Algoritmuselmélet és bonyolultságelmélet MSc hallgatók számára

Bonyolultsági nyelvosztályok

2017. Előadó: Hajnal Péter

1. Bonyolultsági mértékek

Most definiáljuk egy algoritmus/Turing-gép időigényét és tárigényét.

Definíció. Egy T *Turing-gép időigénye* egy ω inputon $\ell := \mathit{TIME}(\omega, T)$, ha futása $\{\kappa_i\}_{i=0}^{\ell}$, vagyis az ℓ-edik konfigurációban kerül először STOP állapotba, illetve ∞, ha futása végtelen (nem éri el a STOP állapotot). Természtesen döntési feladatokra specializált Turing-gépek esetén is használni fogjuk ezt a fogalmat, de a STOP szerepét $\mathit{ELFOGAD/ELVET}$ veszi át.

Definíció. Egy T *Turing-gép tárigénye* egy ω inputon $s := \mathit{SPACE}(\omega, T)$, ha futása során a munkaszem/kéz alatti mező legnagyobb indexe s, illetve ∞, ha a munkaszem/kéz tetszőleges messze elmozdul a munkaszalag baloldali határától.

Mivel a Turing-gép legfeljebb annyi mezőt látogathat meg a munkaszalagon, amennyit mozog, ezért ezen jellemzők között fennáll a nyilvánvaló

$$\mathit{SPACE}(\omega, T) \leq \mathit{TIME}(\omega, T)$$

összefüggés. A futás idő- és tárigényét nyilván az input hosszától való függése alapján ítélhetjük meg. A következő definícióval ezen függést tudjuk kifejezni.

Definíció. Legyen $t\colon \mathbb{N} \to \mathbb{R}$ egy tetszőleges függvény. Azt mondjuk, egy T Turing-gép eleme a $\texttt{TIME}(\texttt{t}(\texttt{n}))$ halmaznak, ha minden $\omega \in \Sigma^*$ esetén

$$\mathit{TIME}(\omega, T) \leq t(|\omega|).$$

Definíció. Legyen $s\colon \mathbb{N} \to \mathbb{R}$ egy tetszőleges függvény. Azt mondjuk, egy T Turing-gép eleme a $\texttt{SPACE}(\texttt{s}(\texttt{n}))$ halmaznak, ha minden $\omega \in \Sigma^*$ esetén

$$\mathit{SPACE}(\omega, T) \leq s(|\omega|).$$

Természetesen ezek az osztályok nyelvosztályokkként is megfogalmazhatók:

Definíció. Legyen $t\colon \mathbb{N} \to \mathbb{R}$ egy tetszőleges függvény. Azt mondjuk, hogy egy L nyelv eleme a $\mathcal{TIME}(t(n))$ halmaznak, ha létezik olyan T Turing-gép, amely

(i) eldönti L-et, és

(*ii*) $T \in \texttt{TIME}(\texttt{t}(\texttt{n}))$, azaz $TIME(\omega, T) \leq t(|\omega|)$.

Definíció. Legyen $s\colon \mathbb{N} \to \mathbb{R}$ egy tetszőleges függvény. Azt mondjuk, hogy egy L nyelv eleme a $\mathcal{SPACE}(s(n))$ halmaznak, ha létezik olyan T Turing-gép, amely

(*i*) eldönti L-et, és

(*ii*) $T \in \texttt{SPACE}(\texttt{s}(\texttt{n}))$, azaz $SPACE(\omega, T) \leq s(|\omega|)$.

2. Bonyolultsági nyelvosztályok

Az imént bevezetett $\mathcal{TIME}(t(n))/\mathcal{SPACE}(s(n))$ jelöléseknél hasznosabbak azok az osztályok ahol az idő, illetve tár korlátozást nem egy függvénnyel, hanem egy „nagyságrenddel" írjuk elő.

