Információs rendszerek elméleti alapjai

Információelmélet

Kolmogorov bonyolultság – kiszámítható függvények

Bevezetés:

A Shannon entrópia szerepe:

- Ami majd a jövőben bekövetkezhet (üzenet) azt hogyan tudjuk a legrövidebbben, várható értékben leírni.
- Ez a jövőre való felkészülés – mi várható, és ami várható, azt különböztessük meg, átlagosan a legrövidebbben.

Kolmogorov bonyolultság – kiszámítható függvények

<u>A fordított feladat</u>: ami a múltban bekövetkezett, azt hogyan tudjuk a legrövidebben jellemezni?

A múlt egy hosszú adatsorként jelenjen meg. Hogyan lehet a legtömörebben leírni?

A leírásból algoritmus (számítógép) állítsa vissza!

Kolomogorov kérdése: Hogyan lehet a „Háború és Békét” legrövidebben kódolni, hogy abból számítógép visszaállítsa?

•1960-as évek az algoritmikus információelmélet kezdetei.

•Kolmogorov 1964.

- Múlt tömör leírása
- Kódhosszat minimalizálni
- Egy nagyon hosszú kódot, hogyan tudok tömörebben előállítani, és hogyan állítható ebből az információ vissza

Kolmogorov bonyolultság – kiszámítható függvények

- A kérdést alapvetően a véletlen-szám generátorok jóságának ellenőrzésére, mérése vetette fel. Mennyire jó egy véletlen-szám tábla? Nem jó, ha lehet tömöríteni, amiből újra kiszámítható.

Kolmogorov bonyolultság – kiszámítható függvények

Mit jelent a tömörítés. Az *x*-t tömöríteni, *x* tömörítése (kódja) *p* egy *f* függvény szerint, ha *f(p)*=*x*.

x alakja	$f(p)$	x – kód p hossza $l(p)$	x hossza (bináris)
$x \in N$	$f_0(p) = p$	$l(p) = l(x)$	$l(p) = l(x) = log_2 x$
$x = 2^k$	$f_1(p) = 2^p$	$l(p) = log\, log\, x$	$l(x) = 2^{l(p)}$
$x = 2^{2^k}$	$f_2(p) = 2^{2^p}$	$l(p) = log\, log\, log\, x$	$l(x) = 2^{2^{l(p)}}$
$x = 2^{\overbrace{2^{\cdot^{\cdot^{2}}}}^{k}}$	$f_3(p) = 2^{\overbrace{2^{\cdot^{\cdot^{2}}}}^{p}}$	$l(p) = \underbrace{log\, log \ldots log\, x}_{p}$	

Kolmogorov bonyolultság – kiszámítható függvények

Ezt a négy függvényt használva *x* kódolásához, nézhetjük, melyik szerint (ha lehet) legrövidebb a kód. Ezzel a módszerrel (*i,p*) pár használható, tudni kell, hogy melyik a dekódoló függvény.

i= melyik függvényt használom (index)

p=kód (tömörítetten a legrövidebb)

Hogyan kódolnánk az 1025-öt (=$x= 1024+1=2^{10}+1$)?

Például $f(p)=f_1(p)+1$

Ez jobb mint az $f_0,\ldots, f_n$, közül egyedül használható f_0 szerinti kód.

Általában véges sok dekódolót választva, a fenti módszerrel tudunk egy továbbit választani. Ami speciális számokra tömörebb kódot használ.

Kolmogorov bonyolultság – kiszámítható függvények

Milyen függvények választhatók a kódoláshoz?
Kiszámítható függvények.

- Rekurzívan felsorolhatók – ezért az (*i*,*p*) pár segítségével konstruált módszer működik.
- A pontos definícióhoz, tulajdonságokhoz, használatához, és a tulajdonságainak vizsgálatához szükség van a kiszámítható függvények fogalmára, és világára
- Az előbbi példában: *f* – kiszámítható, és ezek a függvények megszámlálhatóan sokan vannak, és
- az *f*-ek halmaza rekurzívan felsorolható

Kiszámítható függvények

Parciális rekurzív függvények kiszámíthatók

- Felépítésük: intuitívan kiszámítható alapfüggvényekből, kiszámítható módon újabb függvényeket képezünk
- Kiinduló, vagy kezdeti függvények (initial)

Kiszámítható függvények

$f : N^k \to N, N = \{0,1,\dots,\ \} = N_0$

Alap függvények :

– zéró függvény $Z : N \to N, Z(x) = 0;$

- rákövetkező: $S : N \to N, S(x) = x + 1;$

$\sigma : N \to N, \sigma(x) = x + 1$ alternatív jelölés

- vetítő (projekció): $P_{n,i} : N^n \to N, P(x_1, \dots, x_i, \dots, x_n) = x_i$

