ALGORITMUSOK ÉS BONYOLULTSÁGELMÉLET

Matematika MSc hallgatók számára

2. Előadás

Előadó: Hajnal Péter
Jegyzetelő: Csenkei Anita 2011. február 8.

1. Az alapfogalmak összefoglalása

Az algoritmus naív fogalma: Egy eljárás, ami az adatok megkapása után egy jól definiált lépéssort elvégezve megadja a probléma megoldását. A feladatban szereplő adatokat *inputnak*, az eredményt *outputnak* nevezzük.

Az algoritmikus probléma egy $f : \{\text{input}\} \to \{\text{output}\}$ függvény.

Az információ kódolása: Az inputra, az outputra karaktersorozatként kell gondolnunk. Fontos, hogy mindenki tudja mit jelentenek ezek a karaktersorozatok. Hogy ezt elérjük egy jól definiált módon *kódolni* kell {input} és {output} halmazok elemeit.

Definíció. Σ *ábécé* egy nemüres véges halmaz. Elemeire mint *betűk* vagy *karakterek* hivatkozunk.

Σ^* a Σ ábécét használó véges hosszú sorozatok/*szavak* halmaza. $\epsilon \in \Sigma^*$ a 0 hosszú üres-szó.

Miután megállapodtunk egy kódolásban kapjuk az algoritmikus probléma matematikailag pontos fogalmát:

Az algoritmikus probléma: Egy $f : \Sigma^* \to \Sigma^*$ függvény.

Ennek egy fontos esete a *döntési probléma*: $f \colon \Sigma^* \to \{0, 1\}$. Ekkor csak „egy bit-et számolunk ki". Ha értéke 1, akkor azt mondjuk,hogy az inputot elfogadjuk, ha értéke 0, akkor az inputot elvetjük. Ez azonosítható Σ^* egy L részhalmazával: $L = f^{-1}(1) \subseteq \Sigma^*$. Σ^* részhalmazait *nyelveknek* nevezzük.

Ha nem döntési problémával dolgozunk, akkor azt hangsúlyozhatjuk úgy, hogy *számítási problémaként* hivatkozunk rá.

Az algoritmus formális definíciója: Turing-gép.

A Turing-gépnek szüksége van egy „térre", ahol a számítását végzi. Erről a térről (és a Turing-gép többi komponenséről) sokféle megállapodást köthetünk. Az alábbiakban néhány lehetőséget villantunk meg.

A számítás tere lehet egy (1) homogén pozícióhalmaz vagy egy (2) struktúrált pozícióhalmaz. Először tekintsük a struktúrálatlan esetet.

(1): Ekkor pozícióink egy sorozatot alkotnak. A pozíció egy alternatív neve „mező". Az eljárás alatt mindig a $\{p_i\}_{i\in\mathbb{N}}$ pozícióhalmaz tartalmai amit olvashatunk, illetve felülírhatunk. A pozíciókra úgy gondolunk, mint egy négyzethálós papírból kivágott jobb irányban végtelen (egy sor magas) szalagra. A szalag mezők sorozata, mely mezők tartalma mindig egy-egy karakter. Az input kódolásához használt Σ ábécé elemei mellett egy más Γ véges ábécé karaktereit írhatjuk a mezőkbe. Γ persze tartalmazhatja Σ elemeit is (általában gazdagabb mint a Σ).

Kezdetben szalagunkra írjuk az inputot $\triangleright$ és $\triangleleft$ jelek közé. Az első egy bevésett határolójel, a második csak az input végének jelölésére szolgál. A bevésett „szalag eleje" karaktert nem tudjuk felírni. Ez nem engedi, hogy szemünk lekerüljön a szalagról. Minden más hely a munkánk lehetséges helye.

Ez után kezdünk a munkához. A szalaghoz egy „szemmel" és egy „tollat tartó kézzel" kapcsolódik a Turing-gép. A szem és toll mindig ugyanazon mezőre irányul, egy mező „felett" van. A Turing-gép érdemi munkát végző, „gondolkodó" részét egy „fejnek" gondoljuk. A tollal felülírhatjuk a nézett mezőn lévő karaktert. Akár az input karaktereire is ráírhatunk új információt, ha ezt úgy gondoljuk.

