

Algoritmuselmélet 6. előadás

Katona Gyula Y.
Budapesti Műszaki és Gazdaságtudományi Egyetem
Számítástudományi Tsz.
I. B. 137/b
kiskat@cs.bme.hu

2002 Március 4.

Hash-elés álvéletlen próbával

A $0, h_1(K), h_2(K), \dots, h_{M-1}(K)$ próbasorozat a $0, 1, \dots, M - 1$ számoknak egy a K kulcstól független álvéletlen permutációja.

Hash-elés álvéletlen próbával

A $0, h_1(K), h_2(K), \dots, h_{M-1}(K)$ próbasorozat a $0, 1, \dots, M - 1$ számoknak egy a K kulcstól független álvéletlen permutációja.

A sorozatnak gyorsan és hatékonyan reprodukálhatónak kell lennie \iff véletlen

Hash-elés álvéletlen próbával

A $0, h_1(K), h_2(K), \dots, h_{M-1}(K)$ próbasorozat a $0, 1, \dots, M - 1$ számoknak egy a K kulcstól független álvéletlen permutációja.

A sorozatnak gyorsan és hatékonyan reprodukálhatónak kell lennie \iff véletlen

Ha $h(K) = h(L)$, akkor a K és L kulcsok teljes próbasorozata is megegyezik \implies másodlagos csomósodásnak

Hash-elés álvéletlen próbával

A $0, h_1(K), h_2(K), \dots, h_{M-1}(K)$ próbasorozat a $0, 1, \dots, M - 1$ számoknak egy a K kulcstól független álvéletlen permutációja.

A sorozatnak gyorsan és hatékonyan reprodukálhatónak kell lennie \iff véletlen

Ha $h(K) = h(L)$, akkor a K és L kulcsok teljes próbasorozata is megegyezik \implies másodlagos csomósodásnak

Kvadratikus maradék próba

Legyen M egy $4k + 3$ alakú prímszám, ahol k egy egész.

Hash-elés álvéletlen próbával

A $0, h_1(K), h_2(K), \dots, h_{M-1}(K)$ próbasorozat a $0, 1, \dots, M - 1$ számoknak egy a K kulcstól független álvéletlen permutációja.

A sorozatnak gyorsan és hatékonyan reprodukálhatónak kell lennie \iff véletlen

Ha $h(K) = h(L)$, akkor a K és L kulcsok teljes próbasorozata is megegyezik \implies másodlagos csomósodásnak

Kvadratikus maradék próba

Legyen M egy $4k + 3$ alakú prímszám, ahol k egy egész.
Ekkor a próbasorozat legyen

$$0, 1^2, -(1^2), 2^2, -(2^2), \dots, \left(\frac{M-1}{2}\right)^2, -\left(\frac{M-1}{2}\right)^2.$$

Hash-elés álvéletlen próbával

A $0, h_1(K), h_2(K), \dots, h_{M-1}(K)$ próbasorozat a $0, 1, \dots, M - 1$ számoknak egy a K kulcstól független álvéletlen permutációja.

A sorozatnak gyorsan és hatékonyan reprodukálhatónak kell lennie \iff véletlen

Ha $h(K) = h(L)$, akkor a K és L kulcsok teljes próbasorozata is megegyezik \implies másodlagos csomósodásnak

Kvadratikus maradék próba

Legyen M egy $4k + 3$ alakú prímszám, ahol k egy egész.
Ekkor a próbasorozat legyen

$$0, 1^2, -(1^2), 2^2, -(2^2), \dots, \left(\frac{M-1}{2}\right)^2, -\left(\frac{M-1}{2}\right)^2.$$

\implies Belátjuk, hogy ez tényleg permutáció.

Tétel. *Ha M egy $4k + 3$ alakú prímszám, akkor nincs olyan n egész, melyre $n^2 \equiv -1 \pmod{M}$.*

Tétel. *Ha M egy $4k + 3$ alakú prímszám, akkor nincs olyan n egész, melyre $n^2 \equiv -1 \pmod{M}$.*

Bizonyítás: Indirekt tegyük fel, hogy n egy egész szám és $n^2 \equiv -1 \pmod{M}$.

Tétel. Ha M egy $4k + 3$ alakú prímszám, akkor nincs olyan n egész, melyre $n^2 \equiv -1 \pmod{M}$.

Bizonyítás: Indirekt tegyük fel, hogy n egy egész szám és $n^2 \equiv -1 \pmod{M}$. \implies

$$-1 \equiv (-1)^{\frac{M-1}{2}} \equiv n^{2\frac{M-1}{2}} \equiv n^{M-1} \equiv 1 \pmod{M}.$$

Tétel. Ha M egy $4k + 3$ alakú prímszám, akkor nincs olyan n egész, melyre $n^2 \equiv -1 \pmod{M}$.

Bizonyítás: Indirekt tegyük fel, hogy n egy egész szám és $n^2 \equiv -1 \pmod{M}$. \implies

$$-1 \equiv (-1)^{\frac{M-1}{2}} \equiv n^{2\frac{M-1}{2}} \equiv n^{M-1} \equiv 1 \pmod{M}.$$

Az utolsó lépésnél a kis Fermat-tételt használtuk. ⚡

Tétel. Ha M egy $4k + 3$ alakú prímszám, akkor nincs olyan n egész, melyre $n^2 \equiv -1 \pmod{M}$.

Bizonyítás: Indirekt tegyük fel, hogy n egy egész szám és $n^2 \equiv -1 \pmod{M}$. \implies

$$-1 \equiv (-1)^{\frac{M-1}{2}} \equiv n^{2\frac{M-1}{2}} \equiv n^{M-1} \equiv 1 \pmod{M}.$$

Az utolsó lépésnél a kis Fermat-tételt használtuk. ⚡

Ha $0 \leq i < j \leq \frac{M-1}{2}$, akkor $i^2 \not\equiv j^2 \pmod{M}$.

$\Leftarrow j^2 - i^2 = (j - i)(j + i)$ felbontás egyik tényezője sem lehet osztható M -mel, tehát a szorzatuk sem

Tétel. Ha M egy $4k + 3$ alakú prímszám, akkor nincs olyan n egész, melyre $n^2 \equiv -1 \pmod{M}$.

Bizonyítás: Indirekt tegyük fel, hogy n egy egész szám és $n^2 \equiv -1 \pmod{M}$. \implies

$$-1 \equiv (-1)^{\frac{M-1}{2}} \equiv n^{2\frac{M-1}{2}} \equiv n^{M-1} \equiv 1 \pmod{M}.$$

Az utolsó lépésnél a kis Fermat-tételt használtuk. ⚡

Ha $0 \leq i < j \leq \frac{M-1}{2}$, akkor $i^2 \not\equiv j^2 \pmod{M}$.

$\iff j^2 - i^2 = (j - i)(j + i)$ felbontás egyik tényezője sem lehet osztható M -mel, tehát a szorzatuk sem

Ugyanígy $\implies -i^2 \not\equiv -j^2 \pmod{M}$.

Tétel. Ha M egy $4k + 3$ alakú prímszám, akkor nincs olyan n egész, melyre $n^2 \equiv -1 \pmod{M}$.

Bizonyítás: Indirekt tegyük fel, hogy n egy egész szám és $n^2 \equiv -1 \pmod{M}$. \implies

$$-1 \equiv (-1)^{\frac{M-1}{2}} \equiv n^{2\frac{M-1}{2}} \equiv n^{M-1} \equiv 1 \pmod{M}.$$

Az utolsó lépésnél a kis Fermat-tételt használtuk. ⚡

Ha $0 \leq i < j \leq \frac{M-1}{2}$, akkor $i^2 \not\equiv j^2 \pmod{M}$.

$\iff j^2 - i^2 = (j - i)(j + i)$ felbontás egyik tényezője sem lehet osztható M -mel, tehát a szorzatuk sem

Ugyanígy $\implies -i^2 \not\equiv -j^2 \pmod{M}$.

$i^2 \not\equiv -j^2 \pmod{M} \iff (ij^{-1})^2 \not\equiv -1 \pmod{M}$ ✓

Sikeres keresés költsége:

$$C_N \approx 1 - \log(1 - \alpha) - \frac{\alpha}{2}$$

Sikertelen keresés költsége:

$$C'_N \approx \frac{1}{(1 - \alpha)} - \alpha - \log(1 - \alpha)$$

Sikeres keresés költsége:

$$C_N \approx 1 - \log(1 - \alpha) - \frac{\alpha}{2}$$

Sikertelen keresés költsége:

$$C'_N \approx \frac{1}{(1 - \alpha)} - \alpha - \log(1 - \alpha)$$

Ezek az összefüggések valamivel általánosabban érvényesek az olyan módszerekre, amelyekre $h_i(K) = f_i(h(K))$; vagyis ahol a $h(K)$ érték már az egész próbasorozatot meghatározza.

Kettős hash-elés

G. de Balbine, J. R. Bell, C. H. Kaman, 1970 körül.

Kettős hash-elés

G. de Balbine, J. R. Bell, C. H. Kaman, 1970 körül.

Lényeg: h mellett egy másik h' hash-függvényt is használunk

Kettős hash-elés

G. de Balbine, J. R. Bell, C. H. Kaman, 1970 körül.

Lényeg: h mellett egy másik h' hash-függvényt is használunk a $h'(K)$ értékek relatív prímek legyenek az M táblamérethez

Kettős hash-elés

G. de Balbine, J. R. Bell, C. H. Kaman, 1970 körül.

Lényeg: h mellett egy másik h' hash-függvényt is használunk
a $h'(K)$ értékek relatív prímek legyenek az M táblamérethez
A K kulcs próbasorozata: $h_i(K) := -ih'(K)$.

Kettős hash-elés

G. de Balbine, J. R. Bell, C. H. Kaman, 1970 körül.

Lényeg: h mellett egy másik h' hash-függvényt is használunk a $h'(K)$ értékek relatív prímek legyenek az M táblamérethez

A K kulcs próbasorozata: $h_i(K) := -ih'(K)$.

Ha M és $h'(K)$ relatív prímek

$\implies 0, -h'(K), -2h'(K), \dots, -(M-1)h'(K)$ sorozat elemei mind különbözők modulo M

Kettős hash-elés

G. de Balbine, J. R. Bell, C. H. Kaman, 1970 körül.

Lényeg: h mellett egy másik h' hash-függvényt is használunk a $h'(K)$ értékek relatív prímek legyenek az M táblamérethez

A K kulcs próbasorozata: $h_i(K) := -ih'(K)$.

Ha M és $h'(K)$ relatív prímek

$\implies 0, -h'(K), -2h'(K), \dots, -(M-1)h'(K)$ sorozat elemei mind különbözők modulo M

Fontos sajátossága: különböző K és K' kulcsok próbasorozatai jó eséllyel akkor is különbözők lesznek, ha $h(K) = h(K')$.

Kettős hash-elés

G. de Balbine, J. R. Bell, C. H. Kaman, 1970 körül.

Lényeg: h mellett egy másik h' hash-függvényt is használunk a $h'(K)$ értékek relatív prímek legyenek az M táblamérethez

A K kulcs próbasorozata: $h_i(K) := -ih'(K)$.

Ha M és $h'(K)$ relatív prímek

$\implies 0, -h'(K), -2h'(K), \dots, -(M-1)h'(K)$ sorozat elemei mind különbözők modulo M

Fontos sajátossága: különböző K és K' kulcsok próbasorozatai jó eséllyel akkor is különbözők lesznek, ha $h(K) = h(K')$.

A legjobb ismert implementációk időigénye (empirikus adatok alapján)

$$C_N \approx \frac{1}{\alpha} \log \frac{1}{(1-\alpha)} \quad \text{és} \quad C'_N \approx \frac{1}{1-\alpha}.$$

- A kettős hash-elés kiküszöböli mindkét-féle csomósodást

- A kettős hash-elés kiküszöböli mindkét-féle csomósodást
- Sikertelen keresés esetén minden érdekes α -ra gyorsabb, mint a lineáris próbálás

- A kettős hash-elés kiküszöböli mindkét-féle csomósodást
- Sikertelen keresés esetén minden érdekes α -ra gyorsabb, mint a lineáris próbálás
- Sikeres kereséskor csak az $\alpha \geq 0,8$ tartományban lesz gyorsabb a lineáris próbálásnál.

Hash-függvények

- legyen könnyen (gyorsan) számítható

Hash-függvények

- legyen könnyen (gyorsan) számítható
- és minél kevesebb ütközést okozzon.

Hash-függvények

- legyen könnyen (gyorsan) számítható
- és minél kevesebb ütközést okozzon.

A második követelmény elég nehezen megfogható, mert a gyakorlatban előforduló kulcshalmazok egyáltalán nem véletlenszerűek.

Hash-függvények

- legyen könnyen (gyorsan) számítható
- és minél kevesebb ütközést okozzon.

A második követelmény elég nehezen megfogható, mert a gyakorlatban előforduló kulcshalmazok egyáltalán nem véletlenszerűek.

hasznos tanácsok $\implies h(K)$ értéke lehetőleg a K kulcs minden bitjétől függjön

Hash-függvények

- legyen könnyen (gyorsan) számítható
- és minél kevesebb ütközést okozzon.

A második követelmény elég nehezen megfogható, mert a gyakorlatban előforduló kulcshalmazok egyáltalán nem véletlenszerűek.

hasznos tanácsok $\implies h(K)$ értéke lehetőleg a K kulcs minden bitjétől függjön és a h értékkészlete a teljes $[0, M - 1]$ címtartomány legyen

Osztómódszer

Legyen $h(K) := K \pmod{M}$,
ahol M a tábla vagy a vödörkatalógus mérete.

Osztómódszer

Legyen $h(K) := K \pmod{M}$,
ahol M a tábla vagy a vödörkatalógus mérete.
Feltesszük, hogy a kulcsok egész számok.

Osztómódszer

Legyen $h(K) := K \pmod{M}$,
ahol M a tábla vagy a vödörkatalógus mérete.
Feltesszük, hogy a kulcsok egész számok.
A $h(K)$ számítása gyors és egyszerű.

Osztómódszer

Legyen $h(K) := K \pmod{M}$,
ahol M a tábla vagy a vödörkatalógus mérete.
Feltesszük, hogy a kulcsok egész számok.
A $h(K)$ számítása gyors és egyszerű.

A tábla mérete sem teljesen közömbös.

Például ha M a 2 egy hatványa, akkor $h(K)$ csak a kulcs utolsó néhány bitjétől függ.

Osztómódszer

Legyen $h(K) := K \pmod{M}$,
ahol M a tábla vagy a vödörkatalógus mérete.
Feltesszük, hogy a kulcsok egész számok.
A $h(K)$ számítása gyors és egyszerű.

A tábla mérete sem teljesen közömbös.

Például ha M a 2 egy hatványa, akkor $h(K)$ csak a kulcs utolsó néhány bitjétől függ.

A jó M értékeket illetően van egy széles körben elfogadott recept:

D. E. Knuth javaslata \implies M -et prímmek választjuk, úgy, hogy M nem osztja $r^k + a$ -t, ahol r a karakterkészlet elemszáma (pl. 128, vagy 256) és a , k „kicsi” egészek.