$\pi_i^n : N^n \to N, \pi(x_1, \dots, x_i, \dots, x_n) = x_i$

Definíció : f parciális függvény $f : N^k \to N,$ ha

$f : X \to N, X \subset N^k,$ f kiszámítható (computable)

ha létezik olyan utasítás sorozat, amelyik

$f(x_1, \dots, x_i, \dots, x_n)$ eredménnyel befejeződik (terminálódik)

ha ez az érték definiált,

egyébként pedig nem fejeződik be

ha $X = N^k$, akkor f teljes függvény (total function)

Függvény képzések

Függvénykompozíció:

$f : N^n \to N, N = \{0,1,\ldots,\ \} = N_0$ a

$h : N^k \to N,$ és

$g_1, g_2, \ldots, g_k : N^n \to N$

függvényekből kompozícióval keletkezik,

$f(x_1,\ldots,x_n) = h(g_1(x_1,\ldots,x_n), g_2(x_1,\ldots,x_n), \ldots, g_k(x_1,\ldots,x_n))$

Kiszámítás menete egyszerűen megadható

Függvény képzések

Primitívrekurzió:

$f : N^{n+1} \to N, N = \{0,1,\ldots,\ \} = N_0$ a

$g : N^{n} \to N,$ és

$h : N^{n+2} \to N$

függvényekből primitívrekurzióval keletkezik,

$f(x_1,\ldots,x_n,0) = g(x_1,\ldots,x_n)$ kezdőérték;

$f(x_1,\ldots,x_n,y+1) = h(x_1,\ldots,x_n,y,f(x_1,\ldots,x_n,y))$

Primitív rekurzív függvények kiszámíthatók (computable)

Megj.: Terminológia: Az újabb szakirodalomban egyes szerzők kiszámítható függvényekről beszélnek rekurzív függvények helyett.

Ha g és h kiszámítható akkor f is.

$\bar{x} = (x_1, \dots, x_n)$ adott,

$g(\bar{x})$ valamilyen eljárással kiszámítható

Ha terminál/befejeződik, akkor az

$f(\bar{x}, 0)$ értéket megkaptuk.

Ezt felhasználva, valamint h kiszámítására

szolgáló eljárást alkalmazva kapjuk $f(\bar{x}, 1)$.

Ha ez az eljárás is befejeződik

akkor az $f(\bar{x}, 1)$ kiszámított értékét felhasználva

valamint h kiszámítására szolgáló eljárást alkalmazva.

kapjuk $f(\bar{x}, 2)$, *stb.*

Függvény képzések

Zártság a primitív rekurzióra nézve:

- Primitív rekurzióra zárt függvény osztály *C*, ha *Z, S,* $P_{n,i}$ benne van, és a kompozíció és a primitív rekurzió nem vezet ki belőle.

Primitív rekurziv függvények generáló sorozattal megadhatók :

$g_1, g_2, \ldots, g_n = f$ az f generáló sorozata, ha

vagy - g_i alapfüggvény

vagy - g_i a nála kisebb indexűekből kompozícióval keletkezik

vagy - g_i a nála kisebb indexűekből primitív rekurzióval keletkezik

Primitív rekurzív függvények

Tétel Primitív rekurzív függvények osztálya a primitív rekurzív függvényosztályok metszete (Két zárt osztály metszete is zárt, teljes primitív rekurzív függvények osztályainak metszete, ez a primitív rekurzióra zárt legkisebb függvény osztály).

Bizonyítás:

a) A primitív rekurzív függvények osztálya zárt osztály
   - Alapfüggvény nyilván saját magának a generálója
   - A generáló sorozatok „átindexelő” konkatenálása a kompozíció,
   - illetve a primitív rekurzió (a nála kisebb indexűekből) szerinti generáló sorozat lesz.

b) Ha C zárt , akkor a $\forall g_1, g_2, \ldots, g_n = f$ Benne van,

c) n szerinti indukcióval egyszerűen következik.

Megjegyzés: a primitív rekurzió helyett az iterációt is választhatjuk

Nem minden kiszámítható függvény primitív rekurzív.

$F : N^2 \to N$ az $f : N \to N$ – ből iterációval keletkezik, ha

$F(x,0) = f(x)$

$F(x, y+1) = f(F(x, y))$.

Universzális függvény primitív rekurzív

Tétel Létezik kiszámítható univerzális függvény
:

$U : N^2 \to N$ úgy, hogy bármely

az $f : N \to N -$ primitív rekurzív függvényhez létezik k,

hogy

$U(x,k) = f(x)$ és

$U(x,k) \quad \forall k$ - ra primitív rekurzív.