Ha a fenti vázolt megállapodást kötjük, akkor *egy-szalagos modellről* beszélünk. Egy alternatíva, hogy szalagunk pozíció mindkét irányban a végtelenbe futnak, azaz pozícióink egy $\{p_i\}_{i\in\mathbb{Z}}$ halmazt alkotnak.

(2): Egy másik megállapodás/lehetőség amikor a számítás helye struktúrált. Először azt a lehetőséget említjük, amelyiket legtöbbször fogjuk használni.

Ekkor a számítás helye három szalagra van felosztva (mindegyik egy irányban (jobbra) végtelen sorozata mezőknek). Nevük *inputszalag*, *munkaszalag* és *outputszalag*.

Az inputszalaghoz tartozik egy szem, a munkaszalaghoz egy szem és egy tollat tartó kéz, az outputszalaghoz egy tollat tartó kéz. Tehát az inputszalagot (amely kezdeti tartalma az input karaktersorozat $\triangleright$ és $\triangleleft$ jelek között) csak olvashatjuk. A munkaszalag kezdeti tartalma egy $\triangleright$ karakter, majd üres/szűz jelet tartalmazó mezők. A „szalag eleje" jelet nem írhatjuk felül. A többi mező tartalmát Γ munkaábécé elemeivel írjuk fel. Ha a mező t tartalmát nem akarjuk felírni, akkor t-vel (eredeti tartalmával) írjuk felül. Feltesszük, hogy a „szűz mező" karakter nincs Γ-ban. Γ-ban lehet egy „üres-mező" karakter (ezt a jelet értelmezhetjük úgy is, hogy a kéz elhaladt felette, „csak" egy szinte láthatatlan karcolást ejtett rajta). Tehát a külső szemlélő is láthatja milyen mezők felett nem haladt még el a fej. Az outputszalag csak az output leírására szolgál. Az azt író kéz vagy egy üres mező felett áll, vagy jobbra mozoghat. Minden mozgás előtt az outputszalag aktuális mezőjét felülírjuk Σ (az output kódolására használt ábécé) egy karakterével. Ha ezekkel a megállapodásokkal élünk, akkor a *standard modellről* beszélünk.

Alternatív lehetőségek: A munkaszalag lehet két irányba végtelen szalag. A munkaszalag lehet két dimenziós (mint egy négyzethálós papír). Lehet k darab munkaszalag, ahol k (egy az inputtól nem függő pozitív egész). Ekkor a fej munkaterületre irányuló szem/toll komponense is k-szorozva van. Ez utóbbi esetben k-szalagos Turing-gépről beszélünk ($k > 1$). Tehát egy k-szalagos Turing-gépnek $1+k$ szeme és $k+1$ keze van, a szemek az inputszalag egy mezője és a munkaszalagok egy-egy mezője felett vannak. A munkaszalagok ezen egy-egy mezője felett van az első k kéz, amelyeken túl egy további kéz az outputszalag felett van.

Ha a fentiekben definiált komponenseit leírjuk a Turing-gépnek, akkor azt látjuk, hogy a szalagok mely pozíciói feltt vannak a szemek/kezek és mik az egyes mezők tartalmai. Az így leírt helyzetre úgy gondolhatunk mint a Turing-gépről egy *pillanatnyi fénykép felvételre*.

A fejnek minden pillanatban van egy állapota, mely egy S véges állapothalmazból kerül ki.

Definíció. Egy T Turing-gép egy pillanatnyi fényképe és a hozzátartozó állapot a T gép egy *konfigurációja*.

A Turing-gép „lelke” az átmenetifüggvény, amely megmondja, hogy egy konfigurációból a gép hogyan kerül a következőbe. A változás csak a lokálisan látott tartalomtól (a szemek által látott karakterek és az az állapot, amiben a gép van). Az átmeneti függvény ez alapján írja le, hogy a szemek/kezek hogyan mozogjanak. Ez a mozgás diszkrét folytonos, azaz minden lépés egy mezőnyi lépést vagy maradást jelent. A mozgás előtt, ahol lehet felülírás történik (az input és a határoló karakterek nem írhatók felül). Ezenkívül az átmenetifüggvény előírja az új konfiguráció állapotát.

