33. A BOYER-MOORE ALGORITMUS EGY VÁLTOZATA

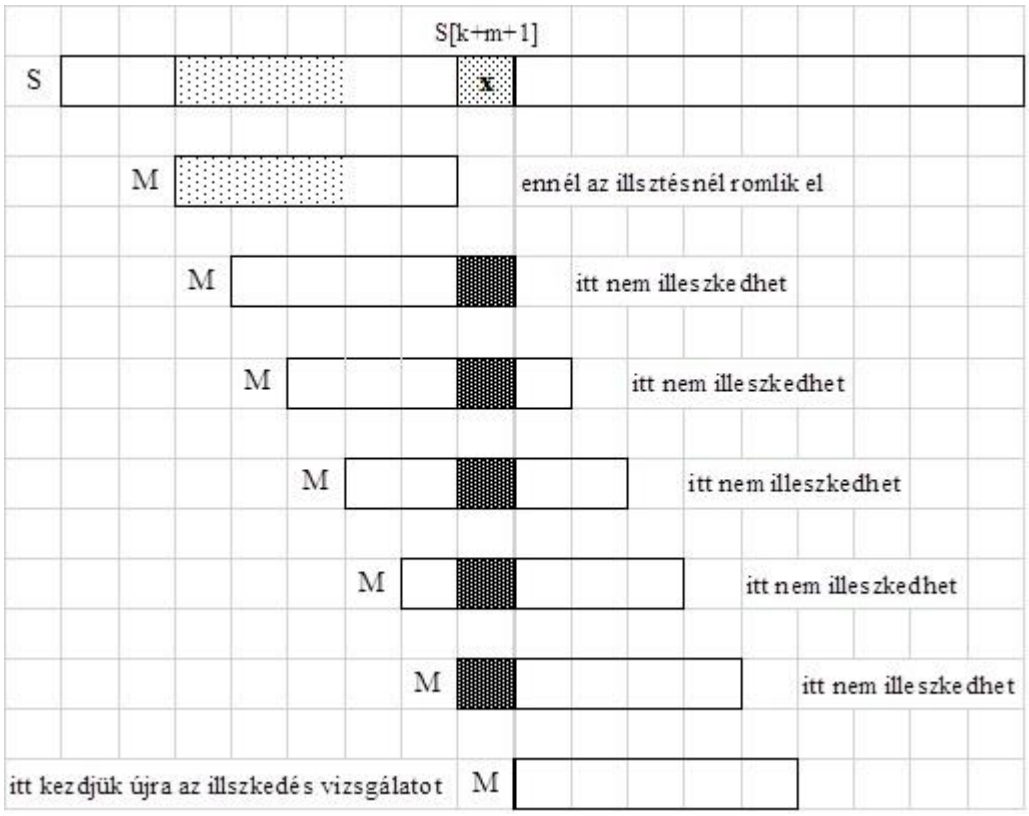
Az egyszerű mintaillesztés műveletigénye legjobb esetben *n*-es volt; most ezen próbálunk javítani. Az algoritmus, amelyet ismertetünk A *Boyer-Moore* mintaillesztés több javasolt változata körül az, amelyiknek a „*Quick Search*” (*QS*) nevet adta a szerzője (*Horspool*).

Az előző fejezetben ismertetett *KMP* eljárás a mintán belüli karaktersorozat előzetes elemzése révén jutott hatékony mintaillesztéshez. A *QS* algoritmus más stratégiát követ. Ha az illeszkedés elromlik, akkor a szövegnek *a minta utáni első karakterét* (*S*[*k*+*m*+1]) vizsgálja. Ennek a karakternek a függvényében dönti el, hogy hány pozícióval lépteti jobbra a mintát.

33.1. A *QS* eljárás működési elve

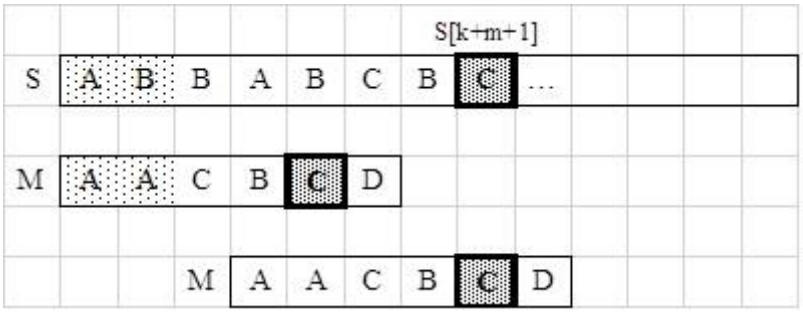
Az eljárás működése során az alábbi két esetet különböztetjük meg:

1. A szöveg minta utáni *első* karaktere *nem fordul elő* a mintában, azaz $S[k+m+1] \notin M$. Ekkor bármely olyan illesztés sikertelen lenne, ahol az $S[k+m+1]$ pozíció fedésben lenne a mintával. Tehát ezt a pozíciót "átugorhatjuk" a mintával, és a szöveg következő, $S[k+m+2]$ indexű karakterétől kezdhetjük újra, a minta elejétől kezdve, az illeszkedés vizsgálatát. Ezt az esetet illusztrálja a 33.1. ábra.