Univerzális függvény

Bizonyítás vázlata:

A generáló sorozatokat (véges számú behelyettesítés, kompozíció, pr.rekurzió) fel tudjuk írni véges ABC feletti szó formájában. A generáló sorozatok grammatikája egyszerű, könnyű eljárást adni, hogy a szó generáló sorozatot jelent-e.

A szavak hossz-lexigrafikus sorrend szerinti felsorolásával megkapható az első $f_1 : N \rightarrow N$, generáló sorozat, majd a második, stb.

Ezek a sorozatok felfoghatók úgy mint valamilyen programok. (E programok sorban definiálják a különböző rekurzív funkciókat; mindegyik önálló programra specifikálnak más rekurzív funkciókat, amelyekre támaszkodva építik fel az újabbakat valamint az alkalmazandó műveleteket).

Univerzális függvény

Az összes ilyen program eldönthető és kiszámíthatóan felsorolható (beszámozható).

Az $U(x,k)$-t, úgy definiáljuk, hogy kikeressük a k-adik generáló sorozatot, majd használjuk a generálás által adott számítást x-re.

A $U(x,n) \rightarrow$ (*az* n-dik program által specifikált függvény alkalmazásának *az* eredménye *az* x számra) , függvény kiszámítható, a konstrukció és a definíciója révén.

Ez a függvény *univerzális* a primitív rekurzív függvényekre.

Univerzális függvény

Állítás: a $h(x)=U(x,x)+1$ nem lehet primitív rekurzív

Bizonyítás: Ha primitív rekurzív volna, akkor legyen a sorszáma k. $h(k)=$ ekkor $U(k.k)$ és $U(k.k)+1$ egyszerre igaz volna, ez ellentmondás!

$h(k)=U(k.k)$ teljes, kiszámítható

$d(k)=U(k.k)+1$, különbözik bármelyik primitív rekurzív függvénytől (a k-dik primitív rekurzív függvénytől a k pontban különbözik)

Primitív rekurzív függvények

Konstans fv.	$\mathrm{j}() = \overbrace{\sigma(\sigma(\ldots\sigma(0())))}^{j}$
Összeadás	$plus(n_1, 0) = \pi_1^1(n_1)$ $plus(n_1, n_2+1) = \sigma(\pi_3^3(n_1, n_2, plus(n_1, n_2)))$
Összeadás egyszerűbben	$plus(n_1, 0) = n_1$ $plus(n_1, n_2+1) = \sigma(plus(n_1, n_2))$

Primitív rekurzív függvények

Kiértékelés	$\begin{aligned} plus(7,4) &= plus(7,3+1) \\ &= \sigma(plus(7,3)) \\ &= \sigma(\sigma(plus(7,2))) \\ &= \sigma(\sigma(\sigma(plus(7,1)))) \\ &= \sigma(\sigma(\sigma(\sigma(plus(7,0))))) \\ &= \sigma(\sigma(\sigma(\sigma(7)))) \\ &= 11 \end{aligned}$
Szorzás	$\begin{aligned} n \times 0 &= 0 \\ n \times (m+1) &= n + (n \times m) \end{aligned}$
Hatvány	$\begin{aligned} n^0 &= 1 \\ n^{m+1} &= n \times n^m \end{aligned}$

Primitív rekurzív függvények

Hatvány torony (dupla)	$n \uparrow\uparrow 0 = 1$ $n \uparrow\uparrow m+1 = n^{n \uparrow\uparrow m}$ $\quad n \uparrow\uparrow m = n^{n^{\cdot^{\cdot^{\cdot^{n}}}}} \}m$
Hatvány torony (tripla)	$n \uparrow\uparrow\uparrow 0 = 1$ $n \uparrow\uparrow\uparrow m+1 = n \uparrow\uparrow (n \uparrow\uparrow\uparrow m)$
Hatvány torony (*k*)	$n \uparrow^k 0 = 1$ $n \uparrow^k m+1 = n \uparrow^{k-1} (n \uparrow^k m).$
Ha *k* egy argumentum	$f(k+1,n,m+1) = f(k,n,f(k+1,n,m)).$

Primitív rekurzív függvények

Faktoriális	$0! = 1$ $(n+1)! = (n+1).n!$
Megelőző	$pred(0) = 0$ $pred(m+1) = m$
Különbség	$n \dot{-} 0 = n$ $n \dot{-} (m+1) = pred(n \dot{-} m)$
Előjel (signum)	$sg(0) = 0$ $sg(m+1) = 1$

Rekurzív függvények

Érdemes rámutatni arra, hogy sokkal speciálisabb ok is van a mögött, hogy kiszámítható függvények alatt ne csak primitív rekurzív függvényeket értsünk.