Osztómódszer

Legyen $h(K) := K \pmod{M}$,
ahol M a tábla vagy a vödörkatalógus mérete.
Feltesszük, hogy a kulcsok egész számok.
A $h(K)$ számítása gyors és egyszerű.

A tábla mérete sem teljesen közömbös.

Például ha M a 2 egy hatványa, akkor $h(K)$ csak a kulcs utolsó néhány bitjétől függ.

A jó M értékeket illetően van egy széles körben elfogadott recept:

D. E. Knuth javaslata $\implies M$ -et prímmek választjuk, úgy, hogy M nem osztja $r^k + a$ -t, ahol r a karakterkészlet elemszáma (pl. 128, vagy 256) és a , k „kicsi” egészek.

M prímmel \implies lényeges feltétel a kvadratikus maradék próbánál.

Osztómódszer

Legyen $h(K) := K \pmod{M}$,
ahol M a tábla vagy a vödörkatalógus mérete.
Feltesszük, hogy a kulcsok egész számok.
A $h(K)$ számítása gyors és egyszerű.

A tábla mérete sem teljesen közömbös.

Például ha M a 2 egy hatványa, akkor $h(K)$ csak a kulcs utolsó néhány bitjétől függ.

A jó M értékeket illetően van egy széles körben elfogadott recept:

D. E. Knuth javaslata $\implies M$ -et prímmek választjuk, úgy, hogy M nem osztja $r^k + a$ -t, ahol r a karakterkészlet elemszáma (pl. 128, vagy 256) és a , k „kicsi” egészek.

M prímmel \implies lényeges feltétel a kvadratikus maradék próbánál.

Könnyű hozzájuk relatív prímmel számot találni \implies kettős hash-elés

Szorzmódszer

β egy rögzített paraméter

Szorzmódszer

β egy rögzített paraméter

$$h(K) := \lfloor M \cdot \{\beta K\} \rfloor.$$

$\{x\}$ jelöli az x valós szám törtrészét

Szorzó módszer

β egy rögzített paraméter

$$h(K) := \lfloor M \cdot \{\beta K\} \rfloor.$$

$\{x\}$ jelöli az x valós szám törtrészét

Szemléletesen $\implies \{\beta K\}$ kiszámításával a K kulcsot „véletlenszerűen” belőjük a $[0, 1)$ intervallumba \implies az eredményt felskálázzuk a címtartományba.

Szorzó módszer

β egy rögzített paraméter

$$h(K) := \lfloor M \cdot \{\beta K\} \rfloor.$$

$\{x\}$ jelöli az x valós szám törtrészét

Szemléletesen $\implies \{\beta K\}$ kiszámításával a K kulcsot „véletlenszerűen” belőjük a $[0, 1)$ intervallumba \implies az eredményt felskálázzuk a címtartományba.

Hatékonyan számítható speciális eset:

$M = 2^t$, $w = 2^{32}$, és legyen A egy a w -hez relatív prím egész.

Ekkor $\beta = \frac{A}{w}$ választás mellett $h(K)$ igen jól számolható.

A számok bináris ábrázolásával dolgozva lényegében egy szorzást és egy eltolást kell elvégezni.

A szorzómódszer jól viselkedik számtani sorozatokon

A szorzómódszer jól viselkedik számtani sorozatokon
pl. $termék_1$, $termék_2$, $termék_3$, ... esetében

A szorzómódszer jól viselkedik számtani sorozatokon

pl. $termék_1, termék_2, termék_3, \dots$ esetében

Megmutatható, hogy a $h(K), h(K + d), h(K + 2d) \dots$ sorozat közelítőleg számtani sorozat lesz, azaz h jól „szétdobja” a kulcsok számtani sorozatait.

A szorzómódszer jól viselkedik számtani sorozatokon

pl. $\text{termék}_1, \text{termék}_2, \text{termék}_3, \dots$ esetében

Megmutatható, hogy a $h(K), h(K + d), h(K + 2d) \dots$ sorozat közelítőleg számtani sorozat lesz, azaz h jól „szétdobja” a kulcsok számtani sorozatait.

Tétel (T. Sós Vera, 1957). *Legyen β irracionális szám, és nézzük a $0, \{\beta\}, \{2\beta\}, \dots, \{n\beta\}$ pontok által meghatározott $n + 1$ részintervallumot $[0, 1)$ -ben. Ezek hosszai legfeljebb 3 különböző értéket vehetnek fel, és $\{(n + 1)\beta\}$ a leghosszabbak egyikét fogja két részre vágni.*

A szorzómódszer jól viselkedik számtani sorozatokon

pl. $\text{termék}_1, \text{termék}_2, \text{termék}_3, \dots$ esetében

Megmutatható, hogy a $h(K), h(K + d), h(K + 2d) \dots$ sorozat közelítőleg számtani sorozat lesz, azaz h jól „szétdobja” a kulcsok számtani sorozatait.

Tétel (T. Sós Vera, 1957). *Legyen β irracionális szám, és nézzük a $0, \{\beta\}, \{2\beta\}, \dots, \{n\beta\}$ pontok által meghatározott $n + 1$ részintervallumot $[0, 1)$ -ben. Ezek hosszai legfeljebb 3 különböző értéket vehetnek fel, és $\{(n + 1)\beta\}$ a leghosszabbak egyikét fogja két részre vágni.*

a $[0, 1)$ -beli számok közül a legegyenletesebb eloszlást a

$\beta = \phi^{-1} = \frac{\sqrt{5}-1}{2} = 0.618033988 \dots$ és a $\beta = \phi^{-2} = 1 - \phi^{-1}$ értékek adják.

A szorzómódszer jól viselkedik számtani sorozatokon

pl. $\text{termék}_1, \text{termék}_2, \text{termék}_3, \dots$ esetében

Megmutatható, hogy a $h(K), h(K + d), h(K + 2d) \dots$ sorozat közelítőleg számtani sorozat lesz, azaz h jól „szétdobja” a kulcsok számtani sorozatait.

Tétel (T. Sós Vera, 1957). *Legyen β irracionális szám, és nézzük a $0, \{\beta\}, \{2\beta\}, \dots, \{n\beta\}$ pontok által meghatározott $n + 1$ részintervallumot $[0, 1)$ -ben. Ezek hosszai legfeljebb 3 különböző értéket vehetnek fel, és $\{(n + 1)\beta\}$ a leghosszabbak egyikét fogja két részre vágni.*

a $[0, 1)$ -beli számok közül a legegyszerűsebb eloszlást a

$\beta = \phi^{-1} = \frac{\sqrt{5}-1}{2} = 0.618033988 \dots$ és a $\beta = \phi^{-2} = 1 - \phi^{-1}$ értékek adják.

\implies érdemes a szorzómódszernél az A -t úgy választani, hogy $\frac{A}{w}$ közel legyen ϕ^{-1} -hez. \implies *Fibonacci-hash-elés*

A kettős hash-elés második függvénye

Olyan h' függvény kell, melynek értékei a $[0, M - 1]$ intervallumba esnek, és relatív prímek az M -hez

A kettős hash-elés második függvénye

Olyan h' függvény kell, melynek értékei a $[0, M - 1]$ intervallumba esnek, és relatív prímek az M -hez

Ha M prím \implies

$$h'(K) := K \pmod{M - 1} + 1.$$

A kettős hash-elés második függvénye

Olyan h' függvény kell, melynek értékei a $[0, M - 1]$ intervallumba esnek, és relatív prímek az M -hez

Ha M prím \implies

$$h'(K) := K \pmod{M - 1} + 1.$$

$\implies h'(K)$ és M relatív prímek

A kettős hash-elés második függvénye

Olyan h' függvény kell, melynek értékei a $[0, M - 1]$ intervallumba esnek, és relatív prímek az M -hez

Ha M prím \implies

$$h'(K) := K \pmod{M - 1} + 1.$$

$\implies h'(K)$ és M relatív prímek

\implies elég sok különböző próbasorozatot ad

Java animáció: Hash-elés

Szekvenciális keresés

Ha egy állomány (tömb, lista, stb.) szegényes szerkezetű \implies nincs jobb , mint „elejétől a végéig” bejárni, vagy legalábbis addig, amíg a keresett adatot meg nem találjuk.

Szekvenciális keresés

Ha egy állomány (tömb, lista, stb.) szegényes szerkezetű \implies nincs jobb, mint „elejétől a végéig” bejárni, vagy legalábbis addig, amíg a keresett adatot meg nem találjuk.

Ha egyenlő eséllyel kell keresni az elemeket \implies

Sikeres keresés átlagos költsége:

$$\frac{1 + 2 + \dots + N}{N} = \frac{N + 1}{2}.$$

Szekvenciális keresés

Ha egy állomány (tömb, lista, stb.) szegényes szerkezetű \implies nincs jobb, mint „elejétől a végéig” bejárni, vagy legalábbis addig, amíg a keresett adatot meg nem találjuk.

Ha egyenlő eséllyel kell keresni az elemeket \implies

Sikeres keresés átlagos költsége:

$$\frac{1 + 2 + \dots + N}{N} = \frac{N + 1}{2}.$$

Sikertelen keresés költsége: N

Szekvenciális keresés

Ha egy állomány (tömb, lista, stb.) szegényes szerkezetű \implies nincs jobb, mint „elejétől a végéig” bejárni, vagy legalábbis addig, amíg a keresett adatot meg nem találjuk.

Ha egyenlő eséllyel kell keresni az elemeket \implies

Sikeres keresés átlagos költsége:

$$\frac{1 + 2 + \dots + N}{N} = \frac{N + 1}{2}.$$

Sikertelen keresés költsége: N

Ha az állományban az elemek nagyság szerint rendezettek \implies

Sikeres keresés átlagos költsége: $\frac{N+1}{2}$.

Szekvenciális keresés

Ha egy állomány (tömb, lista, stb.) szegényes szerkezetű \implies nincs jobb, mint „elejétől a végéig” bejárni, vagy legalábbis addig, amíg a keresett adatot meg nem találjuk.

Ha egyenlő eséllyel kell keresni az elemeket \implies

Sikeres keresés átlagos költsége:

$$\frac{1 + 2 + \dots + N}{N} = \frac{N + 1}{2}.$$

Sikertelen keresés költsége: N

Ha az állományban az elemek nagyság szerint rendezettek \implies

Sikeres keresés átlagos költsége: $\frac{N+1}{2}$.

Sikertelen keresés költsége:

$$\frac{1 + 2 + \dots + N - 1 + N + N}{N + 1} = \frac{N}{2} + \frac{N}{N + 1} < \frac{N}{2} + 1.$$

Tegyük fel, hogy csak sikeres keresésekkel van dolgunk, és legyen p_i annak a valószínűsége, hogy az R_i rekordot keressük.

Tegyük fel, hogy csak sikeres keresésekkel van dolgunk, és legyen p_i annak a valószínűsége, hogy az R_i rekordot keressük. \implies

Sikeres keresés átlagos költsége:

$$C_N = p_1 + 2p_2 + 3p_3 + \dots + Np_N.$$

Hogy érdemes sorba rendezni az R_i rekordokat?

Tegyük fel, hogy csak sikeres keresésekkel van dolgunk, és legyen p_i annak a valószínűsége, hogy az R_i rekordot keressük. \implies

Sikeres keresés átlagos költsége:

$$C_N = p_1 + 2p_2 + 3p_3 + \dots + Np_N.$$

Hogy érdemes sorba rendezni az R_i rekordokat? \implies
csökkenő sorrendben

Tegyük fel, hogy csak sikeres keresésekkel van dolgunk, és legyen p_i annak a valószínűsége, hogy az R_i rekordot keressük. \implies

Sikeres keresés átlagos költsége:

$$C_N = p_1 + 2p_2 + 3p_3 + \dots + Np_N.$$

Hogy érdemes sorba rendezni az R_i rekordokat? \implies
csökkenő sorrendben

Különböző eloszlások esetén:

Egyenletes: $p_i = \frac{1}{N} \implies C_N = \frac{N+1}{2}$

Tegyük fel, hogy csak sikeres keresésekkel van dolgunk, és legyen p_i annak a valószínűsége, hogy az R_i rekordot keressük. \implies

Sikeres keresés átlagos költsége:

$$C_N = p_1 + 2p_2 + 3p_3 + \cdots + Np_N.$$

Hogy érdemes sorba rendezni az R_i rekordokat? \implies
csökkenő sorrendben

Különböző eloszlások esetén:

Egyenletes: $p_i = \frac{1}{N} \implies C_N = \frac{N+1}{2}$

Egy nagyon ferde eloszlás: $p_i = \frac{1}{2^i} \implies C_N = 2 - \frac{1}{2^{N-1}}$

Zipf eloszlás: $p_i = \frac{1}{iH_N}$, ahol

$$H_N = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} = \log n + 0.55721 \dots + o(1)$$

\implies

$$C_N = \sum_{i=1}^N ip_i = \sum_{i=1}^N i \frac{1}{iH_N} = \frac{N}{H_N} \approx \frac{N}{\log N}$$

Zipf eloszlás: $p_i = \frac{1}{iH_N}$, ahol

$$H_N = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} = \log n + 0.55721 \dots + o(1)$$

\implies

$$C_N = \sum_{i=1}^N ip_i = \sum_{i=1}^N i \frac{1}{iH_N} = \frac{N}{H_N} \approx \frac{N}{\log N}$$

80-20 szabály: Tapasztalat \implies Az adatelérési igények 80%-a a rekordoknak körülbelül csak 20%-át érinti.

Zipf eloszlás: $p_i = \frac{1}{iH_N}$, ahol

$$H_N = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} = \log n + 0.55721 \dots + o(1)$$

\implies

$$C_N = \sum_{i=1}^N ip_i = \sum_{i=1}^N i \frac{1}{iH_N} = \frac{N}{H_N} \approx \frac{N}{\log N}$$

80-20 szabály: Tapasztalat \implies Az adatelérési igények 80%-a a rekordoknak körülbelül csak 20%-át érinti.

$$p_i = \frac{c}{i^{1-\vartheta}}, \text{ ahol}$$

$$\vartheta = \frac{\log 0,8}{\log 0,2} \approx \frac{1}{7}, \quad c = \frac{1}{H_N^{(1-\vartheta)}} \quad \text{és} \quad H_N^{(1-\vartheta)} = 1 + \frac{1}{2^{(1-\vartheta)}} + \dots + \frac{1}{N^{(1-\vartheta)}}$$

Zipf eloszlás: $p_i = \frac{1}{iH_N}$, ahol

$$H_N = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} = \log n + 0.55721 \dots + o(1)$$

\implies

$$C_N = \sum_{i=1}^N ip_i = \sum_{i=1}^N i \frac{1}{iH_N} = \frac{N}{H_N} \approx \frac{N}{\log N}$$

80-20 szabály: Tapasztalat \implies Az adatelérési igények 80%-a a rekordoknak körülbelül csak 20%-át érinti.

$$p_i = \frac{c}{i^{1-\vartheta}}, \text{ ahol}$$

$$\vartheta = \frac{\log 0,8}{\log 0,2} \approx \frac{1}{7}, \quad c = \frac{1}{H_N^{(1-\vartheta)}} \quad \text{és} \quad H_N^{(1-\vartheta)} = 1 + \frac{1}{2^{(1-\vartheta)}} + \dots + \frac{1}{N^{(1-\vartheta)}}$$

$$\implies C_N \approx 0,122N$$

Önszervező módszerek

Mit tehetünk abban az esetben, ha a p_i keresési valószínűségeket nem ismerjük, vagy esetleg azok idővel változnak?

