ALGORITMUSOK ÉS BONYOLULTSÁGELMÉLET

Matematika MSc hallgatók számára

3. Előadás

Előadó: Hajnal Péter
Jegyzetelő: Hajnal Péter

2012. február 22.

1. $\mathcal{D}$-n kívül

Enlítettük, hogy a bonyolultságelmélet témája a $\mathcal{D}$-beli nyelvek vizsgálata, összehasonlításuk, bonyolultság szerint struktúrálásuk. A $\mathcal{D}$-n kívüli nyelvek is igen aktívan vizsgáltak. Kutatásuk módszerei és motivációja inkább a matematikai logikához köthető.

Egy új probléma esetén az első kérdés (döntési problémák esetén), hogy $\mathcal{D}$-hez tartozik-e. Nagyon sok matematikailag fontos, központi kérdés esetén kiderült, hogy nem $\mathcal{D}$-beli kérdésről van szó. Egy ilyen matematikai tétel jelentése az, hogy amíg a Church-tézis jól leírja a kiszámíthatóság fogalmát, addíg tudjuk, hogy a probléma általánosságban számítógéppel NEM kezelhető. Természetesen speciális inputokra, különböző feltételek mellett elképzelhető a kiszámíthatóság. Ilyen problémáknál a matematikai kutatásoknak ebbe az irányba kell tartaniuk.

Most néhány ilyen problémát sorolunk fel. Az első Hilbert X. problémája. Ennek történeti jelentősége van. Ez nagyban hozzájhárult a kiszámíthatóság fogalmának tisztázásához, ami elvezetett a Turing-gép definíciójához.

Példa (Hilbert X. Problémája). Legyen

$$DIOPHANTOSZ = \{\lceil p(x) \rceil : p \in \mathbb{Z}[x_1, x_2, \ldots, x_n],\ p\text{-nek van egész gyöke}\}.$$

Hilbert problémájának modern értelmezése, hogy $DIOPHANTOSZ$ nyelv $\mathcal{D}$-hez tartozik-e. (A probléma kitűzésének idejében $\mathcal{D}$ fogalma még nem született meg.) A klasszikus nyelven a probléma az, hogy van-e olyan algoritmus, ami egy adott egész együtthatós polinomról eldönti, hogy van-e egész gyöke.

Két lehetőség volt. Vagy valaki ad egy algoritmust, ami megoldja Hilbert problémáját (azaz DIOPHANTOSZ$\in \mathcal{D}$), a metematikusok közössége pedig megérti, ellenőrzi és elfogadja az algoritmust. A másik lehetőség: nincs ilyen algoritmus. Ebben az esetben ezt bizonyítani kell. Ez nem megy $\mathcal{D}$ definíciója nélkül. Kiderült, hogy a második lehetőseg az igazság.

Hilbert X. problémájának megoldásának története: 1900 Hilbert előadja a problémát, 1935 Church megfogalmazza a Church-tézist, 1936 Turing bevezeti a Turing-gép fogalmát, 1950-es és 60-as évek A Diophantikus halmazok bevezetése és vizsgálata Davies és Robinson vezetésével, 1970 Matijaszevics megteszi az utolsó (legnehezebb) lépéseket, bebizonyítja, hogy DIOPHANTOSZ nem tartozik $\mathcal{D}$-hez.

Természetesen DIOPHANTOSZ$\in \mathcal{S}$ (miért?).

Egy változó illetve lineáris eset könnyen megoldható. A kvadratikus kétváltozós eset is megoldható, de már komoly számelméleti vzisgálatok szükségesek.

Példa (Szóprobléma). SZÓPROBLÉMA inputja tartalmaz egy G csoportot. G-re multiplikatív írásmódot használva hívatkozunk. Mielótt leírnánk a teljes problém'at tisztáznunk kell, hogyan kódolhatunk csoportokat?

Egy lehetséges megoldást ad a kombinatorikus csoportelmélet. Legyen G egy csoport egy B generátorhalmazzal. Ekkor B elemeiből kifejezéseket épithetünk fel, amik a csoport egy-egy elemét írják le. Ha $B = \{a, b, c\}$, akkor $abbaca^{-1}ba^{-1}$ egy ilyen kifejezés. 1, az előző betűkészletből felírt üres szorzat is egy kifejezés, ami a csoport egységelemét írja le. Tehát a kifejezéseink, szakzsargonnal *szavaink*, B elemeiből és B elemeinek inverzéből szorzásokkal felépített kifejezések. Persze különböző szavak írhatják le ugyazt az elemet. A csoportszámtan garantálja, hogy $aa^{-1}b$ és b ugyanazt az elemet írja le.