Az egyik ilyen ok, hogy a primitív rekurzív függvények nem tudnak túl gyorsan nőni.

Ez a konstrukció Ackermann-tól , származik , az *Ackermann* függvények sokkal gyorsabban nőnek mint bármelyik primitív rekurzív függvény

Rekurzív függvények
Ackermann

$Ack: N^2 \rightarrow N$

$Ack(0, y)$	=	$y + 1$
$Ack(x + 1, 0)$	=	$Ack(x, 1)$
$Ack(x + 1, y + 1)$	=	$Ack(x, Ack(x + 1, y))$

$a_m: N \rightarrow N$		
$a_m(x)$	=	$Ack(m,x)$
$a_{m+1}(x)$	=	$Ack(m + 1, x)$
$a_{m+1}(x+1)$	=	$Ack(m+1, x + 1)=Ack(m, Ack(m + 1, x))$
$a_{m+1}(x+1)$	=	$a_m(Ack(m + 1, x))= a_m(a_{m+1}(x))= a_m(A_{m+1}(x))$; iteráció

Rekurzív függvények
Ackermann

a_m: $N \rightarrow N$	Függvények rendkívül felgyorsulnak
$a_3(x)$	exponenciális
$a_4(x)$	x nagyságú hatvány torony
$a_5(x)$	Olyan toronyhatvány, aminek a magassága x magasságú toronyhatvány
Ack(4,2)	19,729 decimalis számjegyből áll

$$A(1,0) = A(0,1) = 2$$
$$A(1,1) = A(0,A(1,0)) = A(0,2) = 3$$
$$A(1,2) = A(0,A(1,1)) = A(0,3) = 4$$
$$A(2,0) = A(1,1) = 3$$
$$A(2,1) = A(1,A(2,0)) = A(1,3) = A(0,A(1,2)) = A(0,4) = 5$$

Rekurzív függvények
Ackermann

f: $N^k \to N$ Függvény az r-Ackermann szinten belül van, ha $X=\max(x_1, \ldots, x_n)$, $f(x_1, \ldots, x_n) \leq A_r(X)$ (véges sok helytől eltekintve)

Állitás: Legyen *f* primitív rekurzív, akkor *f* valamilyen *r*-re, *r* *A* szinten belül van.

Bizonyítás vázlat:

A (primitiv rekurzív) függvényképzések csak véges szinttel növelik meg a függvény *A* szintjét.

Következmény: Az $f(x) = A_x(x)$ függvény nem primitív rekurzív

A (primitiv rekurzív) függvényképzések csak véges szinttel növelik meg a függvény *A* szintjét.

$$f(x) = g(h_1(x), \ldots, h_k(x))$$

$$\alpha_N(\max(h_1(x), \ldots, h_k(x))) \le \alpha_N(\alpha_N(x)) \le \alpha_{N+1}(x)$$

$$f(x,0) = g(x);$$
$$f(x, n+1) = h(x, n, f(x,n)),$$

$$\begin{aligned} f(x,1) = h(x,0,f(x,0)) &\le \alpha_N(\max(x,0,f(x,0))) \\ &\le \alpha_N(\max(x,0,\alpha_N(\max(x)))) \le \alpha_N(\alpha_N(\max(x))) \end{aligned}$$

$$f(x,i) \le \alpha_N^{[i+1]}(\max(x)) \le \alpha_{N+1}(\max(i, \max(x))),$$

Rekurzív függvények Minimalizáció

Definíció: Minimalizáció

$f: N^n \to N$ függvény a g: $N^{n+1} \to N$ függvényből minimalizációval keletkezik úgy, hogy

$f(x_1,\ldots, x_n)=k$, amennyiben

1. $g(x_1,\ldots, x_n, k)=0$,
2. $\forall y<k$-ra $g(x_1,\ldots, x_n, y)\neq 0$, ($k$ az a <u>legkisebb</u> érték, amelyre <u>1. fennáll</u>);
3. $g(x_1,\ldots, x_n, k)$ értelmezve van, $\forall y\leq k$-ra (kiszámítható/kiértékelhető 0-tól k-ig)
4. Nincs értelmezve f az $(x_1,\ldots, x_n)$ helyen, ha vagy
   1. $\forall k$-ra $g(x_1,\ldots, x_n, k)\neq 0$,
   2. Vagy nem teljesül a 3. feltétel

Rekurzív függvények
Minimalizáció

Ezzel parciális függvényeket kapunk (nincs minden helyen értelmezve a lehetséges értelmezési tartományban)