A fenti leirás sok (rejtett) feltételt ír elő. Például ha a gép a $\triangleright$ határolójelen áll, akkor a balra lépés nem megengedett, az output kéz nem mozoghat jobbra.

Megjegyzés. A S véges állapothalmaz lehet struktúrált, mondjuk egy direktszorzat. Ekkor az első komponens tartalmazhat egy információt az inputszalagról, a második komponens pedig a munkaszalagra vonakozó információt tartalmazhat. Egy jellemző példa az állapotra: „Megyek az inputszalag elejére, közben számolom a lépéseket a munkaszalagon”.

Az algoritmus statikus leírása mellett van egy dinamikus szemlélet is.

Definíció. Legyen $\kappa_0(\omega)$ az ω input melletti kiinduló konfiguráció: Ebben az állapot egy START$\in S$ nevű speciális állapot. A kiinduló konfigurációban a fejek az első mezők (a baloldali határolójelek) felett vannak. A munka és output mezők tartalma a határoló jelek után következő minden mezőnél a szűz karakter. Az inputszalagon ω van leírva.

Az f átmeneti függvény alapján megadható egy rákövetkező konfiguráció, majd ennek rákövetkezője és így tovább. Egy ω-tól függő konfigurációsorozatot definiálunk:

$$\kappa_0(\omega) \curvearrowright \kappa_1 \curvearrowright \ldots \curvearrowright \kappa_i \curvearrowright \ldots$$

Ezt nevezzük T futásának ω-n.

Számítási feladatoknál szükség van egy STOP$\in S$ állapotra is. Ha κ_i-ben STOP állapotba jutottunk, akkor a „futás leáll”. Tehát a végtelen konfiguráció sorozatot végessé tesszük úgy, hogy az első olyan konfigurációnál, ahol az állapot STOP megállunk, a további konfigurációkat elhagyjuk. Az ekkori outputszalag-tartalom, az outputszalagra eddig kiírt értékek szolgáltatják a számítás végeredményét. Természetesen így is elképzelhető, hogy egy futás végtelen, azaz sose kerül olyan konfigurációba, ahol STOP az állapot.

Döntési feladatnál egy alternatív megállapodással élünk: STOP állapot helyett ELFOGAD és ELVET$\in S$ állapotok vannak (ekkor nincs is szükség outputszalagra). A szavak természetessé teszik a megállapodás formalizálását.

Definíció. $L \subseteq \Sigma^*$ nyelv kiszámítható/eldönthető, ha van olyan Turing-gép, amelynek minden $\omega \in L$ input esetén futása ELFOGAD állapottal tér véget, míg minden $\omega \notin L$ input esetén futása ELVET állapottal tér véget.

Speciálisan egy nyelvet elfogadó Turing-gép, egy feladatot kiszámító Turing-gép minden inputon megáll, azaz minden input esetén a futása véges.

Megjegyzés. Az, hogy létezik egy L nyelvet eldöntő T Turing-gép az azt jelenti, hogy megválaszthatjuk úgy a k szalagszámot, Γ munkaábécét, S állapothalmazt, továbbá leírhatunk egy f átmeneti függvényt, ami által leírt T Turing-gép a fentiek alapján eldönti L-et.

2. Az algoritmus fogalmának kialakulása

Az algoritmus matematikai fogalma a XX. század első harmadában alakult ki. Egyik ösztönzője Hilbert nevezetes problémái közül a tizedik. Ebben azt kérdezte, hogy: Van-e olyan algoritmus, ami egy adott egészegyütthatós polinomról eldönti, hogy van-e egész gyöke. Később kiderült, a válasz: nem. Ennek bizonyítása már nem képzelhető el az algoritmus fogalmának tisztázása nélkül.

Mivel az algoritmus/eljárás fogalmak mélyen gyökereznek az emberiség történetében ezért matematikai megfelelőjük kialakítása filozófikus kérdéseket vetnek fel. Nem lehet egy leegyszerűsített fogalomra azt mondani, hogy ez a matematikai algoritmus. A XX. század első felében több évtizedes munka eredménye, hogy a ma is elfogadott fogalom kialakult.