Definíció. ∘ *Polinomiális időben eldönthető nyelvek*:

$$\mathcal{P} := \bigcup_{p \in \mathbb{R}[x]} \mathcal{TIME}(p(n)) = \bigcup_{a \in \mathbb{N}} \mathcal{TIME}(an^a + a).$$

A második uniós alak csupán egy speciális polinom sorozatra ritkítja ki az első uniót. A speciális polinom sorozat tudja, hogy bármely $p \in \mathbb{R}[x]$ polinomhoz létezik olyan $a \in \mathbb{N}$ természetes szám, melyre $p(n) \leq an^a + a$, ahol $n \in \mathbb{N}$. Ezért a két unió egyenlősége nyilvánvaló. A két unió csak két alternetíva a megfogalmazásra, nem része a definíciónak.

∘ *Exponenciális időben eldönthető nyelvek*:

$$\mathcal{EXP} := \bigcup_{a \in \mathbb{N}} \mathcal{TIME}(2^{an^a + a}).$$

∘ *Polinomiális tárral eldönthető nyelvek*:

$$\mathcal{PSPACE} := \bigcup_{a \in \mathbb{N}} \mathcal{SPACE}(an^a + a).$$

∘ *Exponenciális tárral eldönthető nyelvek*:

$$\mathcal{EXPSPACE} := \bigcup_{a \in \mathbb{N}} \mathcal{SPACE}(2^{an^a + a}).$$

∘ *Logaritmikus tárral eldönthető nyelvek*:

$$\mathcal{L} := \bigcup_{a \in \mathbb{N}} \mathcal{SPACE}(a \log n),$$

itt megjegyezzük, hogy $\log n$ az $n = 0$ esetben nyilván nem értelmezhető, azonban ettől az elfajuló (nulla hosszúságú input) esettől eltekintünk. Továbbá $\mathcal{L}$ miatt szükséges az inputszalag és a munkaszalag elkülönítése is, mert máskülönben nehéz lenne a logaritmikus tárat mérni/kimutatni.

Könnyen belátható, hogy a nyelvosztályok között az alábbi tartalmazások teljesülnek:

$$\begin{array}{ccccccccccc} \mathcal{L} & \subseteq & \mathcal{PSPACE} & \subseteq & \mathcal{EXPSPACE} & \subseteq & \mathcal{D} & \subseteq & \mathcal{S} & \subseteq & \mathcal{P}(\Sigma^*) \\ & & \cup| & & \cup| & & & & & & \\ & & \mathcal{P} & \subseteq & \mathcal{EXP} & & & & & & \end{array}$$

Természetesen további kapcsolatok is vannak, ezekről a későbbiek során esik szó. Nyelvosztályokból is jelentősen több létezik, mint amit itt megadtunk, a részletekért lásd a Qwiki oldalát: http://qwiki.stanford.edu/index.php/Complexity_Zoo.

3. További tartalmazások

Célunk, hogy belássuk:

$$\mathcal{L} \subseteq \mathcal{P} \subseteq \mathcal{PSPACE} \subseteq \mathcal{EXP} \subseteq \mathcal{EXPSPACE} \subseteq \mathcal{D} \subseteq \mathcal{S} \subseteq \mathcal{P}(\Sigma^*).$$

A fenti tartalmazások jó része nyilvánvaló. A teljes tartalmazás sorozat a következő lemmából nyilvánvaló.

1. Lemma.

$$\mathcal{SPACE}(s(n)) \subseteq \cup_{c\in\mathbb{N}}\mathcal{TIME}(c^{s(n)+\log(n+1)}).$$

Bizonyítás. Legyen $L \in \mathcal{SPACE}(s(n))$. Ekkor megadható olyan T Turing-gép, amely eldönti L-et (speciálisan minden $\omega \in L$-en megáll), és a tárigénye legfeljebb $s(n)$.

A fenti bekezdés egy karakterisztikus bizonyítás kezdés lesz. A továbbiakban ezt $L \in_T \mathcal{SPACE}(s(n))$ jelöléssel rövidítjük.

Legyen $\kappa_0(\omega) \to \kappa_1(\omega) \to \kappa_2(\omega) \to \ldots \to \kappa_\ell(\omega)$ a futás ω-n. Azaz ez egy $\ell \geq 1$ hosszú, véges konfigurációsorozat, ahol az első konfiguráció ($\kappa_0(\omega)$) a kiinduló konfiguráció (ebben az állapot START) és az utolsó állapot ($\kappa_\ell(\omega)$) az első olyan konfigutáció a futás során, amelyben az állapot ELFOGAD/ELVET.