Jelölés : $f(\vec{x}) = \mu g([g(\vec{x}, y) = 0])$

μ operátor definiciója

Pl. : $g(x, y) = x + y + 1, \quad f(x)$ sehol nincs értelmezve

Rekurzív függvények
Speciális minimalizáció

Definició: Speciális minimalizáció: reguláris minimalizáció

f: $N^n \to N$

Függvény a g: $N^{n+1} \to N$ függvényből reguláris minimalizációval keletkezik, ha

1. minimalizációval keletkezik
2. $\forall\ x_1,\ldots,\ x_n,\ \exists\ y\ g\,(x_1,\ldots,\ x_n,\ y)=0$,;
3. g teljes (totális) függvény ($dom(g) = \mathrm{N}$)

h reguláris fv. $\forall\ x_1,\ldots,\ x_n,\ \exists\ y\ h\,(x_1,\ldots,\ x_n,\ y)=0$

Szorzás fv. speciális minimalizációja: <u>reguláris</u>, $\forall\ x$-re, mert

$x \times 0=0$, μ(szorzás)=0

Az összeadás speciális minimalizációja: <u>nem reguláris</u> fv.

$x+y=0$, csak $x=y=0$ esetben,

μ(összeadás)=0 csak 0 pontban van definiálva, és nem definiált

$\forall\ x> 0$

Rekurzív függvények
Speciális minimalizáció

Parciális rekurzív függvények halmaza (Turing értelemben kiszámítható fv.-k):

Primitív rekurzív függvények+minimalizáció

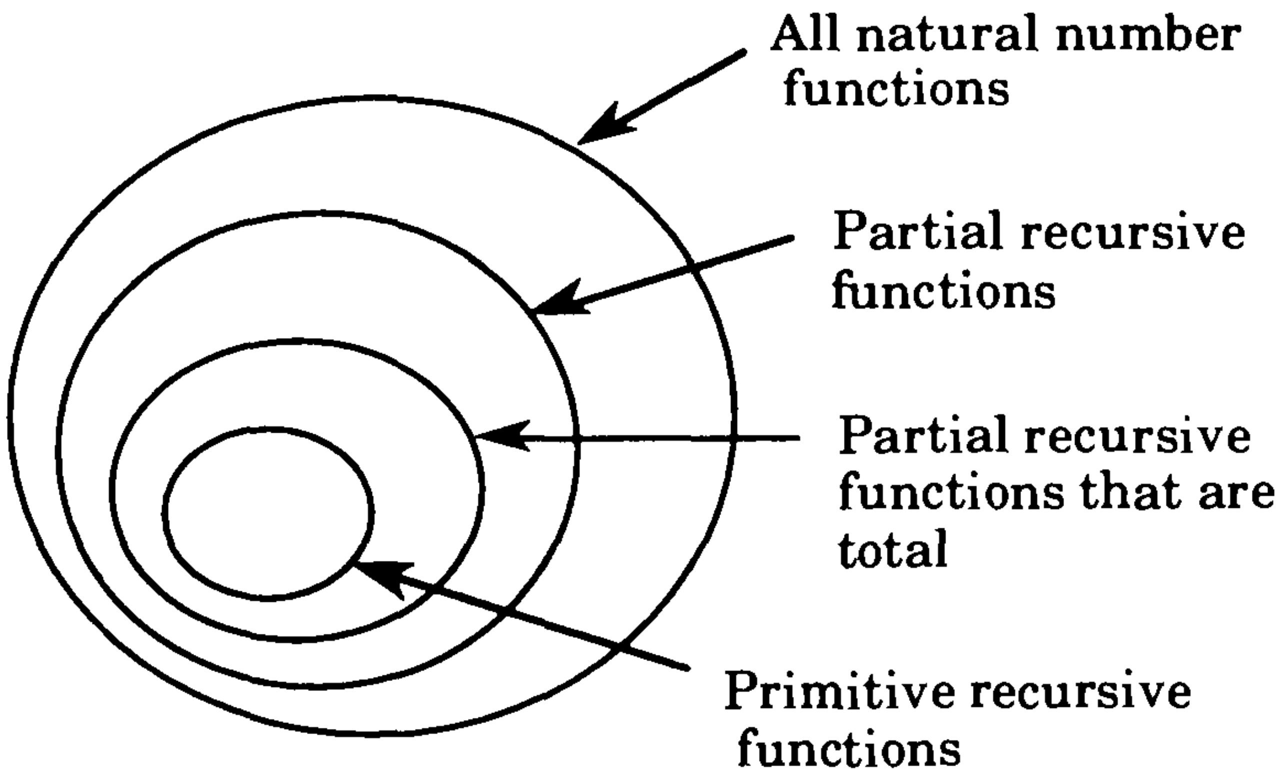
Rekurzív (teljes/totális) függvények

Primitív rekurzív függvények+reguláris minimalizáció

Zártság: - generáló sorozat ugyanúgy definiálható mint primitív rekurzív függvények esetében