Az első matematikus aki kimondta, hogy ő a korrekt definíciót adta meg, az Alonso Church volt. Az 1935-ös bejelentését Church-tézisként hivatkozzák. Ezek után is több évig tartott, amíg általánosan elfogadottá vált. Ezen idő folyamán több másik, párhuzamosan kifejlesztett fogalommal való ekvivalenciája is bizonyított lett. Az elfogadottságának kialakulásában döntő szerepe volt Alan Turing fent ismertett fogalmának és annak a ténynek, hogy Church eredeti kiszámíthatóság fogalma ekvivalens ezzel.

3. A modell szerepe

Felmerül, hogy a fenti definícióban igen sok esetleges megállapodás szerepel. Milyen szerepük van ezeknek a megállapodásoknak? A kérdésre nem lehet egyértelmű választ adni. Igazából többféle válasz is van, amelyeknek van matematikai alapjuk.

I.

A modell nem központi. Modellünk mindegyik változatára alapítható egy kiszámíthatóság/eldönthetőség fogalom. Mindegyik módon ugyanahhoz a fogalomhoz jutunk. Később is látunk hasonló (sót a fentinél erősebb) jelenséget. Ha a futási idő fogalmát bevezetjük és csak annak nagyságrendje érdekel, akkor például vizsgálhatjuk a polinomiális időben eldönthető problémák osztályát. Ez sem fog függeni a megállapodásainktól. Ezeket a tapasztalatokat úgy fogalmazzák meg, hogy a fogalom *robosztusos*.

II.

A modell mégis számít. Erre egy példát adunk. Egy konkrét nyelv eldöntését vizsgáljuk meg két különböző modellben.

Definíció. Legyen

$$PALINDROM = \{\omega = \omega_1, \ldots \omega_n : \text{ahol } \omega_i = \omega_{n+1-i} \text{ minden } i \in \{1, 2, \ldots, n\} \text{ esetén}\}$$

a palindrom szavak nyelve.

Tehát döntési problémával állunk szemben. Adott egy szó, el kell döntenünk, hogy előlről és hátulról olvasva ugyanazt olvasuk-e.

II.a. A STANDARD Modell

Itt egy inputszalag és egy munkaszalag van. Döntési problémához nem kell outputszalag, ezt az S állapothalmaznak ELVET és az ELFOGAD eleme helyettesíti.

Egy Turing-gép algoritmust vázolunk. Ennek során elmondjuk, hogyan néznek ki a Turing-gépről készített pillanatfelvételek és szeemzgetünk az állapothalmazból.

Az állapothalmaz legyen

$$S = \{\text{START, MÁSOLOK, MÁSOLÁS-KÉSZ, INPUTFEJ-ELŐRE,}$$
$$\text{ELŐL-VAGYOK, TESZT, ELFOGAD, ELVET}\}.$$

A START állapotból egyet jobbra lép az input szem és a munka szem/kéz (mindkettő a szalaghatároló jel uáni első mező felett lesz), továbbá a gép a MÁSOLOK állapotba jut. Amíg ez lesz az állapot a gép átmásolja az inputot a munkaszalagra. Közben a két szem (ezzel együtt egy kéz) folyamatosan jobbra mozog. Ez akkor áll le, amikor az inputszalagon az $\lhd$ jel nem olvasható (MÁSOLÁS-KÉSZ állapot). Ebből mindkét szem balra egyet lép (a munka szem/kéz az átmásolt sorozat utolsó karaktere felett lesz), továbbá az INPUTFEJ-ELŐRE állapotba jut a gép. Ekkor a munkaszalag felett a szem/kéz mozdulatlan, az inputszalag feletti szem folyamatosan jobbra mozog. Egész addig, amíg a $\rhd$ jelet nem látja (ELŐL-VAGYOK állapot). Innen TESZT állapotba jut a gép az input szem eggyel jobbra mozgatása után (ekkor az input szem az input első karaktere fölé kerül, közben az output szem az átmásolt input utolsó karaktere felett maradt végig). A TESZT állapotban mindig ellenőrzi a gép, hogy a két szem ugyanazt látja-e. Ha valamikor ez nem teljesül, akkor ELVET állapotba jut, különben az input szem egyet jobbra, a munka szem egyet balra mozdul. Ez addig történik, amíg az input szem az input végét jelző karaktert nem látja (ekkor szükségszerű, hogy a munka szem a munkaszalag kezdetét jelző karakter felett legyen). Ha ez megtörténik, akkor a gép ELFOGAD állapotba kerül.