Könnyű látni, hogy a futás során nem ismétlődhet konfiguráció, azaz $i \neq j$ esetén $\kappa_i \neq \kappa_j$ teljesül. Valóban minden konfiguráció egyértelműen meghatározza a rákövetkezőt, így ismétlődés egy végtelen, periodikus konfigurációsorozathoz vezetne.

Hányféle konfiguráció léphet fel a fenti sorozatban rögzített ω esetén? Legyen $|\omega| = n$. Egy felső becslés a kérdésre adandó válaszra (α_T, β_T konstansok T-től függnek):

$$(n+2)\cdot(s(n)+1)^k \cdot |\Gamma|^{k\cdot s(n))} \cdot |S| \leq \alpha_T(n+1)\alpha_T^{s(n)} \leq \beta_T^{s(n)+\log(n+1)},$$

hiszen az input szem helyzete $n+2$-féle, mind a k munkaszalag felett: a munka szem helyzete $s(n)+1$-féle, a munkaszalag tartalma $(|\Gamma|+1)^{s(n)}$-féle (az $s(n)$-nél nagyobb

indexű munka-mezők szükségszerűen az érintetlen karaktert tartalmazzák), továbbá az állapot $|S|$-féle lehet.

Összefoglalva: Ha a futási idő ${\beta_T}^{s(n))+\log n}$-nél hosszabb lenne, akkor a futás során a konfigurációk ismétlődnének, így a futás végtelen lenne.

Tudjuk, hogy nincs így. Tehát kaptuk, hogy T időigénye automatikusan megfelel az észrevételben szereplőkkel. Azaz a kezdeti T igazolja az állítást. ■

4. Robusztusság

A kiszámíthatóságnál/eldönthetőségnél már szó volt arról, hogy ezek a fogalamak a Turing-gép definícióján alapulnak. A Turing-gép matematikai definíció sok apróságot tartalmaznak, amik bizonyos szemszögből hasznosak. Az apróságokban azonban nem remélhető egységesítsés. Sokan sokféleképpen írhatják le a sz'amukra legszimpatikusabb változatát a Turing-gép fogalmának. Ezek a változatok sokfélék lehetnek. Ennek ellenére nem hatnak ki a kiszámíthatóság/eldönthetőség fogalmára.

Hasonló megjegyzés igaz a fent bevezetett nagyságrendi korlátokkal definiált nyelvosztályokra. $\mathcal{P}$ a polinom időben eldönthető nyelvek osztálya ugyanaz marad, ha nem k, az inputtól elkülönített jobbra végtelen munkaszalaggal dolgozunk, ahová nem engedjük meg az „érintetlen" karakter írását. $\mathcal{P}$ egy robusztus nyelvosztály.

Jóval nagyobb problémát okozna, ha definiálnánk a $\mathcal{LINEAR} := \{L(T) : T \in \cup_{a\in\mathbb{N}}\texttt{Time}(an + a)\}$ nyelvosztályt. Ez a nyelvosztály már nem annyira robusztus. Ha ilyennel talalkoznánk, akkor nagyon részletesen értsük meg mely eldönthetőség definiációra alapul az időkorlát. Hasonlóan komoly technikai problémakat vetne fel az $\mathcal{E} := \{L(T) : T \in \cup_{a\in\mathbb{N}}\texttt{Time}(a^{n+a})\}$ nyelvosztály.

Korábbi definícióink nem voltak véletlenek. A fenti két példával csak azok dolgozzanak, akik a bonyolultságelmélet alapjait már mélyen megértették és szembesültek a technikai nehézségekkel. A következő szekció erre a problémara világít rá egy példával.

5. A PALINDROM nyelv példája

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. Így nem kell outputszalag, ezt az S állapothalmaznak ELVET és az ELFOGAD eleme helyettesíti.

Két Turing-gépet/algoritmust vázolunk. Ennek során elmondjuk, hogyan néznek ki a Turing-gépről készített pillanatfelvételek és szemezgetünk az állapothalmazból. Az egyszerűség kedvéért feltesszük, hogy $\Sigma = \{0, 1\}$.