Önszervező módszerek

Mit tehetünk abban az esetben, ha a p_i keresési valószínűségeket nem ismerjük, vagy esetleg azok idővel változnak?

- A keresett (és megtalált) R_i elemet a tábla elejére visszük. Az eredmény:

R_i	R_1	R_2	\dots		R_N
-------	-------	-------	---------	--	-------

Önszervező módszerek

Mit tehetünk abban az esetben, ha a p_i keresési valószínűségeket nem ismerjük, vagy esetleg azok idővel változnak?

- A keresett (és megtalált) R_i elemet a tábla elejére visszük. Az eredmény:

R_i	R_1	R_2	\dots		R_N
-------	-------	-------	---------	--	-------

- A keresett (és megtalált) R_i elemet felcseréljük a megelőzővel.

R_1	\dots	R_i	R_{i-1}	\dots		R_N
-------	---------	-------	-----------	---------	--	-------

Önszervező módszerek

Mit tehetünk abban az esetben, ha a p_i keresési valószínűségeket nem ismerjük, vagy esetleg azok idővel változnak?

- A keresett (és megtalált) R_i elemet a tábla elejére visszük. Az eredmény:

R_i	R_1	R_2	\dots		R_N
-------	-------	-------	---------	--	-------

- A keresett (és megtalált) R_i elemet felcseréljük a megelőzővel.

R_1	\dots	R_i	R_{i-1}	\dots		R_N
-------	---------	-------	-----------	---------	--	-------

Ha az eloszlás időben változik, akkor az első megoldás ajánlatos. Ha a $\{p_i\}$ eloszlás stabil, azaz időben nem változik, akkor a második heurisztika eredményesebb.

Információtömörítés

A b_1, \dots, b_n betűkből álló szöveget szeretnénk bitsorozatként kódolni.

Információtömörítés

A b_1, \dots, b_n betűkből álló szöveget szeretnénk bitsorozatként kódolni.

uniform kódolás $\implies \lceil \log_2 n \rceil$ a legrövidebb lehetséges kódhosszúság egy betűre

Információtömörítés

A b_1, \dots, b_n betűkből álló szöveget szeretnénk bitsorozatként kódolni.

uniform kódolás $\implies \lceil \log_2 n \rceil$ a legrövidebb lehetséges kódhosszúság egy betűre

eltérő hosszú kódszavak \implies dekódolás problémásabb

Információtömörítés

A b_1, \dots, b_n betűkből álló szöveget szeretnénk bitsorozatként kódolni.

uniform kódolás $\implies \lceil \log_2 n \rceil$ a legrövidebb lehetséges kódhosszúság egy betűre

eltérő hosszú kódszavak \implies dekódolás problémásabb

Definíció. Egy bitsorozatokból álló kód *prefix kód*, ha egy betű kódja sem *prefixe* (kezdőszelete) egy másik kódjának. *Formálisan: ha x és y két különböző betű kódja, akkor nincs olyan z bitsorozat, melyre $xz = y$.*

Információtömörítés

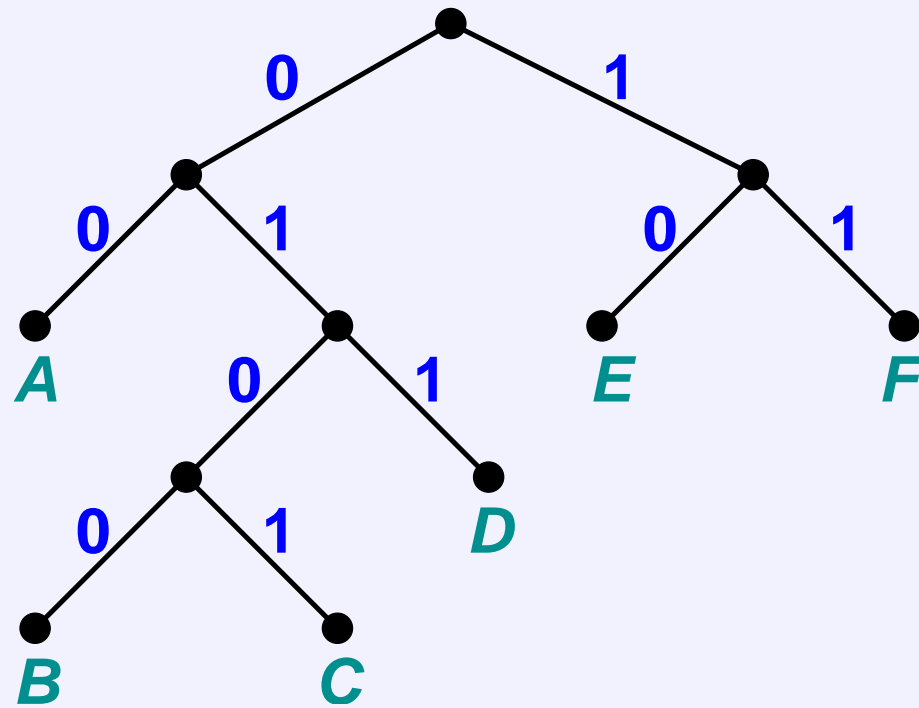
A b_1, \dots, b_n betűkből álló szöveget szeretnénk bitsorozatként kódolni.

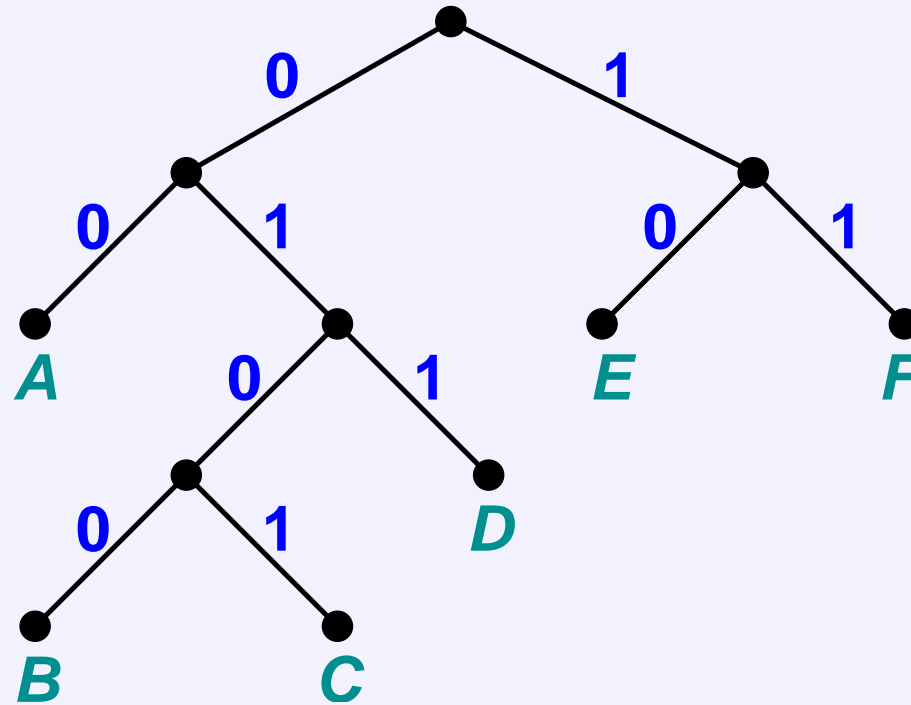
uniform kódolás $\implies \lceil \log_2 n \rceil$ a legrövidebb lehetséges kódhosszúság egy betűre

eltérő hosszú kódszavak \implies dekódolás problémásabb

Definíció. Egy bitsorozatokból álló kód *prefix kód*, ha egy betű kódja sem *prefixe* (kezdőszelete) egy másik kódjának. *Formálisan: ha x és y két különböző betű kódja, akkor nincs olyan z bitsorozat, melyre $xz = y$.*

Egy prefix-tulajdonságú kóddal leírt üzenet egyértelműen visszafejthető.





Probléma: Adott egy szöveg, melyben a b_i karakter q_i -szer fordul elő. Keressünk olyan fát, amelyre a $\sum q_i h(b_i)$ összeg minimális, ahol egy x csúcsra $h(x)$ a gyökértől x -ig vezető úton bejárt élek száma.

Huffman-kód

Optimális ilyen fa:

Huffman-kód

Optimális ilyen fa:

- Kezdetben n izolált csúcspontunk van. b_i címkéje legyen q_i .

Huffman-kód

Optimális ilyen fa:

- Kezdetben n izolált csúcspontunk van. b_i címkéje legyen q_i .
- Tegyük fel, hogy már megépítettük az S_1, \dots, S_k fákat, ezek gyökérpontjai x_1, \dots, x_k , utóbbiak címkéi az r_1, \dots, r_k számok.

Huffman-kód

Optimális ilyen fa:

- Kezdetben n izolált csúcsponunk van. b_i címkéje legyen q_i .
- Tegyük fel, hogy már megépítettük az S_1, \dots, S_k fákat, ezek gyökérpontjai x_1, \dots, x_k , utóbbiak címkéi az r_1, \dots, r_k számok.
Ekkor vesszük a két minimális címkéjű gyökeret (legyenek ezek x_i és x_j).

Huffman-kód

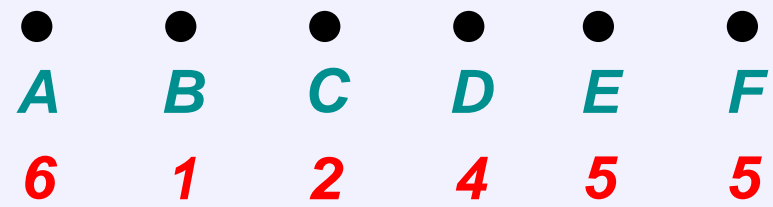
Optimális ilyen fa:

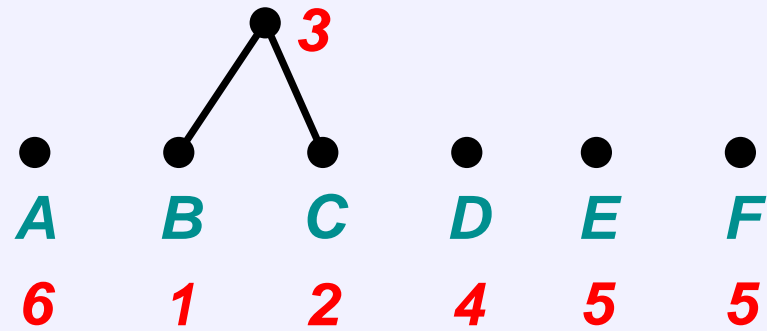
- Kezdetben n izolált csúcspontunk van. b_i címkéje legyen q_i .
- Tegyük fel, hogy már megépítettük az S_1, \dots, S_k fákat, ezek gyökérpontjai x_1, \dots, x_k , utóbbiak címkéi az r_1, \dots, r_k számok.
Ekkor vesszük a két minimális címkéjű gyökeret (legyenek ezek x_i és x_j).
- Ezek fölé egy új y gyökérpontot teszünk, melynek fiai x_i és x_j . Az y címkéje $r_i + r_j$.

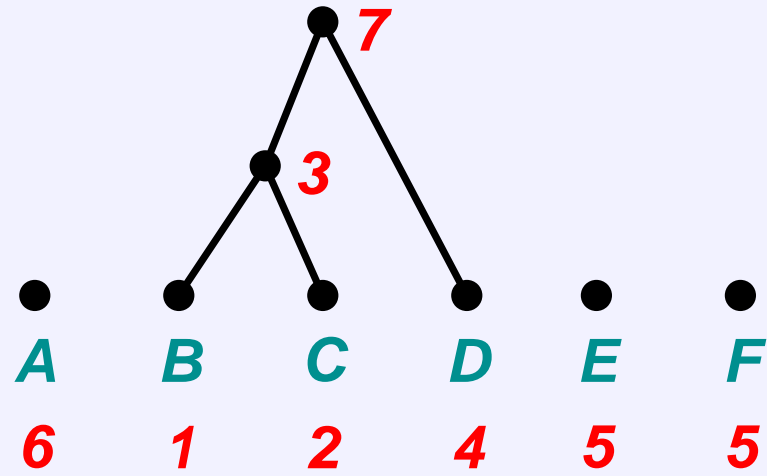
Huffman-kód

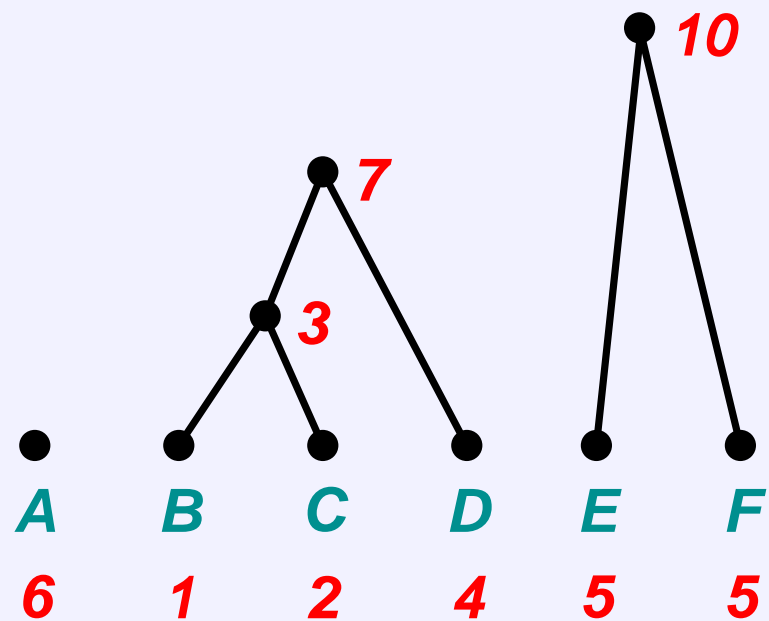
Optimális ilyen fa:

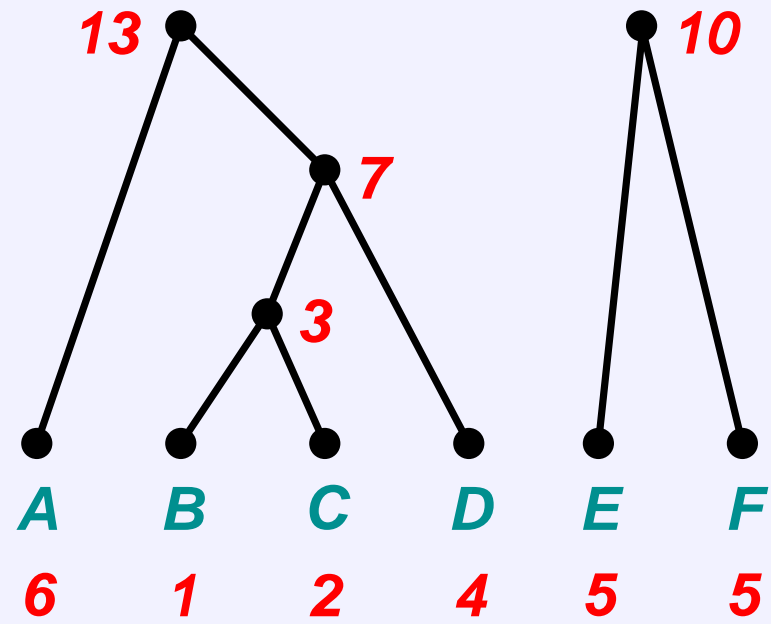
- Kezdetben n izolált csúcspontunk van. b_i címkéje legyen q_i .
- Tegyük fel, hogy már megépítettük az S_1, \dots, S_k fákat, ezek gyökérpontjai x_1, \dots, x_k , utóbbiak címkéi az r_1, \dots, r_k számok.
Ekkor vesszük a két minimális címkéjű gyökeret (legyenek ezek x_i és x_j).
- Ezek fölé egy új y gyökérpontot teszünk, melynek fiai x_i és x_j . Az y címkéje $r_i + r_j$.
- A fák száma eggyel csökken. Megállunk, ha már csak egy fa marad.
Összesen $n - 1$ ilyen összevonó lépés szükséges.