Lényeges különbség a két osztály között:

- A parciális rekurzív függvények generáló sorozata szintaktikusan jellemezhető;
- A rekurzív függvényeknél a reguláris minimalizáció szemantikus, nem ellenőrizhető


All natural number functions
Partial recursive functions
Partial recursive functions that are total
Primitive recursive functions

Rekurzív függvények univerzális fv.

<u>Tétel</u>: Létezik

$U : N^2 \to N$ univerzális parciális rekurzív úgy, hogy bármely

az $f : N \to N$ – parciális rekurzív függvényhez létezik k,

hogy $U(x,k) = f(x)$ és

mindkettő ugyanott van értelmezve

Az átlós (diagonal) konstrukció ellenpélda létrehozására itt nem alkalmazható

$h(x)=U(x,x)+1$, és k legyen a h kódja

$h(k)= U(k.k)$

$h(k)=U(k.k)+1$, csak azt jelenti, hogy h nincs értelmezve k-ban

Rekurzív függvények
Kleene normál form

Az U konstruálása ugyanaz mint a primitív rekurzív függvények esetében.

U képzésében a felismerő rész primitív rekurzív, a sorszám keresése a minimalizáció. Ezért az U parciális függvény.

Kleene normál alak:

$$\exists U : N \to N$$

$$V : N^3 \to N$$

primitív rekurzív függvények úgy, hogy

bármely

$f : N \to N -$ ***parciális rekurzív függvényhez*** létezik $k \in N$,

hogy $f(x) = U\big(\mu t\big(V(k,x,t)=0\big)\big)\ \forall x.$

Rekurzív függvények rekurzívan felsorolható

Definíció : Az $A \subseteq N$ rekurzív halmaz ha karakterisztikus függvénye rekurzív (eldönthető hogy $x \in A$)

Definíció : Az $A \subseteq N$ rekurzívan felsorolható, ha létezik $f: N \to N$ rekurzív függvény, hogy $A=f(N)$ (ha $x \in A$, akkor ezt véges időben megtaláljuk), vagy $A=\emptyset$

$$\chi_{pA} = \begin{cases} 1 & \text{ha } x \in A \\ \text{nem definiált} & \text{ha } x \notin A \end{cases}$$

A parciális karakterisztikus függvénye

Rekurzív függvények
Rice tétel

<u>Rice tétel</u>:

Legyen $\mathcal{A}$ a parciális rekurzív függvények tetszőleges halmaza.

Az $A=\{n \mid f_n \in \mathcal{A}\}$ halmaz akkor és csak akkor eldönthető, ha vagy $\mathcal{A}=\emptyset$ vagy $\mathcal{A}$ az összes parciális rekurzív fv. (A a fv.-k index halmaza)

Rekurzív függvények
Rice tétel

Rice tétel:

Átfogalmazás:

Ha $A=\{n \mid f_n \in \mathcal{A}\}$ (fv. index halmaz) és $A \neq \emptyset$ vagy $A \neq \mathrm{N}$, akkor A nem eldönthető (nem kiszámítható). (a két triviális esettől eltekintve)

Rekurzív függvények
Rice tétel

$A \subseteq N$, a következő állítások ekvivalensek:

1. *A* rekurzívan felsorolható;
2. *A* az értelmezési tartománya (*dom*(*g*)= *A*) valamelyik parciális rekurzív függvénynek g: $N \rightarrow N$;
3. A parciálisan rekurzív karakterisztikus függvény χ_{pA} parciálisan rekurzív;
4. $\exists\, f$: $N \rightarrow N$ olyan parciálisan rekurzív karakterisztikus függvény, amelyre $A = f(N)$.
5. Vagy $A = \emptyset$, vagy $\exists\, f$: $N \rightarrow N$ olyan primitív rekurzív függvény, hogy $A = f(N)$.

Rekurzív függvények
Rice tétel

Az index halmazra (*A*) úgy gondolhatunk mint egy kiszámítható függvényre vonatkozó problémának egy olyan (program) kódjára, amely által adott válasz nem függ a függvény kiszámítására alkalmazott specifikus algoritmustól.

Természetesen sok nem-triviális eldönthetőségi problémára (pl. annak eldöntése, hogy egy természetes szám prím-e?) nem index halmazbeli elemmel, kóddal „kódoljuk”, ezért van eldönthető megoldásuk.