1. algoritmus/Turing-gép

Ez egy egyszalagos Turing-gép lesz a $\Gamma = \{0, 1, 0^{\checkmark}, 1^{\checkmark}\}$ munkaábécével.

A START állapotból egyet jobbra lép az input szem/kéz (a szalaghatároló jel utáni első mező, az input első karaktere felett lesz). Az új állapot ELŐL-PIPÁL lesz: A karaktert „megjegyzi", felülírja pipált változatával és megkeresi az utolsó inputkaraktert. Ehhez HÁTUL-TESZT-0, HÁTUL-TESZT-1 állapotokat használjuk. Ekkor az input szem/kéz folyamatosan jobbra mozog. Ez akkor áll le, amikor az inputszalagon az $\triangleleft$ jel nem olvasható, majd egyet visszalép (ezt a későbbiekben egy kissé felülírjuk). Ekkor megtalálta a karaesett karaktert és teszteli, hogy megegyezik-e az első karakterrel: HÁTUL-TESZT-0-MOST, HÁTUL-TESZT-1-MOST állapotba kerül. Az állapot bitje az előlről hozott „emlékezet". Ha a látott bit nem egyezik a hozott emlékezettel, akkor a gép ELVET állapotba kerül, leáll. Ha egyezik, akkor HÁTUL-PIPÁL állapotba kerül: felülírja a karaktert a pipált változatával és egyet balra lép. Az olvasott karaktertől függően ELŐL-TESZT-0, ELŐL-TESZT-1 állapotba kerül: balra megy, amíg el nem ér egy pipált karaktert és ennek elérésekor visszalép. Ezzel ELŐL-TESZT-0-MOST és ELŐL-TESZT-1-MOST állapotok egyikébe kerül, ahol a bit a „hátulról hozott emlékezet". Vagy leállunk, vagy pipálunk és egyet jobbra lépünk. Az itt látott karakter alapján HÁTUL-TESZT-0, HÁTUL-TESZT-1 állapotok egyikébe kerülünk. (Az állapothalmaz egy kis memóriát szimulál.) Korábban azt mondtuk, hogy ekkor az utolsó input karaktert keressük meg. Most pontosítunk: folyamatosan jobbra mozog az utolsó pipálatlan karakterig. Azaz a mozgás akkor áll le, amikor az inputszalagon az $\triangleleft$ jel vagy pipált jel található, majd egyet visszalép.

A Turing-gép eddigi munkájához következő állapothalmazt használtuk

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

Az elfogadó leálláshoz ismét módosítunk a korábbi (az általános teendőket mutató) leíráson: Ha az HÁTUL-TESZT-0, HÁTUL-TESZT-1, ELŐL-TESZT-0, ELŐL-TESZT-1 állapotba jutáskor egy pipált karaktert látunk, akkor a fenti „generikus" utasítás leírás helyett ELFOGAD állapotba kerülünk: Ez a helyzet/konfiguráció azt jelenti, hogy a bal és jobb pipák „összeértek", inputunk palindrom szó. Az átmeneti függvény formalizálását a fenti „mesélő leírás" alapján az érdeklődő hallgatóra bízzuk.

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 tesztelésére és kevesebbet kellene „ingáznia” a gépnek.

A fenti algoritmus nem hatékony. Az inputnak csak egy példánya áll rendelkezésére egy szemnek. Ez kényszeríti ki az ingázást. Ha van két szalagunk és két szemünk, akkor hatékonyabbak lehetünk.

2. algoritmus/Turing-gép

A standard modellt használjuk, $\Gamma = \Sigma = \{0, 1\}$, az állapothalmaz legyen

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

Az inputot átmásoljuk a munkaszalagra. Ennek befejezésével MÁSOLÁS-KÉSZ állapotba jutunk.

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 $\triangleright$ 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.

A fenti „szöveg” egyszerűen megfogalmazható az átmeneti függvény alkalmas definíciójával.

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 először STOP vagy döntési feladatnál ELFOGAD/ELVET állapotok.

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 $0 \in \Sigma$ a 0^n (n darab 0 karakter) esetén a gépnek el kell ezt fogadni, de ezt nem teheti meg az utolsó karakter elolvasása nélkül. Ehhez viszont legalább n darab jobbra lépést kell tennie.