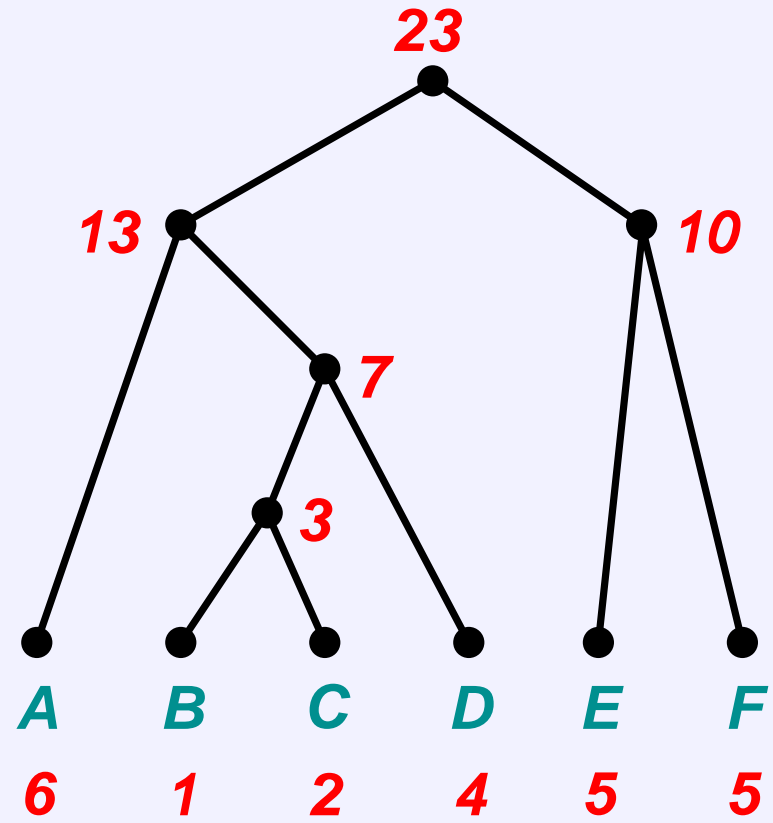


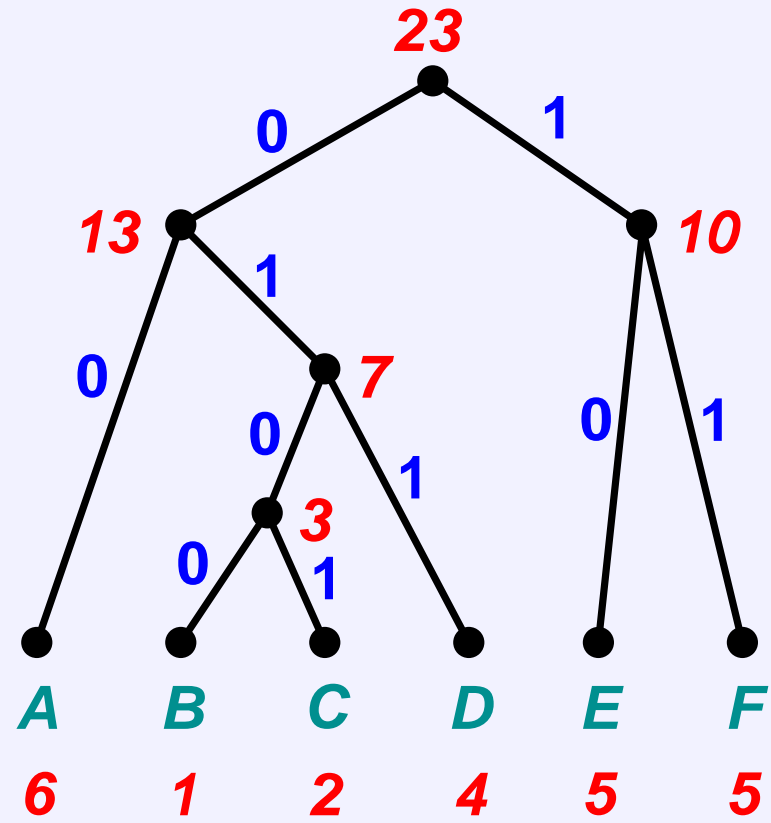










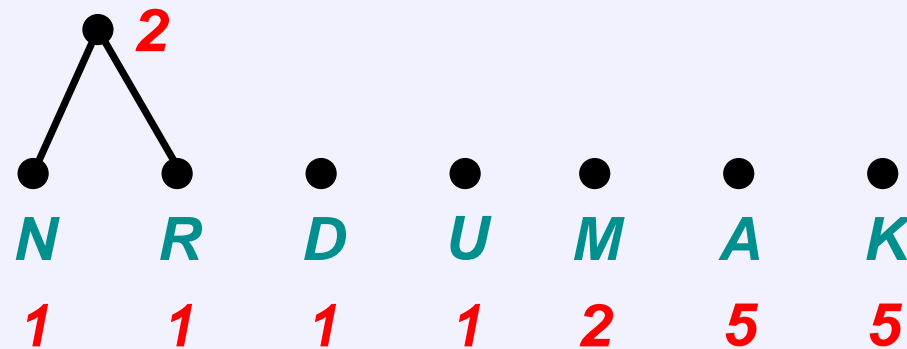


KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.

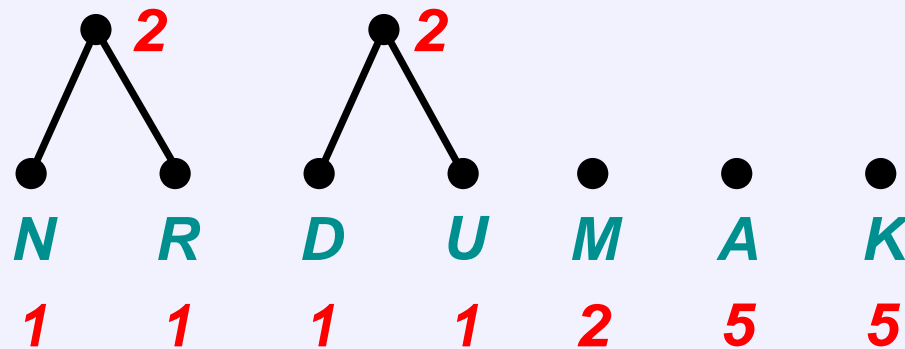
KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.

●	●	●	●	●	●	●
<i>N</i>	<i>R</i>	<i>D</i>	<i>U</i>	<i>M</i>	<i>A</i>	<i>K</i>
<i>1</i>	<i>1</i>	<i>1</i>	<i>1</i>	<i>2</i>	<i>5</i>	<i>5</i>

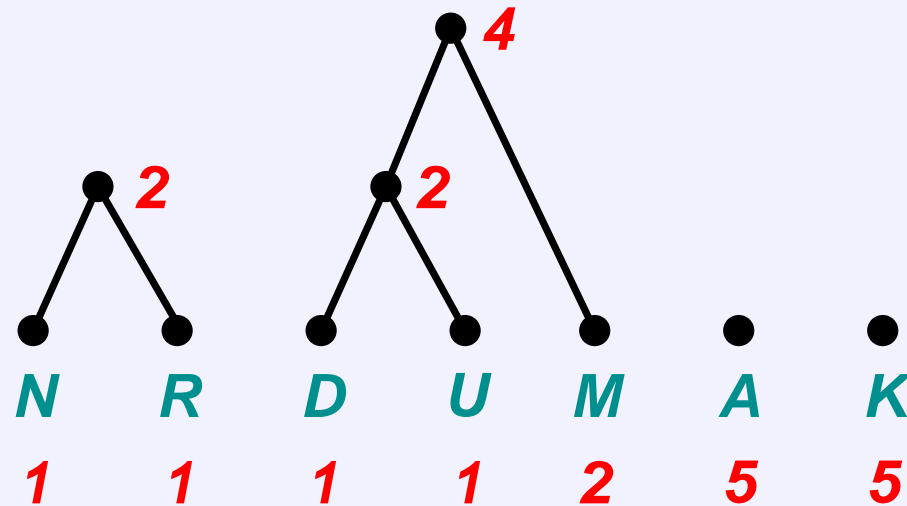
KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



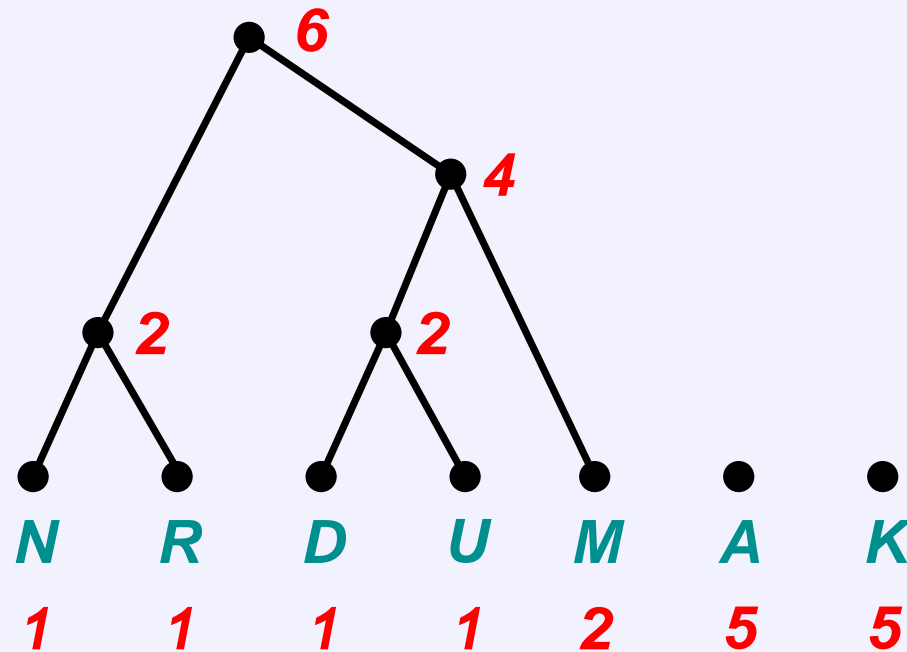
KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



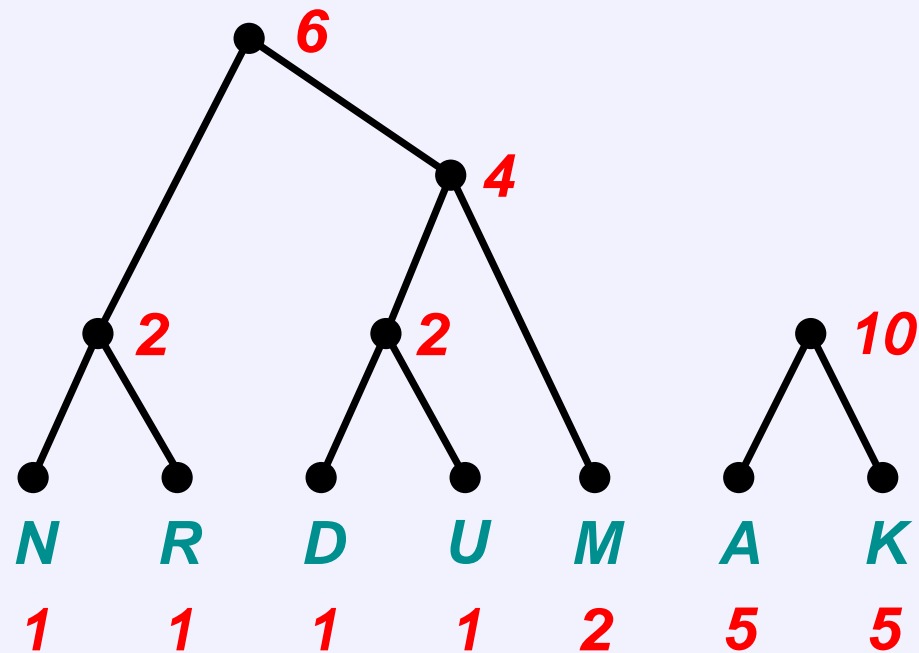
KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



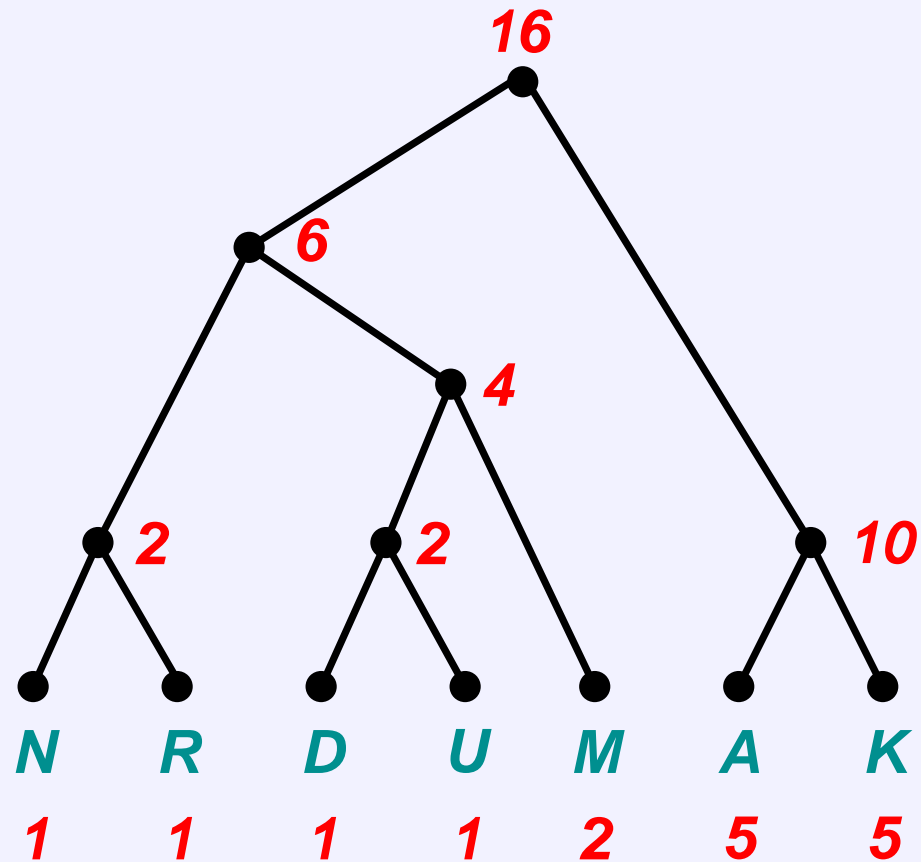
KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