Minden ω inputra a futás hossza legfeljebb $3(n+1) = 3|\omega| + 3 = \mathcal{O}(|\omega|)$, ahol $n = |\omega|$, és az $\mathcal{O}$ (olvasd „nagy ordó”) egy felső becslést jelöl rejtett szorzó és additív konstanssal.

Definíció. T Turing-gép időigénye egy $\omega \in \Sigma^*$ inputon $TIME(\omega; T) = \ell$, mely egy „csonkított” konfigurációsorozat hossza (azaz a konfigurációk végtelen sorozatában megállunk az első olyannál, amelyben az állapot a számítás végét jelzi). Ekkor a futás $\{\kappa_i\}_{i=0}^{\ell}$, azaz az ℓ-edik konfigurációban kerül a gép először STOP vagy döntési feladatnál ELFOGAD/ELVET állapotokba.

Korábbi megállapítáunk az új jelöléssel kimondva

Minden $\omega \in \Sigma^*$ esetén $TIME(\omega; T) = \mathcal{O}(|\omega|)$, ahol T a fenti ismertetett, a PALINDROM nyelvet elfogadó gép.

Ez az eredmény a nagyságrend szempontjából éles. Pontosabban minden PALINDROM-ot kiszámító T Turing-gépre, van olyan ω input, amin T futása legalább $|\omega|$ hosszú. Feltéve, hogy $0 \in \Sigma \neq \{0\}$ a 0^n (n darab 0 karakter) esetén a gépnek el kell ezt fogadni, de ezt nem teheti meg az utolsó karakter elovasása nélkül. Ehhez viszont legalább n darab jobbra lépést kell tennie.

II.a. Az egy szalagos Modell

Most egy szalag van, ami egyben input- és munkaszalag is.

Először egy algoritmust/T' Turing-gépet adunk. Az egyszerűség kedvéért feltesszük, hogy $\Sigma = \{0, 1\}$. A munkaábécé legyen $\Gamma = \{0, 1, 0^{\checkmark}, 1^{\checkmark}\}$.

A következő állapothalmazzal dolgozunk:

$S = \{$START, ELŐLPIPÁL, HÁTUL-TESZT-0, HÁTUL-TESZT-1,
HÁTUL-TESZT-0-MOST, HÁTUL-TESZT-1-MOST, HÁTULPIPÁL,
ELŐL-TESZT-0, ELŐL-TESZT-1, ELŐL-TESZT-0-MOST,
ELŐL-TESZT-1-MOST, ELFOGAD, ELVET$\}$.

START állapotból egyet jobbra lép a szem/kéz (most egyetlen ilyenünk van) és ELŐLPIPÁL állapotba kerül. Ebben az állapotban felülírja a látott karaktert: 0-t $0^{\checkmark}$-re ír át, 1-re $1^{\checkmark}$-et ír rá. A látott karakter alapján HÁTUL-TESZT-0 vagy HÁTUL-TESZT-1 állapotba kerül és elindul jobbra. Elmegy az utolsó pipálatlan karakterhez. Persze ezt olvasáskor nem érzékeli, egyet túl kell mennie. Ha $\triangleleft$ vagy pipált karaktert lát, akkor balra (vissza)lép egyet és korábbi állapotának megfelelően HÁTUL-TESZT-0-MOST vagy HÁTUL-TESZT-1-MOST állapotba kerül. Ha a látott karakter nem egyezik meg az „emlékezett" karakterral, akkor ELVET állapottal megáll a gép. Ha a látott karakter megegyezik az „emlékezett" karakterral, akkor az input átment a teszten, balra lép a gép, HÁTUL-PIPÁLOK állapotba kerül, majd átírja az ott látott 0-t $0^{\checkmark}$-re, illetve az ott látott 1-t $1^{\checkmark}$-re. A kipipált karakter alapján ELŐL-TESZT-0 vagy ELŐL-TESZT-1 állapotban balra mozog a szem/kéz és megkeresi az útjába kerülő első pipa (az utoljára ELŐL-PIPÁLOK állapotban kipipált karakter) utáni mezőt és az itt látott karaktert teszteli, ahogy hátul tette. A részletek kidolgozását (igazából az átmeneti függvény tisztázását), speciálisan az ELFOGAD állapotba jutás feltételének leírását az érdeklődő diákok befejezhetik.