Egy szó elemi egyszerűsítése az xx^{-1}, illetve $x^{-1}x$ egymásutáni két karakter kihúzása. Ha egy $w_1, w_2, w_3, \ldots, w_n$ szósorozatban bármely két egymásutáni szó közül egyik a másik elemi egyszerűsítése, akkor a sorozat bármely két eleme ugyanazt a csoportelemet írja le. Azt mondjuk w_1 és w_n ekvivalens. Ez egy ekvivalenciareláció a B-ből felírható csoportkifejezések halmazán. Az ekvivalenciaosztályok között könnyű szorzást, inverzet, egységosztályt definiálni. Így egy csoporthoz jutunk. Ez a B generátorhalmazhoz tartozó „legbővebb" generált csoport. A neve a B által szabadon generált csoport.

A B által szabadon generált csoport esetén könnyű tervezni egy algoritmust, amley két adott soz'ról eldönti, hogy ugyanazt a csoportbeli elemet írják-e le. Jóval általánosabb csoportok is leírhatók a fenti módszer általánosításával: Adjunk meg elemi egyszerűsítésekkel (és persze elemi bonyolításokkal) nem levezethető szóegyenlőségeket. Ha ilyen összefüggések egy halmazát adjuk meg, akkor ehhez is tartozik egy csoport: az elemi egyszerűsítés/elemi bonyolítás fogalmát ki kell terjeszteni az egyenlőség egyik oldalán szereplő kifejezés átírásával a másik oldalon szereplő kifejezésre. Így ha adott egy B halmaz és T egyenlőségek egy halmaza (ezek bal és jobb oldalán egy-egy szó szerepel), akkor egy $G = \langle B; T\rangle$ csoportot írtunk le.

Amennyiben B és T véges az így leírt csoportok a végesen prezentált csoportok. Például $\langle a, b; ab = ba\rangle$ egy csoport. könnyen ellenőrizhető, hogy ez $(\mathbb{Z}, +) \times (\mathbb{Z}, +)$.

Ezekután a problémánk: Legyen adva egy B véges generátorhalmaz, egy véges T összefüggés halmaz (igy adva van egy $G = G(B; T)$ végesen prezentált csoport). Adott még két B-re épített szó. Döntsük el, hogy azonos csoportbeli elemet írnak-e le.

Definíció.

$$\text{SZÓPROBLÉMA} = \{\ulcorner B, T; w_1 = w_2\urcorner : \text{a } \langle B; T\rangle \text{ csoportban a } w_1 \text{ és } w_2 \text{ csoportelemek megegyeznek}\}$$

A probléma eldönthetetlen,

$$\text{SZÓPROBLÉMA} \notin \mathcal{D}.$$

azaz

Igazából létezik olyan egyetlen végesen generált csoport, amely olyan komplex, hogy az erre vonatkozó szóprobléma (a csoport most nem része az inputnak) is eldönthetetlen.

Példa. HOMEOMORF inputja két topológikus tér. Azt kell eldöntenünk, hogy homeomorfak-e.

Ismét a lényeges kérdés: Hogyan kódolunk topológikus tereket? A legegyszerűbb megoldás a rekurzió: Egyszerű, jól ismertnek vett topológikus terekből egyszerű operációkkal „felépítünk" további, bonyolultabbakat. Talán a legkombinatorikusabb lehetőség, ha szimplexekből indulunk ki. Szimplexek a pontok, szakaszok, háromszögek, tetraéderek. Ezek pontosan a legfeljebb három-dimenziós szimplexek. Minden d természetes szám esetén definiálható egy d-dimenziós szimplex, például a $\mathbb{R}^d$ origója és e_i standard báziselemeinek konvex burka. A felépítés lehet a lap-menti ragasztás. A Könnyű igazolni, hogy csak a kiinduló szimplexek dimenziója és a ragasztásnál használt lapok ismerete elég a leírt topológikus tér homomorfiatípusának ismeretéhez. Ennek leírásához a szimplexeket és lapjaikat azonosítjuk csúcsaik halmazával. A szimpliciális komplexus egy halmazrendszer lesz egy véges V halmaz felett. A szimpliciális komplexus egyetlen tulajdonsággal jellemezhető: minden hozzátartozó halmaz összes részhalmaza is hozzátartozik (egy szimplex csúcsainak tetszőleges csúcshalmaza egy jól meghatározott lapja — ami szintén egy szimplex — csúcshalmaza).