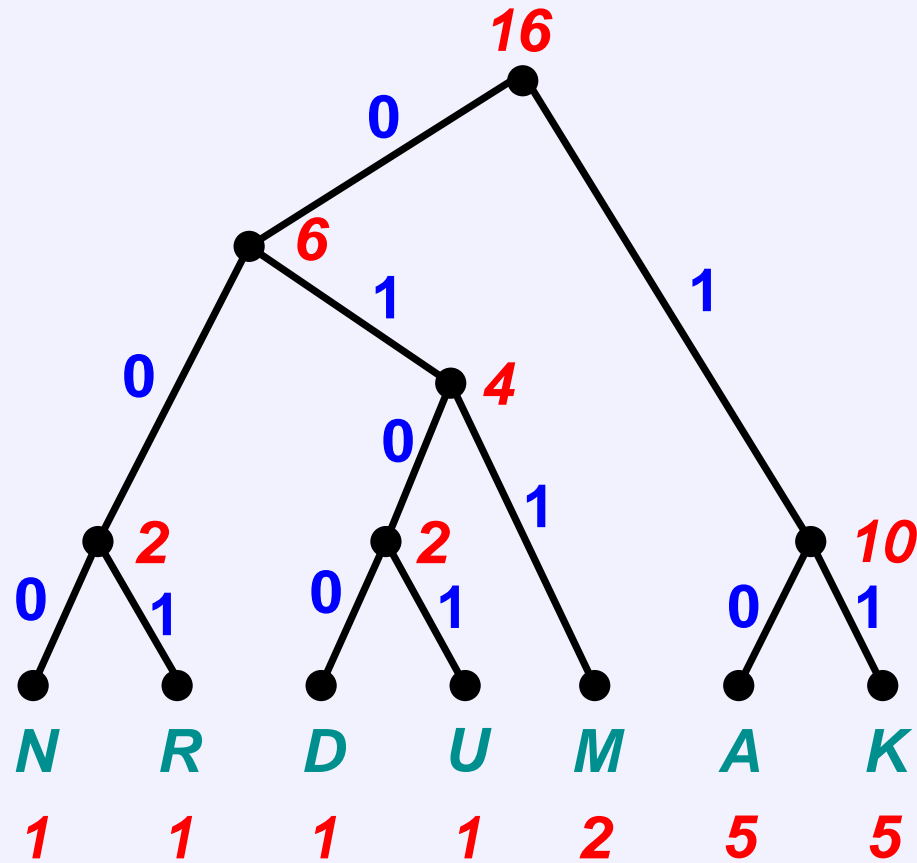
KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



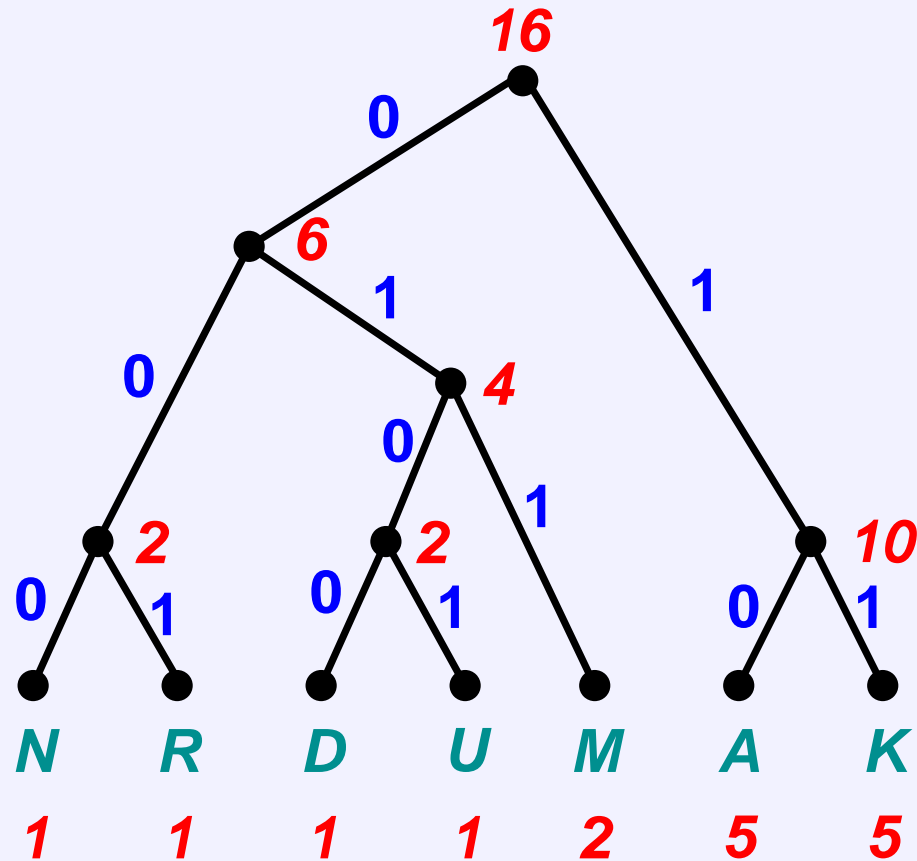
KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.

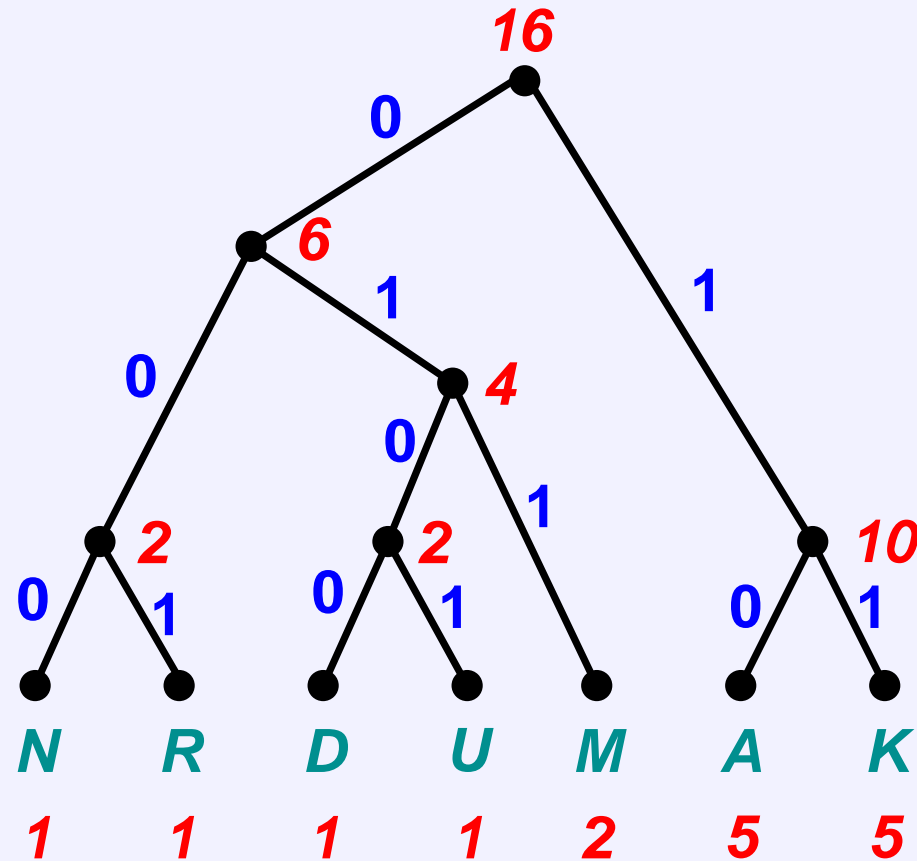


KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



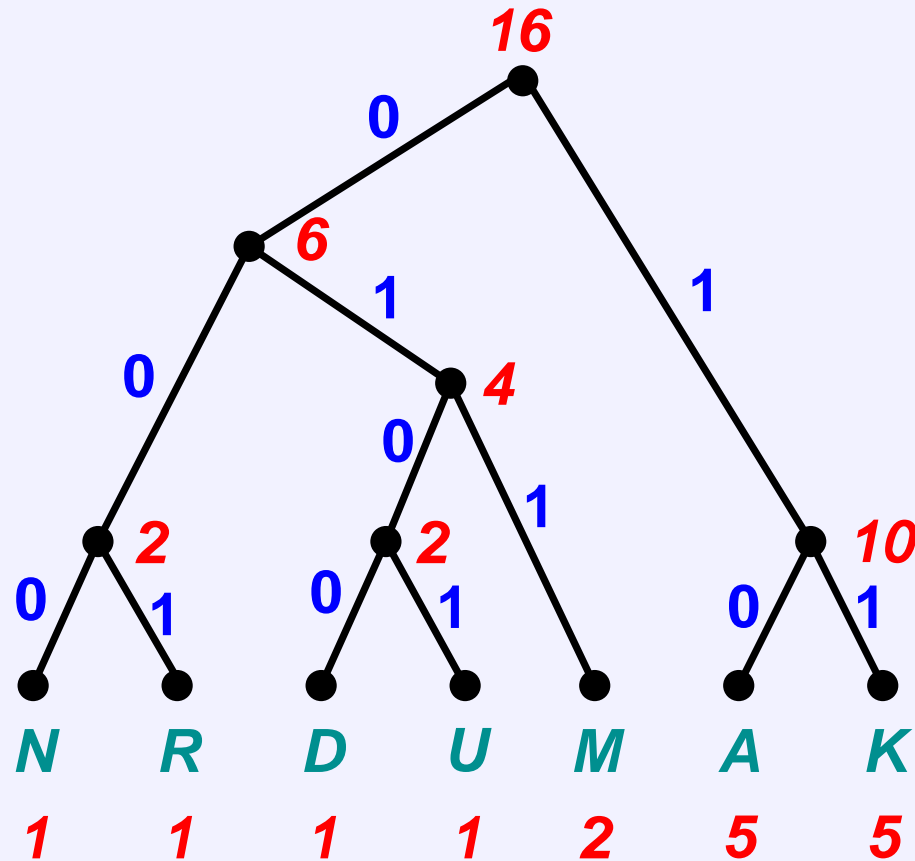
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



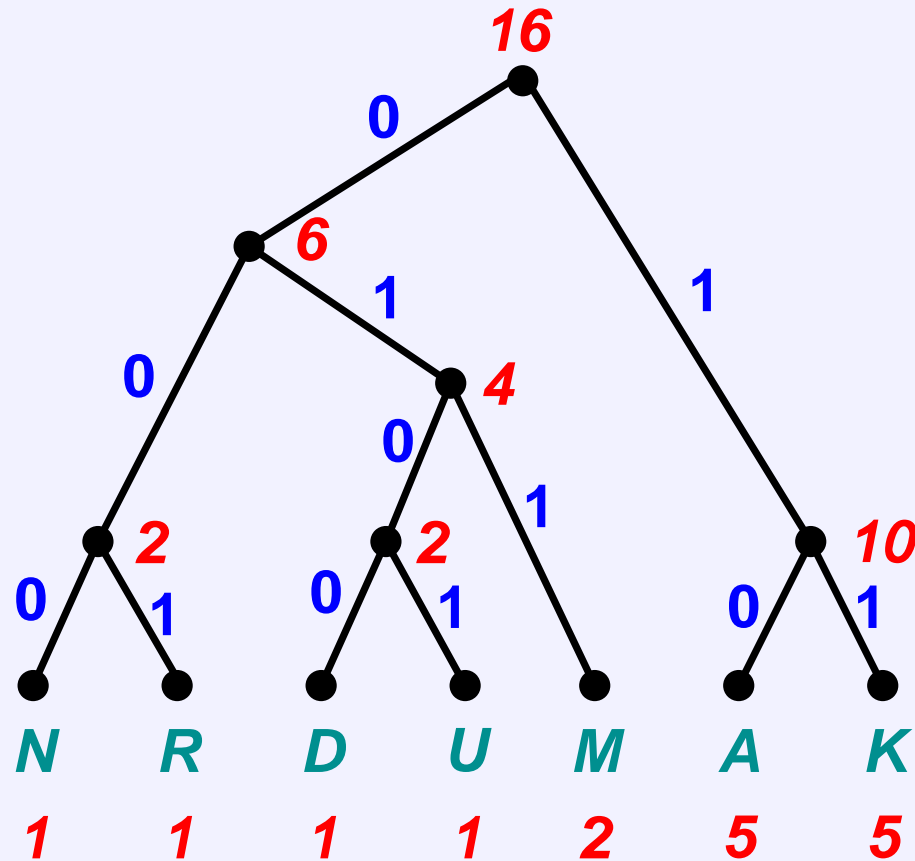
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
kódszó: 11
K

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



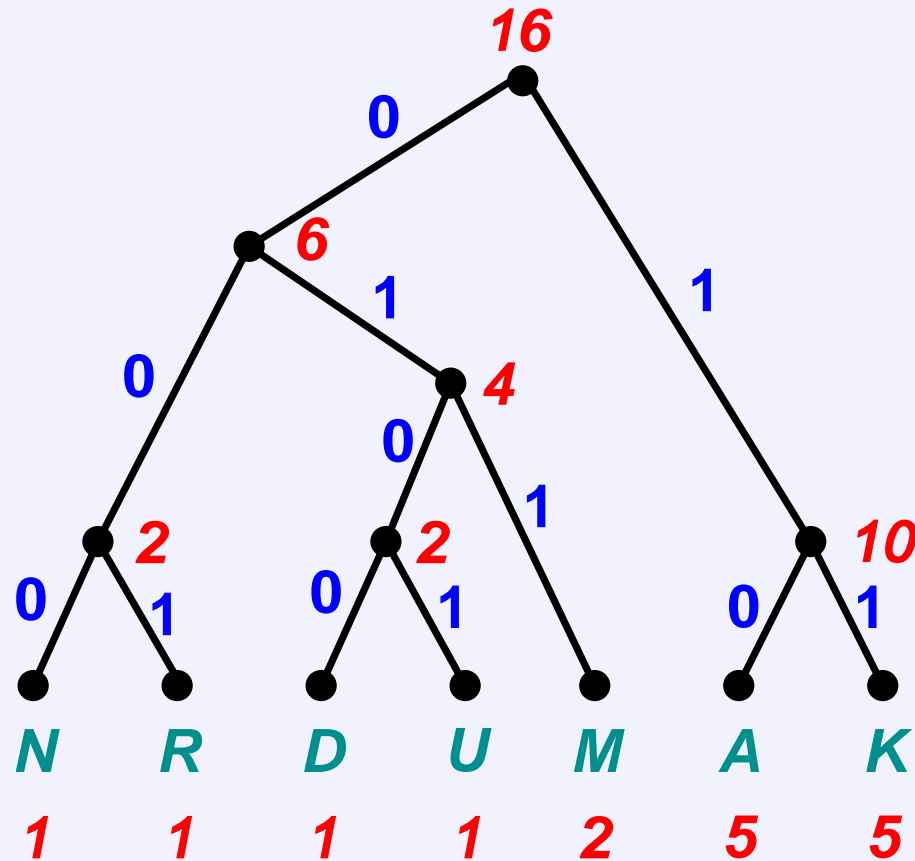
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
kódszó: 1110
KA