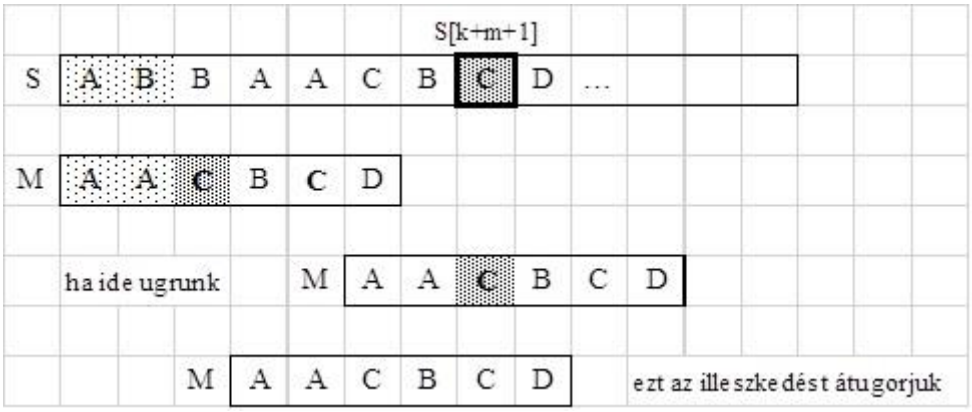
33.1. ábra. Példa: a szöveg minta utáni első karaktere nem fordul elő a mintában

2. A szöveg minta utáni *első* karaktere *előfordul* a mintában, azaz $S[k+m+1] \in M$. Ekkor vegyük a mintabeli előfordulások közül *jobbról* az *elsőt* (balról, a minta elejétől számítva az utolsót) és annyi pozícióval léptessük jobbra a mintát, hogy ez a mintabeli karakter fedésbe kerüljön a szöveg $S[k+m+1]$ karakterével, majd a minta elejétől kezdve vizsgáljuk meg újra az illeszkedést. Ez az eset látható a 33.2. ábrán.



33.2. ábra. Példa - szöveg minta utáni első karaktere előfordul a mintában

Az $S[k+m+1]$ karakternek azért kell jobbról az első mintabeli előfordulását tekinteni, mert ha létezik egy ettől balra lévő előfordulás is, és amennyiben azt választanánk, akkor esetleg átlépnénk egy jó illeszkedést, mint ahogyan a 33.3. ábrán látható esetben ez megtörténik. Akárcsak a *KMP* algoritmusnál, itt is követni kell az óvatos legkisebb biztonságos eltolás elvét.



33.3. ábra. Példa egy jó illeszkedés átugrására

31.2. A *QS* algoritmus

A QS algoritmusnak is a lényegi részét adja a minta alkalmas eltolásának ismételt végrehajtása. Az eltolás mértékének meghatározása bevezetjük a *shift* függvényt, amely az ábécé minden betűjére megadja az eltolás nagyságát, ha az illeszkedés elromlása esetén az illető betű lenne a szöveg minta utáni első karaktere.

Definíció: $shift : H \to [1..m+1]$

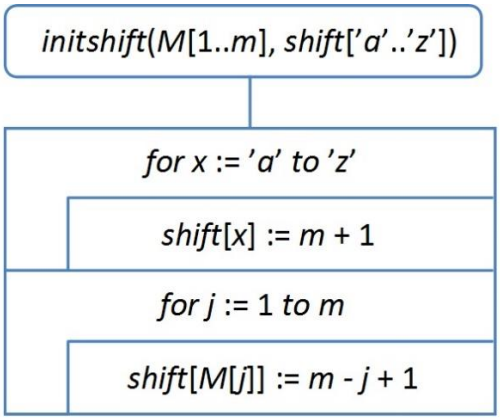
$$shift(x) = \begin{cases} m+1 & \text{, ha } x \notin M \\ m-j+1 & \text{, ha } M[j] = x \wedge \forall i \in [j+1..m] : M[i] \neq x \end{cases}$$

Megjegyzések:

- A definícióban is jól láthatóan elválik a fent említett két eset.
- Mivel a *shift* függvény csak a mintától függ, értékeit előre kiszámíthatjuk, és eltárolhatjuk egy vektorba, amit az ábécé betűivel indexelünk.