A HOMEOMORF probléma (pontosabban a SZIMPLICIÁLIS-KOMPLEXUSOK-HOMEOMORFIZMUSA probléma) nem eldönthető. Azaz

$$\text{HOMEOMORF} \notin \mathcal{D}.$$

Példa. A POST problémában adott Σ véges ábécé. Az input egy dominó készlet: Véges sok dominótípus, ahol egy típus egy alsó és egy felső minta, ami egy-egy Σ^*-beli szó. Minden típusból végtelen sok dominónk áll rendelkezésünkre. Azt kell eldönteni, hogy ki tudunk-e rakni dominóinkból egy sort úgy, hogy az alsó és felső minták összeolvasva (konkatenálva) ugyanaz a szót adják.

A probléma a mi elemi tárgyalásunk helyett a félcsoportok nyelvén is elmondható. Az irodalomban legtöbbször félcsoportokra vonatkozó problémaként ismertetik ezt a nyelvet.

A probléma nem eldönthető.

$$\text{POST} \notin \mathcal{D}.$$

A fenti matematikai motivációjú problemák után megadunk egy eldönthetetlen nyelvet, ami a Turing-gépek definiációjával kapcsolatos. Be is bizonyítjuk eldönthetetlenségét. A példa Turing nevéhez fűződik, az egyik első kiszámíthatatlansági eredmény.

Példa.

Definíció.

$$\text{MEGÁLLÁS} = \{\lceil T, \omega \rceil : T \text{ leáll } \omega\text{-n, azaz STOP vagyis ELVET/ELFOGAD állapotba kerül}\}.$$

1. Tétel (Turing-tétel). *(i)* $\textit{MEGÁLLÁS} \in \mathcal{S}$,

(ii) $\textit{MEGÁLLÁS} \notin \mathcal{D}$.

Bizonyítás. A tétel első része következik az univerzális Turing-gép leírásából. A szimuláló gép leállását úgy kell módosítani, hogy ne a leálló állapotnak megfelelő állapotba jusson, hanem a leállás tényét bejelentő ELFOGAD állapotba kerüljön.

A második állítás bizonyítása indirekten történik, azaz tegyük fel, hogy létezik I Turing-gép, amely eldönti a MEGÁLLÁS nyelvet. Továbbiakban az indirekt feltevés I gépére alapítva egy kissé módosított gépet írunk le.

Definíció. Legyen $\widetilde{I}$ egy Turing-gép, amely eldönti, hogy i input kódol-e Turing-gépet. Amennyiben nem a gép ELVET állapotba kerül. Amennyiben $i = \lceil T \rceil$ az I MEGÁLLÁS nyelvet eldöntő Turing-gépet futtatja (i, i)-n. Ha a futtatás ELFOGAD állapottal ér véget, akkor jobbra-balra lépegető végtelen ciklusba megy át, ha ELVET állapottal ér véget, akkor STOP állapottal leáll.

Kérdés: Mit csinál $\widetilde{I}$ gép $\lceil \widetilde{I} \rceil$-n? Azaz mi történik, ha a saját kódján futtatjuk az $\widetilde{I}$ gépet?

A definíció alapján „kibontja magát", és a MEGÁLLÁS nyelvet eldöntő géppel eldönti, hogy a megadott inputon (ami esetünkben saját kódján) hogyan dolgozik. Ha I ELFOGAD állapotba kerül (a $\widetilde{I}$ gép $\lceil \widetilde{I} \rceil$-n megáll), akkor nem áll le (mármint $\widetilde{I}$ az $\lceil \widetilde{I} \rceil$-n), és ha I ELVET állapotba kerül (a $\widetilde{I}$ gép $\lceil \widetilde{I} \rceil$-n nem áll meg), akkor leáll. Mindenféleképpen ellentmondásra jutunk. ■

A bizonyítás lényege hasonlít Cantor bizonyítására, hogy $[0, 1]$ nem felsorolható/megszámlálhatóan végtelen halmaz (átlós módszer), csak ebben az esetben számok és sorszámok helyett gépek, illetve inputok kódjai szerepelnek.