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



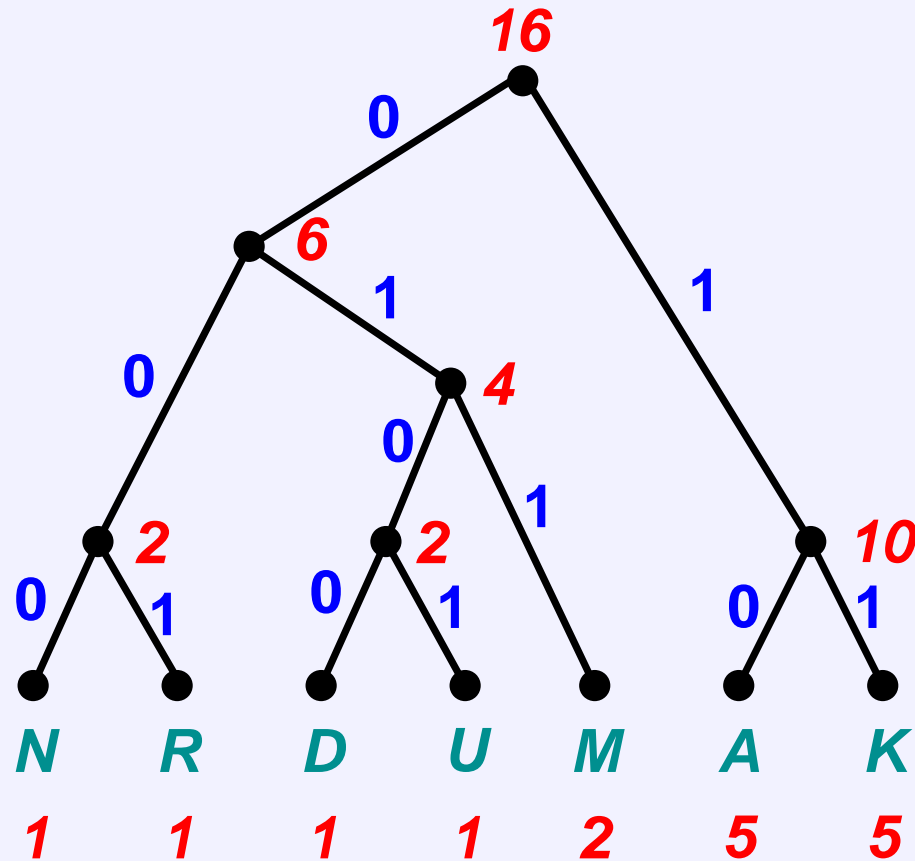
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
kódszó: 111011
KAK

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



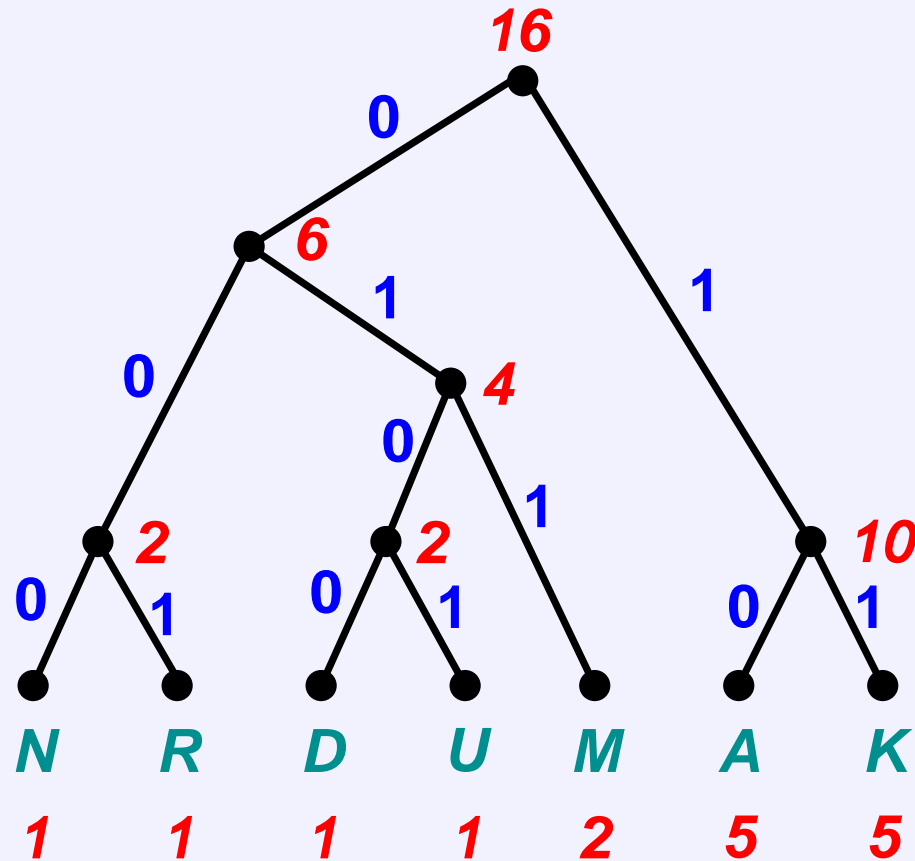
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
 kódszó: 1110110101
 KAKU

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



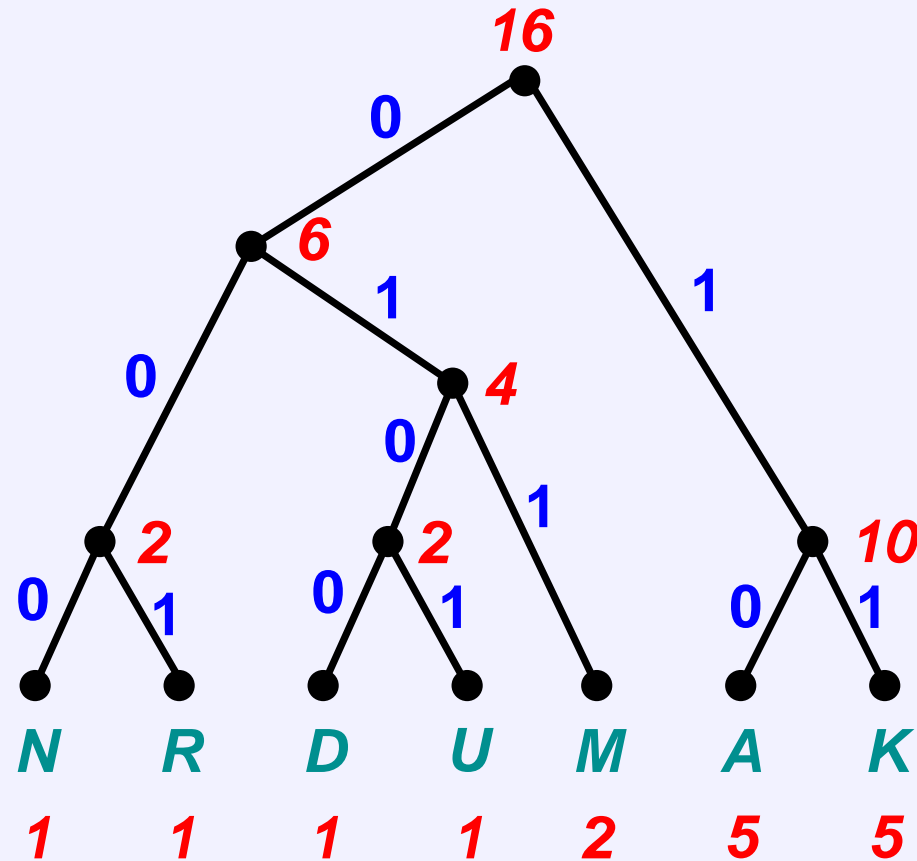
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
kódszó: 111011010111
KAKUK

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



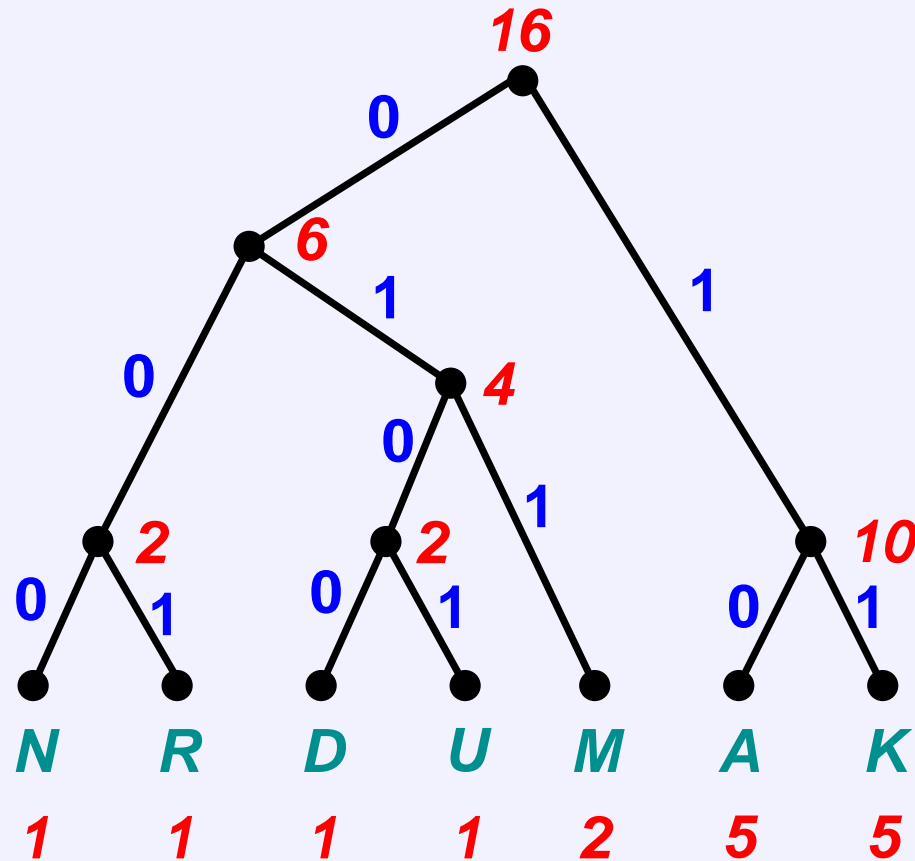
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
kódszó: 11101101011111
KAKUKK

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



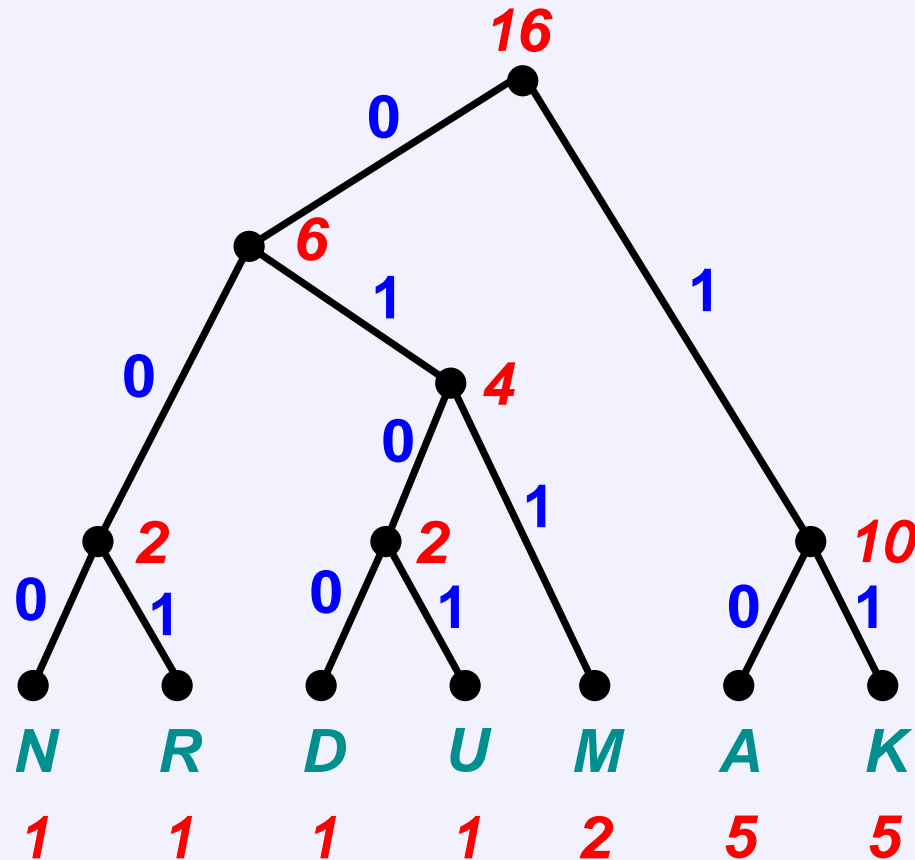
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
kódszó: 1110110101111011
KAKUKKM

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



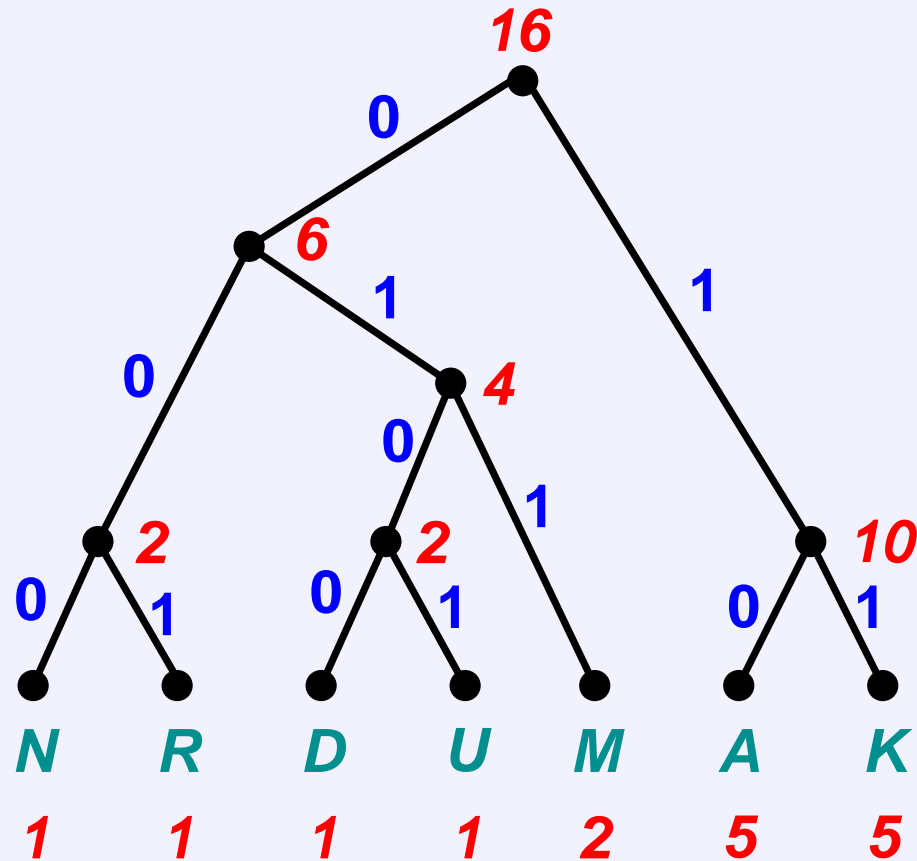
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
 kódszó: 1110110101111101110
 KAKUKKMA