A második algoritmus fontos eleme volt, hogy kettő szemet (input és munka) használjunk, így ez az egyszalagos modellben **nem** valósítható meg.

Az algoritmusok időigényét is meghatároztuk:

Észrevétel. 1. algoritmus (T_1 Turing-gép) négyzetes rendű: $TIME(\omega, T_1) \leq |\omega|^2$,

2. algoritmus (T_2 Turing-gép) lineáris futásidejű: $TIME(\omega, T_2) \leq 3|\omega| + 3$.

Látjuk, hogy a második algoritmus, amely kisebb futásidejű, nem valósítható meg az egyszalagos modellen. A következő tétel szerint, ha egyszalagos modellt használó algoritmussal akarjuk eldönteni a *PALINDROM* nyelvet, akkor a négyzetes időigény lényegében nem javítható.

2. Tétel. *Ha T egy olyan Turing-gép, amely az egyszalagos modellben eldönti a PALINDROM nyelvet, akkor* $\forall\, n,\ \exists\, \omega \in \Sigma^n$:

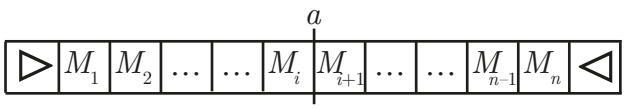
$$TIME(\omega, T) \geq \alpha_T |\omega|^2,$$

valamely α_T pozitív konstansra.

A tétel bizonyításához szükségünk lesz néhány új fogalomra.

Definíció.

Az inputszalag két szomszédos mezőjének közös határát *ajtónak* nevezzük. Ha az inputszalag mezőit úgy képzeljük el, mint végtelen egymásba nyíló szobasorozatot, akkor azt mondhatjuk, hogy a fej csak az ajtókon át tud közlekedni.



1. ábra. Az M_i és M_{i+1} mezőket elválasztó a ajtó.

Tekintsük egy T Turing-gép ω inputon való futását. Ekkor egy

$$\kappa_0(\omega) \to \kappa_1 \to \kappa_2 \to \ldots \to \kappa_\ell$$

konfigurációsorozatot kapunk, ahol

$$\ell := \min\{n \mid T \text{ állapota } \kappa_n\text{-ben } ELFOGAD \text{ vagy } ELVET\}$$

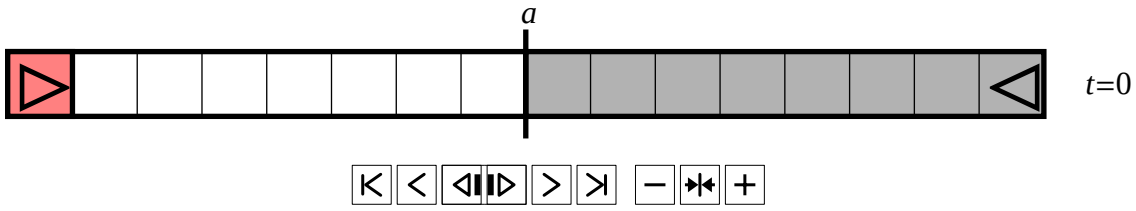
az időpont, amelyben eldöntjük, hogy ω eleme-e a *PALINDROM* nyelvnek. Ez az időpont biztosan véges, hiszen a *PALINDROM* nyelv az eldönthető nyelvek osztályába tartozik (ezt igazolják a korábban említett T_1, T_2 algoritmusok).

Most vegyük azokat a κ_j, κ_{j+1} konfigurációkat, amelyekben az inputszem egyszer az M_i, másszor az M_{i+1} mezőt nézi. Jelölje s_j a mezőket elválasztó a ajtón történő átlépéskor a Turing-gép állapotát. Ezen s_j állapotok sorozatát $\sigma(a,\omega)$ jelöli. Szemléletesen úgy képzelhetjük a $\sigma(a,\omega)$ sorozatot, hogy az a ajtóban egy őr áll, amely a fej minden áthaladásakor feljegyzi annak állapotát.

Nyilvánvaló, hogy a $\sigma(a,\omega)$ sorozatból kiolvasható az ajtón való áthaladás iránya, hiszen a fej balról érkezik, ezért a páratlan sorszámú állapotok a balról-jobbra($\rightarrow$) történő átlépéskor, míg a páros sorszámú állapotok a jobbról-balra($\leftarrow$) történő átlépéskor kerülnek feljegyzésre.