A kurzus továbbiakban részében a $\mathcal{D}$ halmaz nyelveivel dolgozunk. Célunk az eldöntési problémák összehasonlítása, a döntési feladatok nehézségének mérése.

2. Nem-determinizmus

Az eddig tárgyalt kiszámíthatósági fogalom olyan volt, hogy ismert input esetén egyértelmű volt, milyen konfigurációsorozatot követve fut a gép. Az emberi gondolkozás nem így számol.

Az eredeti kiszámíthatósághoz egy jelzőt teszünk: a definiált Turing-gép egy determinisztikus gép. Vannak nem-determinisztikus gépek is. Az alábbiakban a nem-determinisztikus Turing gépekre két alternatív definíciót is adunk.

Definíció (I. változat). Hasonlóan mint a determinisztikus Turing-gépeknél, itt is vannak szalagok, fejek, állapotok, stb. Azonban az átmeneti függvényt már máshogy definiáljuk:

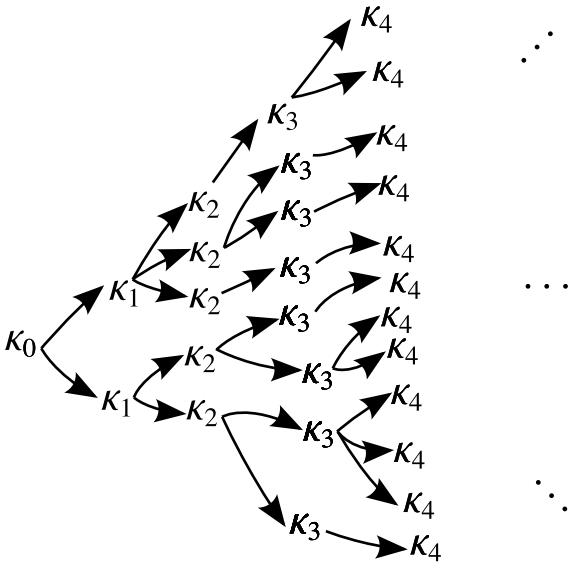
$$\delta \colon \Sigma \times \Gamma \times S \to \mathcal{P}(\{\leftarrow, ., \rightarrow\} \times \Gamma \times \{\leftarrow, ., \rightarrow\} \times S) \setminus \{\emptyset\}.$$

Azaz a konfiguráció látott része függvényében nem egy update-szabály adott, hanem update-szabályok egy halmaza. Így az ω inputhoz tartozó futás nem meghatározott (idegen szóval nem-determinisztikus), azaz a $\kappa_0(\omega)$ kezdőkonfigurációból több lehetséges konfiguráció felé mehetünk az átmenetifüggvény által leírt halmaz mindegyik eleme egy-egy lehetséges rákövetkező konfigurációt ad. Így egy $\kappa_0(\omega)$-ban gyökereztetett fa írja le a gép lehetséges futásait.

A nem-determinizmus megértéséhez nagyon fontos, hogy tisztázzuk mikor is számit ki egy gép egy nyelvet.

Definíció. Az ω inputot pontosan akkor fogadja el a T nem-determinisztikus Turing-gép, ha létezik ELFOGAD állapotba vezető futása. Azaz ω elvetése ekvivalens azzal, hogy ω-n minden futása ELVET állapothoz vezet.