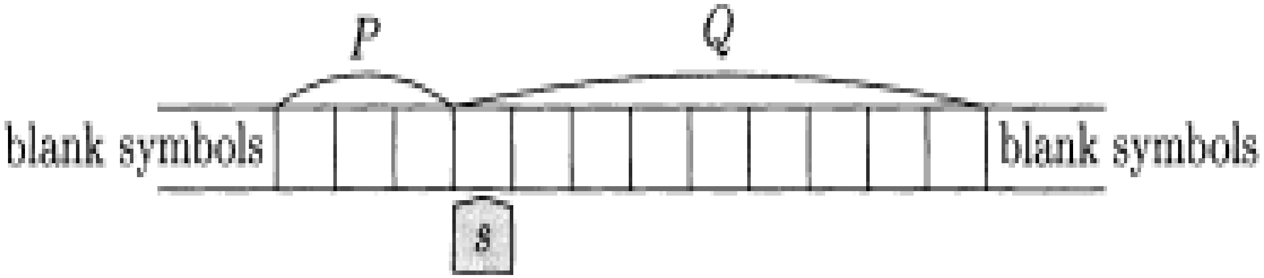
(Egy szám prímtényezős felbontása az NP).

Rekurzív függvények
Rice tétel

Ha *A* listába rendezhető, abból nem következik, hogy *A* eldönthető. Ha létezik egy eljárás arra, hogy felsoroljuk, listázzuk *A* elemeit, akkor ha $n \in A$, akkor meg fog jelenni a listán.

Azonban azt nem tudhatjuk általában, hogy mikor jelenik meg, ha $n \notin A$, akkor ez az eljárás nem fogja megmondani, hogy $n \notin A$. Azonban, ha $N \setminus A$ listázható, akkor *A* is eldönthető. Adott *n*-re *A és* $N \setminus A$ elemei felsorolhatók egy listában, , és csak azt kell megfigyelni, hogy melyik listában jelenik meg *n.*

Turing gép



Turing gép

Turing gép

(Q,T, δ, q_0,E) q_0 kezdő állapot, E az elfogadó állapot, Q az állapotok halmaza, T a szalag abc,

δ: $Q \times T \rightarrow Q \times T \times$(Jobb, bal, helyben).

Egy függvény kiszámítható a természetes számok halmazán, ha kiszámítható egy Turing géppel a természetes számok halmazán. Vagyis a Turing gép megáll, a megoldás leolvasható a szalagról.

Church –Turing tézis – a parciális rekurzív függvények pontosan azok, amelyek kiszámíthatók egy Turing géppel. (intuitíven az algoritmussal leírható parciális függvények azok, amelyek kiszámíthatók)

Más jelentés: ha egy függvény kiszámítható, az azt jelenti, hogy minden véges halmazon belül az összes értelmezési helyen véges időben megkapjuk az értékeit. Viszont általában nem tudjuk, hogy ahol még nem kaptuk meg az értékeit, ott értelmezve van-e,

Kiszámítható a függvény tér-idő táblázata. Véget ér onnan kezdve csupa „blank”(Λ).

Az ilyen táblázatok nem mind kiszámítható függvények táblázatai.

Kolmogorov bonyolultság (Kolmogorov-Chaitin-Solomonoff) Technikai előkészítés

N= {0,1, ….} nem negatív egészek

Ω={0,1}*, véges bináris szavak

Kölcsönösen egyértelmű megfeleltetés:

N	Ω
0	Λ
1	0
2	1
3	00
4	01
5	10
6	11
7	000
8	001

x lehet egyaránt $x \in N$ *vagy* $x \in \Omega$

x hossza $l(x)$

xy konkatenáció

x*y szorzat

Kolmogorov bonyolultság
Rendezett párok kódolása

	0	***1***	***2***	***3***
0	0	2	5	9
1	1	4	8	
2	3	7		
3	6			

Mellékátló lentről-fel

$P(x,y)=z$, $l(z)$- nagy mindkét változóban

Kolmogorov bonyolultság
Rendezett párok kódolása

Önkorlátozó kód: x=$\alpha_1 \ldots \alpha_n$; $l(x)=n$

$\alpha(x)= \alpha_1 \alpha_1 \ldots \alpha_n \alpha_n 01$ $l(\alpha(x))=2l(x)+2$
vagy

$\overline{x}= 1^n 0x=1\ldots10\ \alpha_1 \ldots \alpha_n$

$l(\overline{x})=2l(x)+1$

Standard önkorlátozó kód:

