben felhasznált TM gép elég sajátos, hiszen itt $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, B\}$. Az Ilyen TM gépet *standard TM gépnek* nevezzük. Igaz viszont az alábbi

Állítás 1. *Minden TM gép ekvivalens egy standard TM géppel.*

Ebből az állításból és az előző okfejtésből következik az alábbi eredmény:

Következmény 1. *Minden TM géphez és $k \in \mathbb{N}$ -számhoz hozzárendelhetünk egy $f\colon \mathbb{N}^k \longrightarrow \mathbb{N}$ parciális függvényt.*

Ezt még kiegészithetjük azzal hogy:

- Ha vannak olyan szavak amire a TM gép nem áll meg, akkor ez a függvény valóban csak $\mathbb{N}^k$ egy (nemüres) részhalmazán értelmezett. Ekkor a TM gép nyelve "csak" rekurzívan felsorolható, a kapott $f\colon \mathbb{N}^k \longrightarrow \mathbb{N}$ függvényt pedig parciálisan rekurzívnak nevezzük.
- A kapott függvény pontosan akkor valódi függvény -azaz akkor mindenütt értelmezett, ha a TM gép nem kerülehet ciklusba, ami azt jelenti hogy $\mathcal{L}(\mathcal{M})$ nyelve rekurzív. Ekkor az f függvényt is rekurzív függvénynek nevezzük.

Példák 2.: $f(n) = n!$, $f(n) = [\log_2 n]$, $f(n, m) = (n + m, n - m)$ egy, illetve kétváltozós rekurzív egész függvények.

(D) Church tézis

Az előbbi eredményt úgy is megfogalmazhatjuk, hogy a rekurzív és parciálisan rekurzív egész függvények nem mások mint a Turing gépek által előállított egész függvények.

Kiszámítható függvény fogalma. Egy $f\colon \mathbb{N}^k \longrightarrow \mathbb{N}$ egész függvényt *kiszámíthatónak* nevezünk, ha minden $f(i_1, i_2, ..., i_k)$ értéke -amennyiben itt a függvény értelmezett- véges lépésben kiszámítható, csak aritmetikai és logikai műveleteket használva.

Tézis (Church): *A kiszámítható függvények osztálya megeggyezik a parciálisan rekurzív függvények osztályával.*

Más megfogalmazásban: *Minden (parciális) kiszámítható függvény előállítható egy Turing géppel.*

Church tézise csupán egy másfajta megfogalmazása annak a posztulátumnak (axiomának), hogy minden algoritmus származtatható egy Turing géppel, hiszen azt jelenti, hogy mindazok a parciális függvények amelyek előállíthatók egy algoritmussal, azok előállíthatók egy Turing géppel is. Church tézisének egy másik ekvivalens formája az alábbi:

Állítás 2. *Minden TM gép ekvivalens egy „RAM"-al.*

RAM:= „Random Acces Machine" - valójában egy több veremmel rendelkező idealizált regiszter gép aminek a vermei (memoria rekeszei) korlátlan kapacitásúak -ezért nem ekvivalens egy veremautomatával (aminek a verme véges)

Ezek az állítások (tézisek) valójában mind azt fejezik ki, hogy minden eljárás (algoritmus, procedura) szimulálható egy Turing géppel. A Church tézis nem bizonyítható, de széles körben elfogadott. A bizonyításnak elvi akadálya van, nevezetesen az hogy az algoritmus vagy eljárás fogalmának nincs formális matematikai definíciója. A fenti állítások egyik fontos következménye az alábbi:

Következmény 2. *Minden szubrutin végrehajtható egy megfelelő Turing géppel.*

Gyakorlaton majd megismerkedünk egy Turing géppel ami azt a szubrutint hajtja végre aminek a neve *valódi kivonás*. Ha $m.n \in \mathbb{N}$, akkor ezek valódi különbségét így definiálhatjuk:

$$m \div n = \begin{cases} m-n & \text{ha } m \geq n \\ 0, & \text{különben} \end{cases}$$

Például $5 \div 3 = 2$, de $3 \div 5 = 0$.

(D) Eldönthetőségi problémák

1) Először David Hilbert vetette fel azt a kérdést hogy hogy létezik-e olyan $P(n), n \in \mathbb{N}$ (a természetes számok halmazán értelmezett) predikátum hogy sem $P(n)$, sem $\neg P(n)$ igaz voltát nem lehet a természetes számok Peano-féle axiomarendszeréből levezetni? K. Gödel osztrák matematikus 1932-ben konstruktív módon bizonyította hogy van ilyen $P(n)$ állítás. Felmerült a kérdés, hogy esetleg algoritmikusan nem lehet-e az ilyen állítások igaz vagy hamis voltát eldönteni. A. Turing, a róla elnevezett gép segítségével egy újabb bizonyítását adta Gödel válaszának és megmutatta hogy $P(n)$ igaz vagy hamis volta valamilyen algoritmus segítségével sem dönthető el.

2) A Nagy Fermat Sejtés ami több száz éven keresztül foglalkoztatta a matematikusokat, viszont nem eldönthetetlen kérdés. Igaz voltát 2002-ban be is bizonyították. A sejtés az, hogy nem léteznek olyan $x, y, z \in \mathbb{Z}$ és $n \in \mathbb{N}$, $n \geq 3$ számok, hogy az

$$x^n + y^n = z^n$$

egyenlőség teljesüljön.

Más eldönthetelen problámák: 3) A sík lefedése bizonyos szabálytalan alakzatokkal (például romboid és dárda: lásd 3. Ábra) eldönthetelen probléma (Gardner, 1967)
4) Kaotikus mozgásokra vonatkozó problémák: például az időjárás alakulása egy hétnél hosszabb időszakra eldönthetetlen. Kaotikus az a mozgás, ahol a kezdeti feltételek nagyon kicsiny változása a végeredmény nagyfokú változását vonhatja maga után - ilyen az időjárás is.
5) Kolmogorov orosz matematikus igazolta hogy a naprendszer stabilitása egy adott t_0 időpontban, legalábbis a bolygók mozgását leíró mechanikai differenciálegyenletek alapján eldönthetetlen. Az egyenletek alapján ugyanis a stabilitási és instabilitási pontok sűrün helyezkednek el a fázistérben, egy stabilitási pont tetszőlegesen szűk környezetében van egy instabilitási pont is - ez egyébként sajátossága az ún. *három test problémának*, amikor három tömeg gravitációs erővel hat egymásra. A Kolmogorov féle példában több test van: a Nap és a bolygók: ezek mind hatnak a Föld mozgására.
6) Látni fogjuk hogy a Formális Nyelvek tárgy több eldönthetetlen problémát is tartalmaz.

A témakörre vonatkozó pontos definíciókat a következő előadásban vezetjük majd be.