Könnyű látni, hogy minden ω inputon a fenti gép futása $\mathcal{O}(|\omega|^2)$ Ha ω egy palindrom szó, akkor a fenti T' gép futásának hosszát könnyen kiszámíthatjuk és ennek nagyságrendje $|\omega|^2$ lesz. A konstansokat nem számoltuk ki. Lényegtelen is, az állapotok számának növelésével a futási idő csökkenthető. Például használhatnánk a HÁTUL-TESZT-000, HÁTUL-TESZT-001, HÁTUL-TESZT-010, HÁTUL-TESZT-011, HÁTUL-TESZT-100, HÁTUL-TESZT-101, HÁTUL-TESZT-110, HÁTUL-TESZT-111 állapotokat input bitek hármasainak együttes tetszetlésére és kevesebbet kellene „ingáznia" a gépnek. Ennek ellenére a nagyságrend nem javítható.

1. Tétel. *Minden PALINDROM nyelvet kiszámító T egy-szalagos Turing-gép (!!) esetén van olyan $c_T > 0$ konstans és minden n inputmérethez van olyan $\omega \in \Sigma^n$, hogy*

$$TIME(\omega; T) \geq c_T \cdot n^2$$

teljesüljön.

Bizonyítás. Néhany terminológia bevezetésével kezdjük. Két szomszédos mező közös határát *ajtónak* nevezzük. Egy a ajtó az a^-, tőle balra és a^+, tőle jobbra lévő mezőket választja el. Ha a fej áthalad az ajtón, akkor a mozgása kétféle lehet: $\curvearrowright$, odairány vagy $\curvearrowleft$, visszairány.

Egy T Turing-gép ω inputon való futása minden ajtóhoz hozzárendel egy állapotsorozatot, amely azokat az állapotokat sorolja fel sorrendben, amelyekben a fej volt az áthaladások során. Legyenek ezek az állapotok: $\sigma(a; \omega) = (s_1, s_2, s_3, \ldots)$.

Megjegyzés. A mozgás irányát nem kell külön számon tartani, mert ezt tudjuk plusz információ nélkül is: az első áthaladás $\curvearrowright$, amellyel kezdve a mozgások/áthaladások iránya alternál.

Legyen a egy ajtó, amely az n hosszú inputok két szomszédos karakterét választja el. Legyen $\omega, \omega' \in \Sigma^n$ két n hosszú input. Mindkét inputnak van az a ajtóhoz tartozó állapotsorozata: $\sigma(a;\omega)$ és $\sigma(a;\omega')$. Legyen $\widetilde{\omega} = \omega|_a\omega'$ az az input, amely a ajtó előtti része ω-ból, az a ajtó utáni része ω'-ból adódik.

Észrevétel. Ha $\sigma(a;\omega) = \sigma(a;\omega')$, akkor T futása az $\widetilde{\omega} = \omega|_a\omega'$ inputon „összerakható" T-nek ω-n, illetve ω'-n való futásából.

Valóban. A futáshoz ismerni kell a szem/kéz mozgását, az állapotok sorozatát és az átírások pozícióinak, az átírt karaktereknek sorozatát. Az a ajtótól balra történt dolgok ugyanazok lesznek, amit az ω-n történő futásban látunk. Az a ajtótól jobbra történt dolgok ugyanazok lesznek, amit az ω'-n történő futásban látunk.

Ezt formálisan teljes indukcióval láthatjuk be a konfigurációsorozat indexére vonatkozólag. A lényeg, hogy az átmeneti függvény úgy szabályozza a gépet, hogy más helyen, más időben történt eseményekről csak az állapot révén emlékezhet a gép. A $\sigma(a;\omega) = \sigma(a;\omega')$ feltevés alapján az a ajtótól balra a gép az $\widetilde{\omega} = \omega|_a\omega'$ input mellett mindig ugyanazt látja, amit az ω inputon látna. Míg az a ajtótól jobbra a gép az $\widetilde{\omega} = \omega|_a\omega'$ input mellett mindig ugyanazt látja, amit az ω' inputon látna.