Jelölje $\omega\overset{a}{|}$ az ω input a ajtó előtti (tőle balra található) részét és $\overset{a}{|}\omega$ az ω input a ajtó utáni (tőle jobbra található) részét. Ha például az a ajtó a 6. ábrának megfelelően helyezkedik el, akkor $\omega\overset{a}{|} = \omega_1 \dots \omega_i$ és $\overset{a}{|}\omega = \omega_{i+1} \dots \omega_n$. Természetesen a két rész kiadja a teljes ω inputot: $\omega = \left(\omega\overset{a}{|}\right)\left(\overset{a}{|}\omega\right)$.

Észrevétel. Ha ismerjük az $\omega\overset{a}{|}$ inputrészletet és a $\sigma(a,\omega)$ állapotsorozatot, akkor meg tudjuk mondani, hogy a Turing-gép „hogyan működik”, amikor a szem az a ajtó előtti mezőket pásztázza. Azt nem tudhatjuk, hogy a szem meddig volt a jobb oldalán, de amint átlépi az ajtót (és amíg a bal oldalon marad) képesek vagyunk T futását leírni. Természetesen ugyanez elmondható az $\overset{a}{|}\omega$ darabra is; ekkor az ajtó jobb oldalán ismerjük T lépéseit. Az alábbi animáció illusztrálja az inputszem mozgását és az ajtó elválasztó szerepét. A megfigyelt mezők pirosak, ha $\omega\overset{a}{|}$ elemei és kékek, ha az $\overset{a}{|}\omega$ részei. A $\sigma(a,\omega)$ állapotsorozat a színváltások előtti állapotokból tevődik össze.



3. Következmény. *Legyenek $\omega, \omega' \in \Sigma^n$ tetszőleges inputok és a egy ajtó. Tegyük fel, hogy $\sigma(a,\omega) = \sigma(a,\omega')$ és a T Turing-gép futásának eredménye megegyezik a két inputon. Ekkor az $\widetilde{\omega} = \left(\omega\overset{a}{|}\right)\left(\overset{a}{|}\omega'\right)$ inputon is ugyanazt számolja ki T. Valójában ennél többet is mondhatunk, hiszen az ajtón történő áthaladások számától függ, hogy melyik inputon (ω-n vagy ω'-n) fejeződik be a futás.*

Definíció. Tegyük fel, hogy $3 \mid n$. Legyen

$$I_0 := \{\alpha 0^{\frac{n}{3}}\overleftarrow{\alpha} : \alpha \in \Sigma^{\frac{n}{3}}\} \subseteq PALINDROM \cap \Sigma^n.$$

Tehát I_0 azon n hosszú palindrom szavakból áll, amelyekben egy $n/3$ hosszú szó és fordítottja $n/3$ nullát fog közre.

Megjegyzés. Mivel egy $\alpha 0^{\frac{n}{3}}\overleftarrow{\alpha} \in I_0$ elemet egyértelműen meghatároz az α szó, ezért $|I_0| = |\Sigma|^{\frac{n}{3}} = |\{0,1\}|^{\frac{n}{3}} = 2^{\frac{n}{3}}$.

4. Következmény. *Legyenek ω, $\omega' \in I_0$ különböző szavak és a egy középső ajtó (azaz valamelyik a középső $n/3$ nullát elválasztó ajtó közül). Ekkor $\sigma(a,\omega) \neq \sigma(a,\omega')$.*

Bizonyítás. Indirekt módon tegyük fel, hogy $\sigma(a,\omega) = \sigma(a,\omega')$. Ekkor az előző következmény alapján, ha mindkét inputon *ELFOGAD* állapottal áll le a Turing-gép, akkor az $\widetilde{\omega} = \left(\omega \overset{a}{|}\right)\left(\overset{a}{|}\, \omega'\right) = \alpha 0^{\frac{n}{3}}\overleftarrow{\alpha'}$ inputot is elfogadja, vagyis $\widetilde{\omega} \in I_0$. Azonban $\omega \neq \omega'$, így $\alpha \neq \alpha'$, amiből $\widetilde{\omega} \notin I_0$ következik. Ez az ellentmondás igazolja az állítást. ■

Észrevétel. Ha feltételezzük, hogy $|\Sigma| \geq 2$ és $|S| \geq 3$, akkor a t-nél rövidebb állapotsorozatok számára az

$$1 + |S| + \cdots + |S|^{t-1} = \frac{|S|^t - 1}{|S| - 1} < |S|^t - 1 < |S|^t$$

felső becslés adható.