$x'=\overline{n}x;$

$l(x')=2\log_2 n+ l(x)+ 1= 2\log_2 (l(x))+ l(x)+ 1$

Lehet folytatni $\overline{n}$ *helyett,* n', *stb.*

Kolmogorov bonyolultság
Rendezett párok kódolása
Visszafejtés

$$\gamma_0(x,y)=\bar{x}y \quad \pi_{01}(\bar{x}y)=x \quad \pi_{02}(\bar{x}y)=y$$

$$\gamma_1(x,y)=x'y \quad \pi_{11}(x'y)=x \quad \pi_{12}(x'y)=y$$

$$l(\gamma_1(x,y))=2l(l(x))+1+l(x)+l(y)$$

Kolmogorov bonyolultság
Rendezett párok kódolása

x mennyire tömöríthető

Bonyolultság adott függvény szerint:

Def: Az x$\in\Omega$ bonyolultsága az *f*: *N*$\rightarrow$*N*, parciális rekurzív függvény szerint:

$$C_f(x) = min\{l(p): f(p) = x\}$$

∞ ha nincs p, amire $f(p) = x$

Megjegyzés: $C_f(x)$ nem mindig kiszámítható

Relatív bonyolultság

Kolmogorov-Chaitin-Solomonoff alaptétele, invariancia tétel (újabban)

Tétel: Létezik f_0: $N \rightarrow N$ optimális parciális rekurzív függvény, amelyhez bármely f: $N \rightarrow N$ parciális rekurzív függvényhez létezik k_f konstans, hogy

$C_f(x) \leq C_f(x) + k_f$, $\forall$ x-re

Következmény: (invariancia) f_0, g_0 optimális függvények,

$| C_{f0}(x) - C_{g0}(x) | < k_{f0,g0,}$ $\forall$ x-re

Tehát $C_{f0}(x)$ konstans erejéig egyértelmű

Relatív bonyolultság

Rögzítsük mostantól az f_0 optimális függvényt (Általában jó az univerzális Turing-gép megfelelő változata)

<u>Definició</u>: Az $x \in N$, Ω Kolmogorov bonyolultságán $C(x) = C_{f0}(x)$ függvényt értjük

Matematikai kitérő – Hossz-lexigrafikus rendezés

A véges bináris stringeken ($2^{<\omega}$) definiált rendezés, amelyben σ kisebb mint τ ($\sigma <_L \tau$), ha $|\sigma|<|\tau|$ vagy $|\sigma|=|\tau|$ és ahol *n* a legkisebb olyan *n*, $\sigma(n)=0$, és $\sigma(n)\neq \tau(n)$

Matematikai kitérő –

Values of $A(m, n)$

$m \backslash n$	0	1	2	3	4	n
0	1	2	3	4	5	$n+1$
1	2	3	4	5	6	$n+2=2+(n+3)-3$
2	3	5	7	9	11	$2n+3=2\cdot(n+3)-3$
3	5	13	29	61	125	$2^{(n+3)}-3$
4	13 $=2^{2^2}-3$	65533 $=2^{2^{2^2}}-3$	$2^{65536}-3$ $=2^{2^{2^{2^2}}}-3$	$2^{2^{65536}}-3$ $=2^{2^{2^{2^{2^2}}}}-3$	$2^{2^{2^{65536}}}-3$ $=2^{2^{2^{2^{2^{2^2}}}}}-3$	$\underbrace{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}}_{n+3}-3$

Matematikai kitérő –

Values of $A(m, n)$

m\n	0	1	2	3	4	n
0	0+1	1+1	2+1	3+1	4+1	$n+1$
1	A(0,1)	A(0,A(1,0))	A(0,A(1,1))	A(0,A(1,2))	A(0,A(1,3))	$n+2=2+(n+3)-3$
2	A(1,1)	A(1,A(2,0))	A(1,A(2,1))	A(1,A(2,2))	A(1,A(2,3))	$2n+3=2\cdot(n+3)-3$
3	A(2,1)	A(2,A(3,0))	A(2,A(3,1))	A(2,A(3,2))	A(2,A(3,3))	$2^{(n+3)}-3$
4	A(3,1)	A(3,A(4,0))	A(3,A(4,1))	A(3,A(4,2))	A(3,A(4,3))	$\underbrace{2^{2^{\cdot^{\cdot^{2}}}}}_{n+3}-3$
5	A(4,1)	A(4,A(5,0))	A(4,A(5,1))	A(4,A(5,2))	A(4,A(5,3))	$A(4, A(5, \text{n-1}))$
6	A(5,1)	A(5,A(6,0))	A(5,A(6,1))	A(5,A(6,2))	A(5,A(6,3))	$A(5, A(6, \text{n-1}))$

Matematikai kitérő –