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



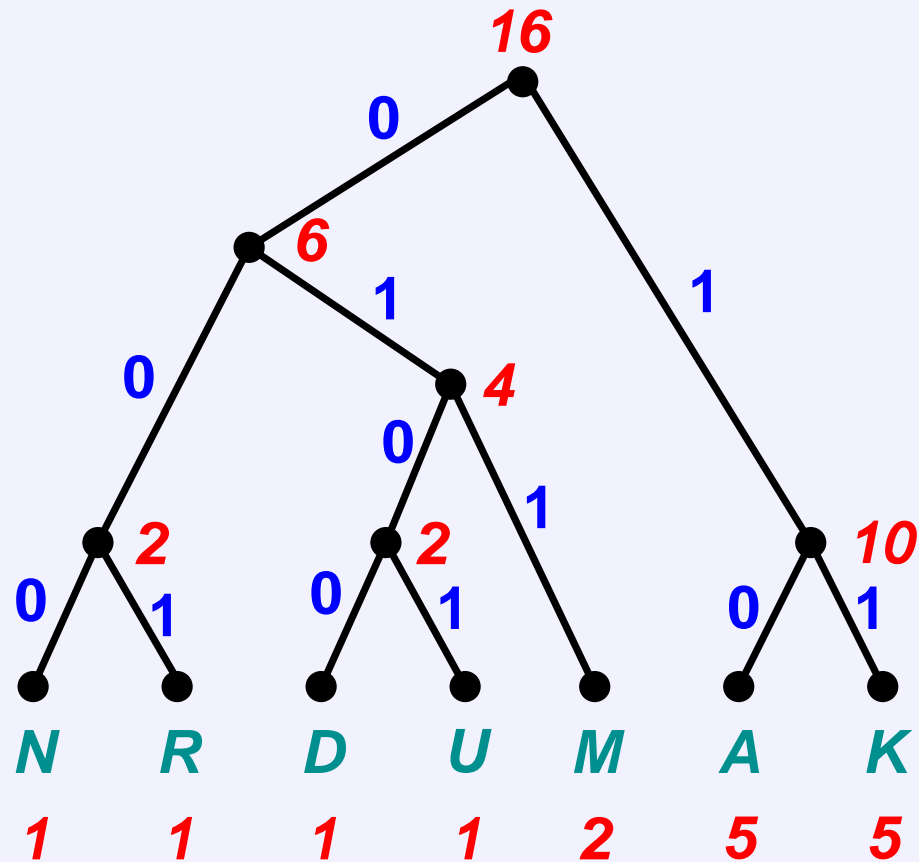
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
kódszó: 11101101011111011100100
KAKUKKMAD

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



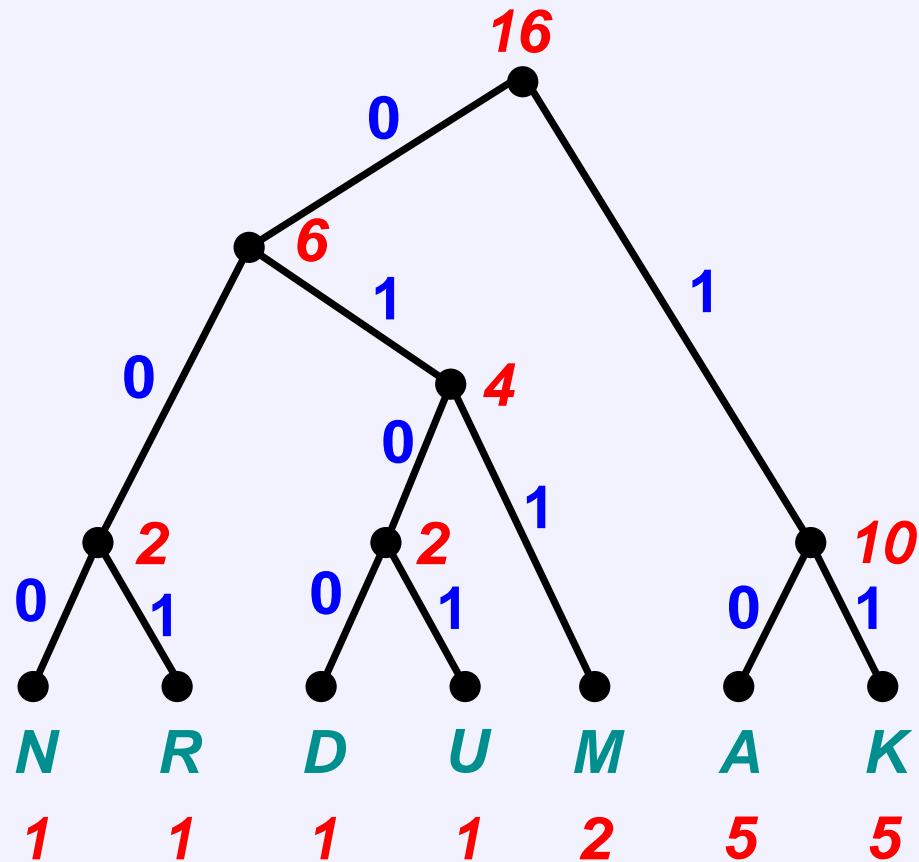
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
 kódszó: 1110110101111101110010010
 KAKUKKMADA

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



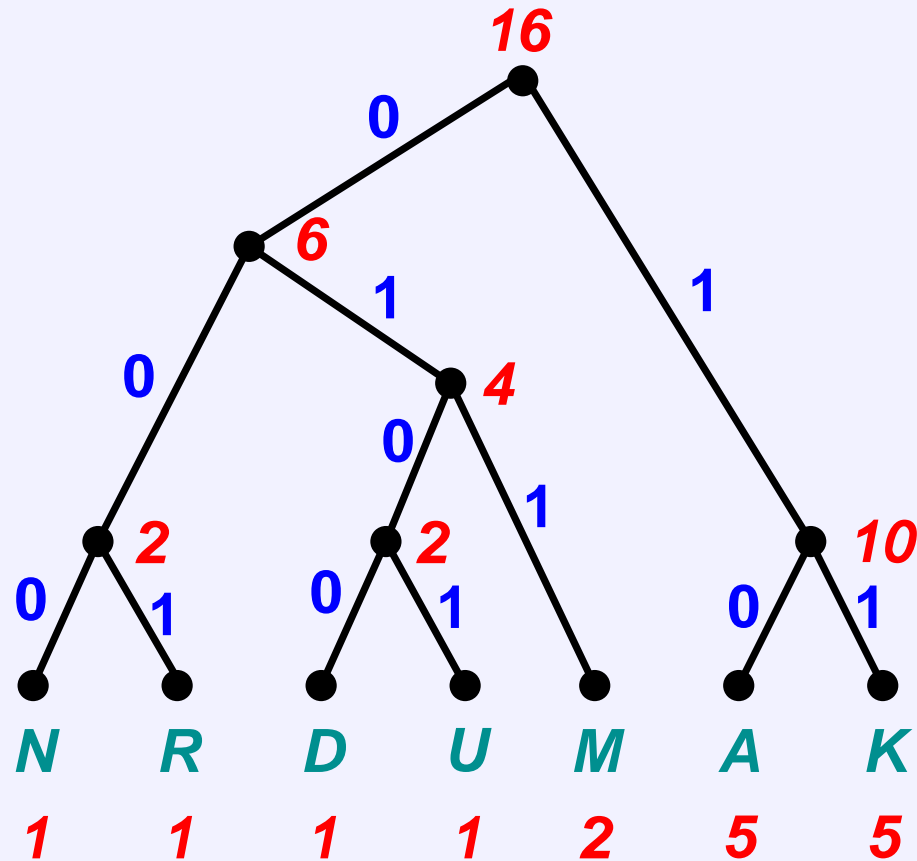
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
kódszó: 111011010111101110010010001
KAKUKKMADAR

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



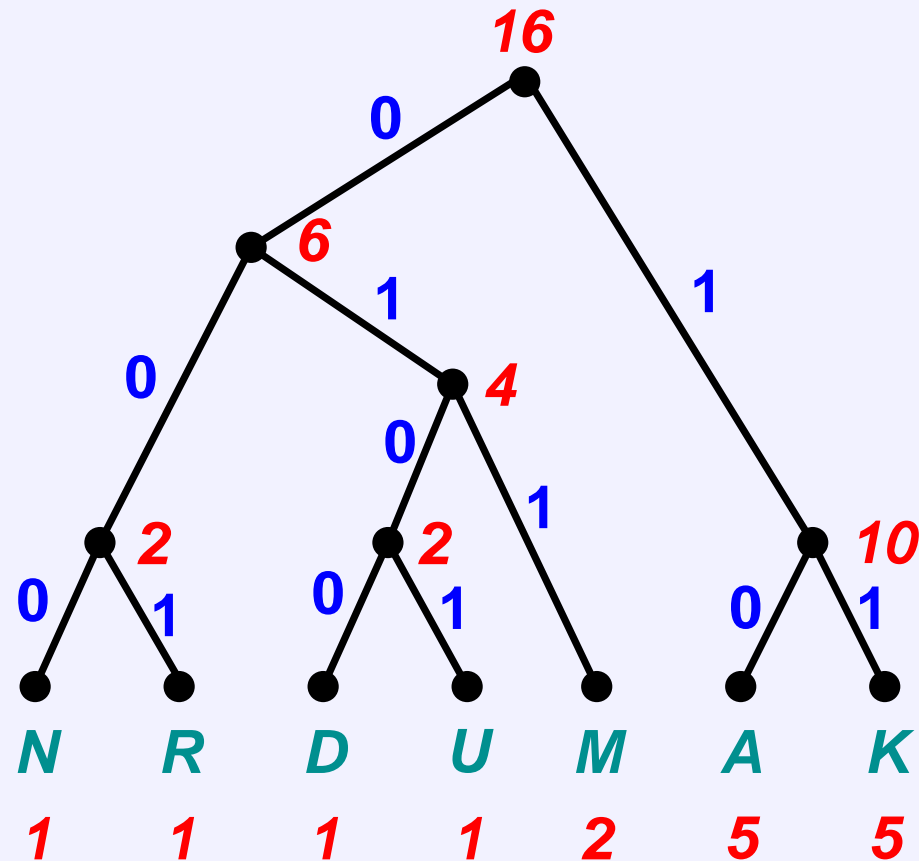
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
kódszó: 11101101011110111001001000110
KAKUKKMADARA

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



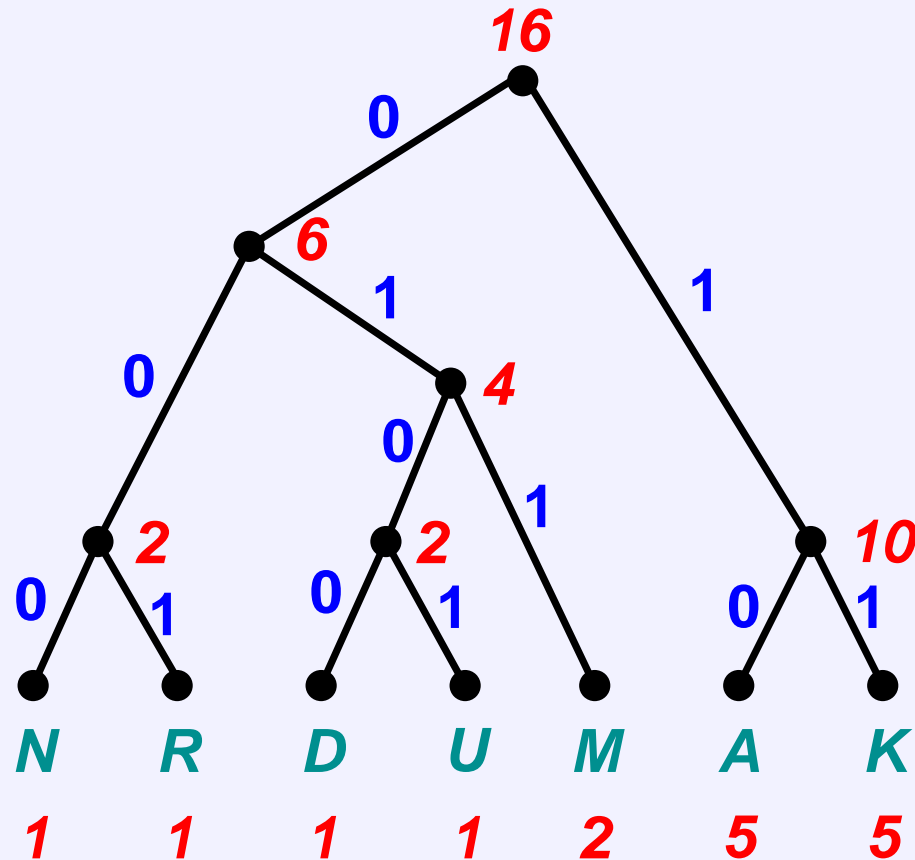
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
 kódszó: 11101101011110111001001000110011
 KAKUKKMADARAM