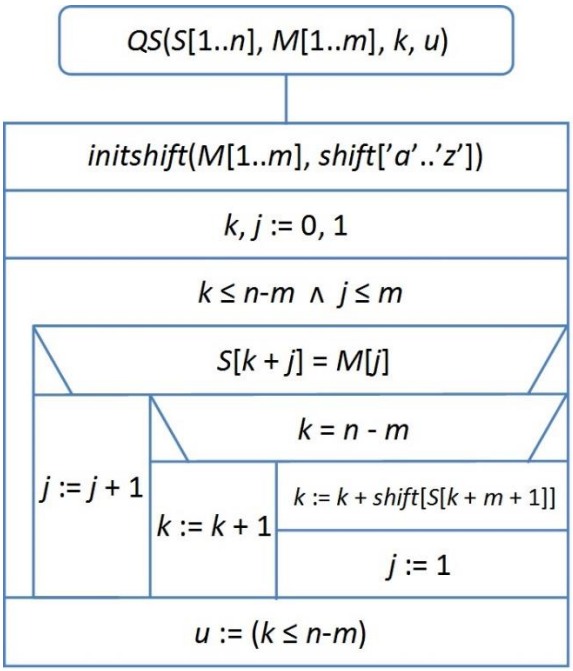
Most nézzük a *shift* függvény értékeinek a kiszámítására szolgáló *initshift* eljárást, amelynek algoritmusát a 33.4. ábra szemlélteti..

Az első ciklus feltölti a *shift* tömböt az *m*+1 értékkel. A második ciklus adja meg a minta össze betűjére – a jobbról az első mintabeli elfordulásának megfelelő – ugrás nagyságát. Azért balról-jobbra megyünk végig a mintán, hogy a többször előforduló betűk esetén az utolsó (jobbról az első) előfordulásnak megfelelő léptetést jegyezzük fel.



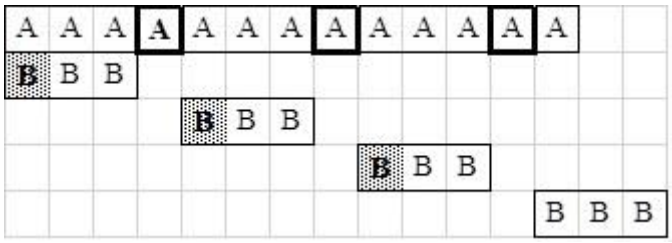
33.4. ábra. Az *inshift* eljárás algoritmusa

A *QS* algoritmus a *shift* függvény értékeinek kiszámításával kezdődik, majd *k* eltolásokkal próbáljuk illeszteni a mintát. Amennyiben az illeszkedés elromlik, akkor a *shift* függvénynek megfelelően változtatjuk az eltolás mértékét. (Ügyelnünk kell arra is, hogy amikor a mintát a szöveg végére illesztjük, és az illeszkedés elromlik, akkor ne olvassunk túl a szövegen.)



33.5. ábra. A *QS* mintaillesztés algoritmusa

A *műveletigény* meghatározását a *legjobb esettel* kezdjük. Ha a minta olyan karakterekből áll, amelyek nem fordulnak elő a szövegben, akkor már a minta első karakterénél elromlik az illeszkedés. Továbbá, ha a szövegben a minta utáni karakter nem fordul elő a mintában, akkor azt átugorhatjuk. Ezt az esetet illusztrálja a 33.6. ábra.

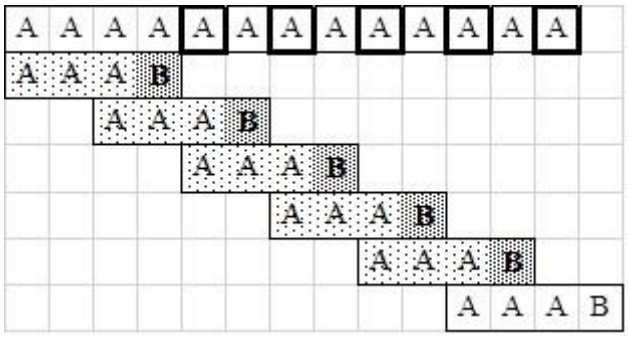


33.6. ábra. A *QS* algoritmus számára legkedvezőbb eset

A legkedvezőbb esetben a műveletigény:

$$m\ddot{O}(n,m) = \Theta\left(\frac{n}{m+1}\right)$$

A legrosszabb esetben egyrészt a minta végén romlik el az illeszkedés, másrészt mindig csak kicsiket tudunk ugrani. Ilyen esetet mutat a 33.7. ábra.



33.7. ábra. A *QS* algoritmus számra legkedvezőtlenebb eset

A legkedvezőtlenebb esetben a műveletigény:

$$M\ddot{O}(n) = \Theta(n * m)$$

A *QS* algoritmust *szekvenciális* input fájlra, illetve más, közvetlenül nem indexelhető sorozatra, csak *puffer* használatával lehet alkalmazni, mivel – ahogyan az a legrosszabb esetben is látható – szükség lehet a szövegben való visszalépésre.