⋆ ⋆ ⋆

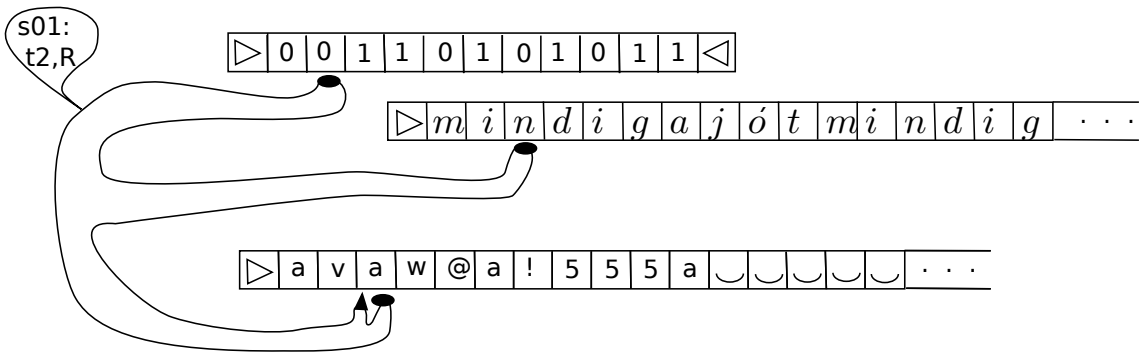


1. ábra.

Definíció (II. változat). Ebben az esetben egy plusz szalagunk lesz az input- és munkaszalagok között, az úgynevezett tanú/bizonyítás szalag. Ez a szalag csak olvasható és a fej csak jobbra tud mozogni rajta. Az átmeneti függvényt ugyanúgy definiáljuk, mint a determinisztikus esetben, és a futás is determinisztikus lesz, azaz ω és τ (a tanúszalag tartalma) egyértelműen meghatároz egy konfigurációsorozatot:

$$\kappa_0 = \kappa_0(\omega, \tau) \to \kappa_1 \to \kappa_2 \to \dots .$$

Az ω inputot pontosan akkor fogadja el egy nem-determinisztikus Turing-gép, ha van olyan τ tanúszalag tartalom, amelyre a futás ELFOGAD állapotba kerül.



2. ábra.

A két változatban egy lényeges különbség, hogy az elsőben a nem-determinizmus a futás során „szétszórt", az utolsó lépés előtt sem meghatározott a végső állapot. A második változatban a nem-determinizmus a τ választásával jelentkezik. A futás ezekután determinisztikus lesz.

Az alábbiakban a nem-determinisztikus számításon alapuló nyelvosztályokat vezetjük be. A nem-determinizmus kétféle változata közül bármelyikre alapozhatjuk

definícióinkat (úgy, hogy ugyanahhoz a nyelvosztályokhoz jussunk). Mi a második (tanúszalagos) szemléletet követjük.

Definíció. Legyen T egy II. értelemben vett nem-determinisztikus Turing-gép.

Ekkor $\mathcal{TIME}(\omega, \tau;T)$ és $\mathcal{SPACE}(\omega, \tau;T)$ a determinisztikus eset lemásolásával definiálható.

Legyen

$$\mathcal{NTIME}(\omega;T) = \begin{cases} \min\{\mathcal{TIME}(\omega, \tau;T) : \text{ahol } \tau \text{ olyan, hogy } \omega\text{-n} \\ \qquad\qquad\qquad\qquad T \text{ ELFOGAD állapotba jut}\}, & \omega \in L, \\ \min\{\mathcal{TIME}(\omega, \tau;T) : \text{ahol } \tau \in \Sigma^*\}, & \omega \notin L. \end{cases}$$

Azaz az elfogadó futások közül a „legzseniálisabb" tanúszalag-tartalom szabja meg az idő korlátot.

$$\mathcal{NSPACE}(\omega;T) = \begin{cases} \min\{\mathcal{SPACE}(\omega, \tau;T) : \text{ahol } \tau \text{ olyan, hogy } \omega\text{-n} \\ \qquad\qquad\qquad\qquad T \text{ ELFOGAD állapotba jut}\}, & \omega \in L, \\ \min\{\mathcal{SPACE}(\omega, \tau;T) : \text{ ahol } \tau \in \Sigma^*\}, & \omega \notin L. \end{cases}$$

Ezekután a nem-determinisztikus osztályok definíciója már értelemszerű.

Definíció.