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



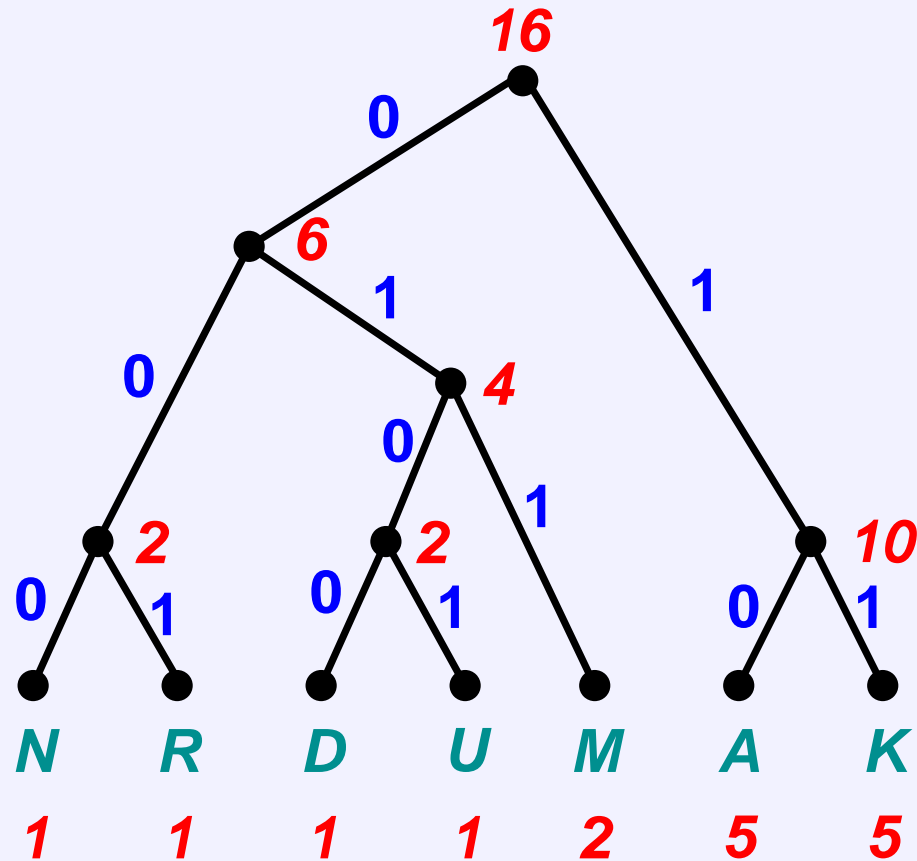
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
kódszó: 11101101011110111001001000110011000
KAKUKKMADARAMN

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



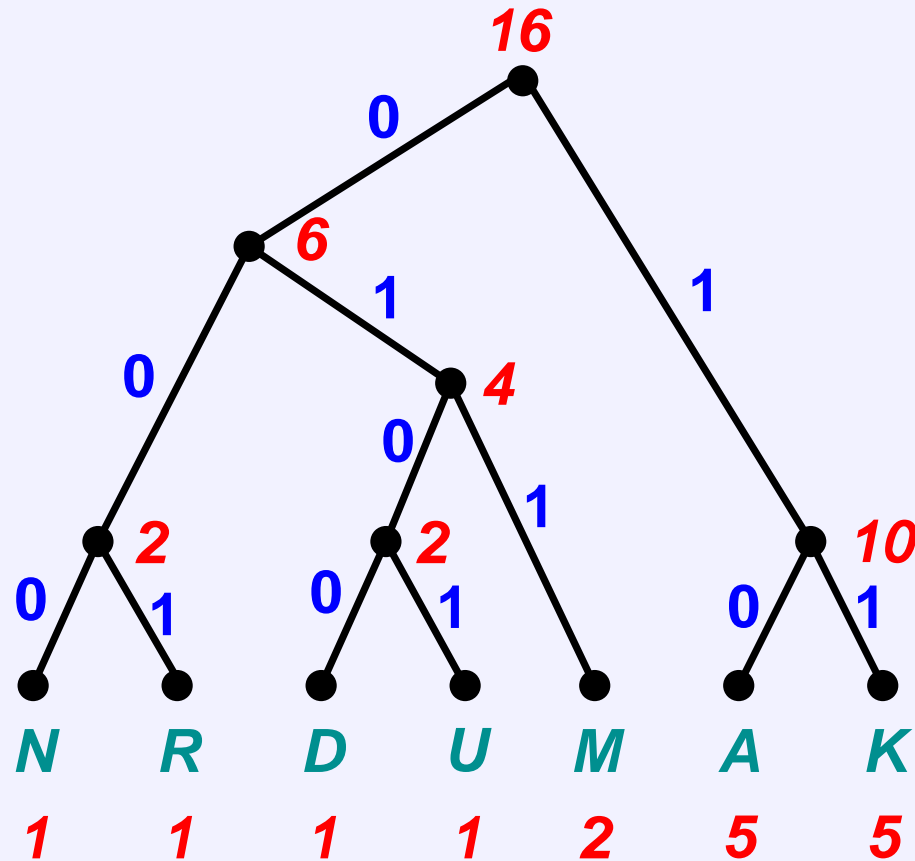
A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
kódszó: 1110110101111011100100100011001100010
KAKUKKMADARAMNA

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
 kódszó: 111011010111101110010010001100110001011
 KAKUKKMADARAMNAK

KAKUKKMADARAMNAK \implies 7 betű \implies uniform kódolással betűnként 3 bit, összesen 48 bit.



A betűk kódjai: K: 11, A: 10, M: 011, U: 0101, D: 0100, N: 000, R: 001
kódszó: 111011010111101110010010001100110001011
KAKUKKMADARAMNAK

összesen 40 bit

Optimális kód

Tétel. *A Huffman-fa optimális. Pontosabban fogalmazva, a Huffman-fa esetén az $I = \sum q_i h(b_i)$ összeg minimális azon bináris fák között, amelyek levelei b_1, \dots, b_n .*

Optimális kód

Tétel. *A Huffman-fa optimális. Pontosabban fogalmazva, a Huffman-fa esetén az $I = \sum q_i h(b_i)$ összeg minimális azon bináris fák között, amelyek levelei b_1, \dots, b_n .*

Bizonyítás: A Huffman-fa által adott I érték legyen $H(q_1, q_2, \dots, q_n)$.

Optimális kód

Tétel. *A Huffman-fa optimális. Pontosabban fogalmazva, a Huffman-fa esetén az $I = \sum q_i h(b_i)$ összeg minimális azon bináris fák között, amelyek levelei b_1, \dots, b_n .*

Bizonyítás: A Huffman-fa által adott I érték legyen $H(q_1, q_2, \dots, q_n)$.
konstrukció \implies

$$H(q_1, \dots, q_n) = H(q_1 + q_2, q_3, \dots, q_n) + q_1 + q_2$$

Optimális kód

Tétel. *A Huffman-fa optimális. Pontosabban fogalmazva, a Huffman-fa esetén az $I = \sum q_i h(b_i)$ összeg minimális azon bináris fák között, amelyek levelei b_1, \dots, b_n .*

Bizonyítás: A Huffman-fa által adott I érték legyen $H(q_1, q_2, \dots, q_n)$.
konstrukció \implies

$$H(q_1, \dots, q_n) = H(q_1 + q_2, q_3, \dots, q_n) + q_1 + q_2$$

Jelölje $Opt(q_1, q_2, \dots, q_n)$ a q_i gyakoriságok esetén elérhető optimális I -értéket.

Optimális kód

Tétel. A Huffman-fa optimális. Pontosabban fogalmazva, a Huffman-fa esetén az $I = \sum q_i h(b_i)$ összeg minimális azon bináris fák között, amelyek levelei b_1, \dots, b_n .

Bizonyítás: A Huffman-fa által adott I érték legyen $H(q_1, q_2, \dots, q_n)$.
konstrukció \implies

$$H(q_1, \dots, q_n) = H(q_1 + q_2, q_3, \dots, q_n) + q_1 + q_2$$

Jelölje $Opt(q_1, q_2, \dots, q_n)$ a q_i gyakoriságok esetén elérhető optimális I -értéket.

Lemma. Legyen továbbra is $q_1 \leq q_2$ a két legkisebb gyakoriság. Ekkor

$$Opt(q_1, q_2, \dots, q_n) = Opt(q_1 + q_2, q_3, \dots, q_n) + q_1 + q_2.$$

Bizonyítás: Minden belső csúcsnak két fia van.

Bizonyítás: Minden belső csúcsnak két fia van.

Legyen most x egy levél az optimális fánkban, melyre $h(x)$ a lehető legnagyobb, x -nek van a fában egy y testvére, címkéik q_1 és q_2 .

Bizonyítás: Minden belső csúcsnak két fia van.

Legyen most x egy levél az optimális fánkban, melyre $h(x)$ a lehető legnagyobb, x -nek van a fában egy y testvére, címkéik q_1 és q_2 .

Töröljük az x és y csúcsokat, és írjuk az új levélbe a $q_1 + q_2$ címkét.

Bizonyítás: Minden belső csúcsnak két fia van.

Legyen most x egy levél az optimális fánkban, melyre $h(x)$ a lehető legnagyobb, x -nek van a fában egy y testvére, címkéik q_1 és q_2 .

Töröljük az x és y csúcsokat, és írjuk az új levélbe a $q_1 + q_2$ címkét.

A fa I -értéke legyen J .

Bizonyítás: Minden belső csúcsnak két fia van.

Legyen most x egy levél az optimális fánkban, melyre $h(x)$ a lehető legnagyobb, x -nek van a fában egy y testvére, címkéik q_1 és q_2 .

Töröljük az x és y csúcsokat, és írjuk az új levélbe a $q_1 + q_2$ címkét.

A fa I -értéke legyen J . \implies

$$\text{Opt}(q_1, q_2, \dots, q_n) = J - (h(x) - 1)(q_1 + q_2) + q_1 h(x) + q_2 h(x) = J + q_1 + q_2,$$

Bizonyítás: Minden belső csúcsnak két fia van.

Legyen most x egy levél az optimális fánkban, melyre $h(x)$ a lehető legnagyobb, x -nek van a fában egy y testvére, címkéik q_1 és q_2 .

Töröljük az x és y csúcsokat, és írjuk az új levélbe a $q_1 + q_2$ címkét.

A fa I -értéke legyen J . \implies

$$\text{Opt}(q_1, q_2, \dots, q_n) = J - (h(x) - 1)(q_1 + q_2) + q_1 h(x) + q_2 h(x) = J + q_1 + q_2,$$

\implies az új fának is optimálisnak kell lennie a $q_1 + q_2, q_3, \dots, q_n$ gyakoriságokra vonatkozóan, hiszen, ha J -n tudnánk javítani, akkor az eredeti fán is. ✓

Bizonyítás: Minden belső csúcsnak két fia van.

Legyen most x egy levél az optimális fánkban, melyre $h(x)$ a lehető legnagyobb, x -nek van a fában egy y testvére, címkéik q_1 és q_2 .

Töröljük az x és y csúcsokat, és írjuk az új levélbe a $q_1 + q_2$ címkét.

A fa I -értéke legyen J . \implies

$$\text{Opt}(q_1, q_2, \dots, q_n) = J - (h(x) - 1)(q_1 + q_2) + q_1 h(x) + q_2 h(x) = J + q_1 + q_2,$$

\implies az új fának is optimálisnak kell lennie a $q_1 + q_2, q_3, \dots, q_n$ gyakoriságokra vonatkozóan, hiszen, ha J -n tudnánk javítani, akkor az eredeti fán is. ✓

Bizonyítás: (Tétel) Indukció, $n = 2$ ✓

Bizonyítás: Minden belső csúcsnak két fia van.

Legyen most x egy levél az optimális fánkban, melyre $h(x)$ a lehető legnagyobb, x -nek van a fában egy y testvére, címkéik q_1 és q_2 .

Töröljük az x és y csúcsokat, és írjuk az új levélbe a $q_1 + q_2$ címkét.

A fa I -értéke legyen J . \implies

$$\text{Opt}(q_1, q_2, \dots, q_n) = J - (h(x) - 1)(q_1 + q_2) + q_1 h(x) + q_2 h(x) = J + q_1 + q_2,$$

\implies az új fának is optimálisnak kell lennie a $q_1 + q_2, q_3, \dots, q_n$ gyakoriságokra vonatkozóan, hiszen, ha J -n tudnánk javítani, akkor az eredeti fán is. ✓

Bizonyítás: (Tétel) Indukció, $n = 2$ ✓

Tegyük fel, hogy legfeljebb $n - 1$ betű esetén igaz \implies

Bizonyítás: Minden belső csúcsnak két fia van.

Legyen most x egy levél az optimális fánkban, melyre $h(x)$ a lehető legnagyobb, x -nek van a fában egy y testvére, címkéik q_1 és q_2 .

Töröljük az x és y csúcsokat, és írjuk az új levélbe a $q_1 + q_2$ címkét.

A fa I -értéke legyen J . \implies

$$\text{Opt}(q_1, q_2, \dots, q_n) = J - (h(x) - 1)(q_1 + q_2) + q_1 h(x) + q_2 h(x) = J + q_1 + q_2,$$

\implies az új fának is optimálisnak kell lennie a $q_1 + q_2, q_3, \dots, q_n$ gyakoriságokra vonatkozóan, hiszen, ha J -n tudnánk javítani, akkor az eredeti fán is. ✓

Bizonyítás: (Tétel) Indukció, $n = 2$ ✓

Tegyük fel, hogy legfeljebb $n - 1$ betű esetén igaz \implies

Az indukciós feltevés szerint

$$\text{Opt}(q_1 + q_2, q_3, \dots, q_n) = H(q_1 + q_2, q_3, \dots, q_n)$$

Bizonyítás: Minden belső csúcsnak két fia van.

Legyen most x egy levél az optimális fánkban, melyre $h(x)$ a lehető legnagyobb, x -nek van a fában egy y testvére, címkéik q_1 és q_2 .

Töröljük az x és y csúcsokat, és írjuk az új levélbe a $q_1 + q_2$ címkét.

A fa I -értéke legyen J . \implies

$$\text{Opt}(q_1, q_2, \dots, q_n) = J - (h(x) - 1)(q_1 + q_2) + q_1 h(x) + q_2 h(x) = J + q_1 + q_2,$$

\implies az új fának is optimálisnak kell lennie a $q_1 + q_2, q_3, \dots, q_n$ gyakoriságokra vonatkozóan, hiszen, ha J -n tudnánk javítani, akkor az eredeti fán is. ✓

Bizonyítás: (Tétel) Indukció, $n = 2$ ✓

Tegyük fel, hogy legfeljebb $n - 1$ betű esetén igaz \implies

Az indukciós feltevés szerint

$$\text{Opt}(q_1 + q_2, q_3, \dots, q_n) = H(q_1 + q_2, q_3, \dots, q_n)$$

\implies

$$\text{Opt}(q_1, q_2, q_3, \dots, q_n) = H(q_1, q_2, q_3, \dots, q_n)$$

Bizonyítás: Minden belső csúcsnak két fia van.

Legyen most x egy levél az optimális fáinkban, melyre $h(x)$ a lehető legnagyobb, x -nek van a fában egy y testvére, címkéik q_1 és q_2 .

Töröljük az x és y csúcsokat, és írjuk az új levélbe a $q_1 + q_2$ címkét.

A fa I -értéke legyen J . \implies

$$\text{Opt}(q_1, q_2, \dots, q_n) = J - (h(x) - 1)(q_1 + q_2) + q_1 h(x) + q_2 h(x) = J + q_1 + q_2,$$

\implies az új fának is optimálisnak kell lennie a $q_1 + q_2, q_3, \dots, q_n$ gyakoriságokra vonatkozóan, hiszen, ha J -n tudnánk javítani, akkor az eredeti fán is. ✓

Bizonyítás: (Tétel) Indukció, $n = 2$ ✓

Tegyük fel, hogy legfeljebb $n - 1$ betű esetén igaz \implies

Az indukciós feltevés szerint

$$\text{Opt}(q_1 + q_2, q_3, \dots, q_n) = H(q_1 + q_2, q_3, \dots, q_n)$$

\implies

$$\text{Opt}(q_1, q_2, q_3, \dots, q_n) = H(q_1, q_2, q_3, \dots, q_n)$$

Java animáció: Huffman-fa