2. Következmény. *Ha $\sigma(a;\omega) = \sigma(a;\omega')$ és T futása ω-n és ω'-n is elfogadó, akkor az $\widetilde{\omega} = \omega|_a\omega'$ inputon is elfogadó.*

Azaz ha ω és ω' palindrom inputokra $\sigma(a;\omega) = \sigma(a;\omega')$, akkor $\widetilde{\omega} = \omega|_a\omega'$ is palindrom.

Például $\omega = 101|110011101$ és $\omega' = 101|010010101$ esetén $\omega|_a\omega' = 101|010010101$

Mostantól kezdve a technikai részleteket egyszerűsítő $3|n$ feltétellel élünk! Legyen

$$I_0 = \{\omega_0 0^{n/3} \overleftarrow{\omega_0} : \omega_0 \in \Sigma^{n/3}\}$$

Nyilván $I_0 \subset PALINDROM$, továbbá $|I_0| = |\Sigma|^{n/3}$.

Ha a a középső ($n/3$ darab) karaktert határoló egyik ajtó, akkor azt „középső ajtónak" nevezzük.

$\omega_0, \omega_0' \in \Sigma^{n/3}$ két KÜLÖNBÖZŐ karaktersorozatokhoz tartozó I_0-beli $\omega_0 0^{n/3}\overleftarrow{\omega_0}$ és $\omega' = \omega_0' 0^{n/3}\overleftarrow{\omega_0'}$ (elfogadandó) inputokra és a középső ajtóra $\omega|_a\omega'$ biztos NEM palindrom. Így kapjuk a következő fontos észrevételt.

Észrevétel. Legyen $\omega \neq \omega' \in I_0$ és egy a középső ajtó. Ekkor $\sigma(a;\omega) \neq \sigma(a;\omega')$.

A fenti észrevétel ad számunkra $|\Sigma|^{n/3}$ különböző ω inputot, amelyre a $\sigma(a;\omega)$ sorozatok különbözők minden a középső ajtóra.

Észrevétel. A 0 hosszú állapotsorozatok száma 1. Az egy hosszú állapotsorozatok száma $|S|$. A kettő hosszú állapotsorozatok száma $|S|^2$. Az x hosszú állapotsorozatok száma $|S|^x$. A legfeljebb x hosszú állapotsorozatok száma $1+|S|+|S|^2+\ldots+|S|^x$. Ez utóbbi szám pontosan $(|S|^{x+1}-1)/(|S|-1)$, ami könnyen becsülhető felülről $|S|^{x+1}$-nel.

3. Következmény. *Ha x olyan, hogy $|S|^{x+1} \leq |I_0|/2$, akkor I_0 elemeinek legalább fele olyan, hogy a hozzá tartozó $\sigma(a;\omega)$ sorozat hosszabb mint x, minden középső ajtóra.*

Fontos észrevenni, hogy x választható úgy, hogy teljesítse a fenti feltételt és közben $x \geq \beta_T \cdot n$. A továbbiakban x értékét $\beta_T \cdot n$-nek vesszük.

Legyen N_a azon „nehéz" I_0-beli inputok halmaza, amelyeken T futása olyan, hogy legalább x-szer áthalad az a ajtón.

Ekkor

$$\begin{aligned}\sum_{\omega \in I_0} TIME(\omega, T) &\geq \sum_{\omega \in I_0} \sum_{a \text{ középső ajtó}} |\sigma(a;\omega)| = \sum_{a \text{ középső ajtó}} \sum_{\omega \in I_0} |\sigma(a;\omega)| \geq \\ &\geq \sum_{a \text{ középső ajtó}} \sum_{\omega \in N_a} |\sigma(a;\omega)| \geq \sum_{a \text{ középső ajtó}} \sum_{\omega \in N_a} x \geq \\ &\geq \sum_{a \text{ középső ajtó}} |N_a| \cdot x \geq \frac{n}{3} \cdot |N_a| \cdot x \geq \frac{n}{3} \cdot \frac{|I_0|}{2} \cdot \beta_T n.\end{aligned}$$

Mindkét oldalt $|I_0|$-mal osztva kapjuk, hogy az I_0-beli inputok átlagos futási ideje legalább $\frac{\beta_T}{6} \cdot n^2$. Ez az állítást igazolja. ■