$\mathcal{NP} = \{L :$ létezik olyan T nem-determinisztikus Turing-gép, ami L-et fogadja el
létezik olyan $i \in \mathbb{N}$, hogy minden ω-ra $\mathcal{NTIME}(\omega;T) \leq |\omega|^i + i.\}$

$\mathcal{NEXP} =$
$\{L :$ létezik olyan T nem-determinisztikus Turing-gép, ami L-et fogadja el
létezik olyan $i \in \mathbb{N}$, hogy minden ω-ra $\mathcal{NTIME}(\omega;T) \leq 2^{|\omega|^i+i}.\}$

$\mathcal{NL} =$
$\{L :$ létezik olyan T nem-determinisztikus Turing-gép, ami L-et fogadja el
létezik olyan $i \in \mathbb{N}$, hogy minden ω-ra $\mathcal{NSPACE}(\omega;T) \leq i \log(|\omega| + 1).\}$

$\mathcal{NPSPACE} =$
$\{L :$ létezik olyan T nem-determinisztikus Turing-gép, ami L-et fogadja el
létezik olyan $i \in \mathbb{N}$, hogy minden ω-ra $\mathcal{NSPACE}(\omega;T) \leq |\omega|^i + i.\}$

$\mathcal{NEXPSPACE} =$
$\{L :$ létezik olyan T nem-determinisztikus Turing-gép, ami L-et fogadja el
létezik olyan $i \in \mathbb{N}$, hogy minden ω-ra $\mathcal{NSPACE}(\omega;T) \leq 2^{|\omega|^i+i}.\}$

Ismét megjegyezzük, hogy (a nem-determinizmus kétféle változatán túl) az alapmodell különféle megállapodásait használva a definiált osztályok nem változnak.

Észrevétel. A determinisztikus osztályok zártak a komplementálásra. Például, ha $L \in_T \mathcal{P}$, akkor $\overline{L} = \Sigma^* - L$ is $\mathcal{P}$-hez tartozik. Az állitás bizonyít'sához legyen $\widetilde{t}$ az a Turing-gép, amit T-bőL az alábbi egyszerű változtatással kapunk: Az átmeneti függvényt úgy írjuk át, hogy ha T-nel az ELVET állapototírja el, akkor $\widetilde{T}$ (minden más megtartásával) az ELVET állapotba jusson. Illetve fordítva. Ezzel $\widetilde{T}$ pontosan a T által elvetett inputokat fogadja el. Azaz a kiszámított nyelv akomplementer nyelv lesz. Eközben ω inputhoz ugyanolyan hosszú futás tartozik (sőt a konfigurációk sorozatában csak az utolsó konfigurációk különböznek, azok is csak az állapotban. Speciálisan a munkaszalag tartalma T és $\widetilde{T}$ futásánál ugyanazok lesznek. Azaz T és $\widetilde{T}$ bonyolultság ugyanaz lesz.

A fenti észrevétel a nem-determinisztikus esetben távolról sem nyilvánvaló, sőt ha egy osztály zárt a komplementálásra, akkor annak igazolása nehéz. Emiatt a következő definíciók jogosak.

Definíció.

$$\begin{aligned}
co\,\mathcal{NP} &= \{\overline{L} : L \in \mathcal{NP}\},\\
co\,\mathcal{NEXP} &= \{\overline{L} : L \in \mathcal{NEXP}\},\\
co\,\mathcal{NL} &= \{\overline{L} : L \in \mathcal{NL}\},\\
co\,\mathcal{NPSPACE} &= \{\overline{L} : L \in \mathcal{NPSPACE}\},\\
co\,\mathcal{NEXPSPACE} &= \{\overline{L} : L \in \mathcal{NEXPSPACE}\},
\end{aligned}$$

3. Összefoglalás

Több idő, több nyelv. Több tár, több nyelv. A nem determinizmus ereje több nyelv. Ezen állítások nyilvánvalóak a definíciókból (amennyiben a „több" azt jelenti, hogy legalább annyi). Az is természetes, hogy korlátozott idő korlátozott tárfelhasználást is jelent. Ezek alapján az eddigi nyelvosztályokról a következő tartalmazások nyilvánvalók:

$$\begin{array}{ccccc}
 & & \mathcal{P} & \subseteq & \mathcal{EXP}\\
 & & \cap| & & \cap|\\
\mathcal{L} & \subseteq & \mathcal{PSPACE} & \subseteq & \mathcal{EXPSPACE}\\
\cap| & & \cap| & & \cap|\\
\mathcal{NL} & \subseteq & \mathcal{NPSPACE} & \subseteq & \mathcal{NEXPSPACE}\\
 & & \cup| & & \cup|\\
 & & \mathcal{NP} & \subseteq & \mathcal{NEXP}\\
 & & \cup| & & \cup|\\
 & & \mathcal{P} & \subseteq & \mathcal{EXP}
\end{array}$$