5. Következmény. *Ha $|I_0| = |\Sigma|^{\frac{n}{3}} \geq |S|^t$, akkor $\exists\, \omega \in I_0$, hogy $\sigma(a,\omega)$ hossza legalább t, ahol a egy középső ajtó.*

Bizonyítás. Indirekt módon tegyük fel, hogy nincs ilyen ω. Ekkor bármely $\omega \in I_0$ és a középső ajtó esetén $\sigma(a,\omega)$ hossza kisebb, mint t. Ez $|I_0| > |S|^t$ állapotsorozatot jelent. Azonban a t-nél rövidebb állapotsorozatok számára adott becslés alapján, mint $|I_0| < |S|^t$. Ez ellentmondás, tehát az állításban szereplő ω input létezik. ■

Megjegyzés. Az előbbi következményből az is leolvasható, hogy $|I_0| = |\Sigma|^{\frac{n}{3}} \geq |S|^t$ esetén $t \sim \beta_T \cdot n$.

6. Következmény. *Ha $|I_0| \geq 2|S|^t$, akkor létezik legalább $|I_0|/2$ olyan szó I_0-ban, amelyre $|\sigma(a,\omega)| \geq t$ és mindegyik különböző állapotsorozatot ad, amikor kiszámítjuk. Ezek halmazát jelölje I_1. Tehát*

$$I_1 = \{\omega \in I_0 : |\sigma(a,\omega)| \geq t\} \subseteq I_0.$$

Az előbbi megjegyzés alapján $t \sim \gamma_T \cdot n$, alkalmas γ_T konstansra.

Ezek után térjünk rá az 1. Tétel bizonyítására

Bizonyítás. Azok az időpontok nyilván nem relevánsak, amikor a fej nem mozdul,

ezért az alábbi alsó becslés adható

$$\sum_{\omega\in I_0} \mathit{TIME}(\omega,T) \geq \sum_{\omega\in I_0} \sum_{\substack{a \text{ középső} \\ \text{ajtó}}} |\{t : t \text{ időpillanatban a fej átlép } a\text{-n}\}|$$

az összegben megjelenő halmaz számossága $\sigma(a,\omega)$ definíciója alapján $|\sigma(a,\omega)|$,

$$= \sum_{\omega\in I_0} \sum_{\substack{a \text{ középső} \\ \text{ajtó}}} |\sigma(a,\omega)| = \sum_{\substack{a \text{ középső} \\ \text{ajtó}}} \sum_{\omega\in I_0} |\sigma(a,\omega)|$$

ahol kettős összegben a szokásos sorrendcsere történt. Az 5. Következmény alapján

$$\geq \sum_{\substack{a \text{ középső} \\ \text{ajtó}}} \sum_{\omega\in I_1} t = \sum_{\substack{a \text{ középső} \\ \text{ajtó}}} |I_1| \cdot t \geq \sum_{\substack{a \text{ középső} \\ \text{ajtó}}} \frac{|I_0|}{2} \cdot t$$

mivel a középső ajtók száma $n/3$, ezért ez

$$= \frac{n}{3} \cdot \frac{|I_0|}{2} \cdot t = \frac{\gamma_T}{6} \cdot n^2 \cdot |I_0|$$

ahonnan $|I_0|$-val való osztás után

$$\frac{1}{|I_0|} \sum_{\omega\in I_0} \mathit{TIME}(\omega,T) \geq \frac{\gamma_T}{6} \cdot n^2$$

adódik. Tehát azt kaptuk, hogy az átlagos futásidő négyzetesen függ az input hosszától. A γ_T konstans függ a Turing-géptől, ezáltal értéke csökkenthető, de a négyzetes jelleg megmarad. ■