Reiknirit Knuths, Morrisar og Pratts (1970)

Bergur Snorrason

April 8, 2024

- Gefum okkur langan streng s og styttri streng p.

- Gefum okkur langan streng s og styttri streng p.
- Hvernig getum við fundið alla hlutstrengi s sem eru jafnir p.

- Gefum okkur langan streng s og styttri streng p.
- Hvernig getum við fundið alla hlutstrengi s sem eru jafnir p.
- Fyrsta sem manni dettur í hug er að bera p saman við alla hlutstrengi s af sömu lengd og p.

```c
void frumstaed_strengjaleit(char* s, int n, char* p, int m, int *r)
{
    int i, j;
    for (i = 0; i < n; i++) r[i] = 0;
    for (i = 0; i < n - m + 1; i++)
    {
        for (j = 0; j < m; j++) if (s[i + j] != p[j]) break;
        if (j >= m) r[i] = 1;
    }
}
```

- Gerum ráð fyrir að strengurinn s sé af lengd n og strengurinn p sé af lengd m.

- Gerum ráð fyrir að strengurinn s sé af lengd n og strengurinn p sé af lengd m.
- Fjöldi hlutstrengja í s að lengd m er $n - m + 1$.

- Gerum ráð fyrir að strengurinn s sé af lengd n og strengurinn p sé af lengd m.
- Fjöldi hlutstrengja í s að lengd m er $n - m + 1$.
- Strengjasamanburðurinn tekur línulegan tíma.

- Gerum ráð fyrir að strengurinn s sé af lengd n og strengurinn p sé af lengd m.
- Fjöldi hlutstrengja í s að lengd m er $n - m + 1$.
- Strengjasamanburðurinn tekur línulegan tíma.
- Svo tímaflækja leitarinnar er $\mathcal{O}(\qquad\qquad)$.

- Gerum ráð fyrir að strengurinn s sé af lengd n og strengurinn p sé af lengd m.
- Fjöldi hlutstrengja í s að lengd m er $n - m + 1$.
- Strengjasamanburðurinn tekur línulegan tíma.
- Svo tímaflækja leitarinnar er $\mathcal{O}(nm - m^2)$.

- Gerum ráð fyrir að strengurinn s sé af lengd n og strengurinn p sé af lengd m.
- Fjöldi hlutstrengja í s að lengd m er $n - m + 1$.
- Strengjasamanburðurinn tekur línulegan tíma.
- Svo tímaflækja leitarinnar er $\mathcal{O}(nm - m^2)$.
- Ef $m = n/2$ þá er $nm - m^2 = n^2/2 - n^2/4 = n^2/4$ tímaflækjan er í raun $\mathcal{O}(\quad)$.

- Gerum ráð fyrir að strengurinn s sé af lengd n og strengurinn p sé af lengd m.
- Fjöldi hlutstrengja í s að lengd m er $n - m + 1$.
- Strengjasamanburðurinn tekur línulegan tíma.
- Svo tímaflækja leitarinnar er $\mathcal{O}(nm - m^2)$.
- Ef $m = n/2$ þá er $nm - m^2 = n^2/2 - n^2/4 = n^2/4$ tímaflækjan er í raun $\mathcal{O}(n^2)$.

- Gerum ráð fyrir að strengurinn s sé af lengd n og strengurinn p sé af lengd m.
- Fjöldi hlutstrengja í s að lengd m er $n - m + 1$.
- Strengjasamanburðurinn tekur línulegan tíma.
- Svo tímaflækja leitarinnar er $\mathcal{O}(nm - m^2)$.
- Ef $m = n/2$ þá er $nm - m^2 = n^2/2 - n^2/4 = n^2/4$ tímaflækjan er í raun $\mathcal{O}(n^2)$.
- Dæmi um leiðinlega strengi væri $s =$ „*aaaaaaaaaaaaaaaa*" og $p =$ „*aaaaaaab*".

- Gerum ráð fyrir að strengurinn s sé af lengd n og strengurinn p sé af lengd m.
- Fjöldi hlutstrengja í s að lengd m er $n - m + 1$.
- Strengjasamanburðurinn tekur línulegan tíma.
- Svo tímaflækja leitarinnar er $\mathcal{O}(nm - m^2)$.
- Ef $m = n/2$ þá er $nm - m^2 = n^2/2 - n^2/4 = n^2/4$ tímaflækjan er í raun $\mathcal{O}(n^2)$.
- Dæmi um leiðinlega strengi væri $s =$ „$aaaaaaaaaaaaaaaa$" og $p =$ „$aaaaaaab$".
- Þessi aðferð virkar þó sæmilega ef strengirnir eru nógu óreglulegir.

- Gerum ráð fyrir að strengurinn s sé af lengd n og strengurinn p sé af lengd m.
- Fjöldi hlutstrengja í s að lengd m er $n - m + 1$.
- Strengjasamanburðurinn tekur línulegan tíma.
- Svo tímaflækja leitarinnar er $\mathcal{O}(nm - m^2)$.
- Ef $m = n/2$ þá er $nm - m^2 = n^2/2 - n^2/4 = n^2/4$ tímaflækjan er í raun $\mathcal{O}(n^2)$.
- Dæmi um leiðinlega strengi væri $s =$ „$aaaaaaaaaaaaaaaa$" og $p =$ „$aaaaaaab$".
- Þessi aðferð virkar þó sæmilega ef strengirnir eru nógu óreglulegir.
- Dæmi um það hvenær þessi aðferð er góð er ef maður er að leita að orði í skáldsögu.

- Aðferðin er líka nógu góð ef $\mathcal{O}(n^2)$ er ekki of hægt.

- Aðferðin er líka nógu góð ef $\mathcal{O}(n^2)$ er ekki of hægt.
- Það er þó óþarfi að útfæra hana því hún fylgir með flestum forritunarmálum, til dæmis:

- Aðferðin er líka nógu góð ef $\mathcal{O}(n^2)$ er ekki of hægt.
- Það er þó óþarfi að útfæra hana því hún fylgir með flestum forritunarmálum, til dæmis:
  - Í `string.h` í C er `strstr(..)`.

- Aðferðin er líka nógu góð ef $\mathcal{O}(n^2)$ er ekki of hægt.
- Það er þó óþarfi að útfæra hana því hún fylgir með flestum forritunarmálum, til dæmis:
  - Í `string.h` í C er `strstr(..)`.
  - Í `string` í C++ er `find(..)`.

- Aðferðin er líka nógu góð ef $\mathcal{O}(n^2)$ er ekki of hægt.
- Það er þó óþarfi að útfæra hana því hún fylgir með flestum forritunarmálum, til dæmis:
  - Í `string.h` í C er `strstr(..)`.
  - Í `string` í C++ er `find(..)`.
  - Í `String` í Java er `indexOf(..)`.

- Aðferðin er líka nógu góð ef $\mathcal{O}(n^2)$ er ekki of hægt.
- Það er þó óþarfi að útfæra hana því hún fylgir með flestum forritunarmálum, til dæmis:
  - Í `string.h` í C er `strstr(..)`.
  - Í `string` í C++ er `find(..)`.
  - Í `String` í Java er `indexOf(..)`.
- Munið bara að ef $n > 10^4$ er þetta yfirleitt of hægt.

- Við getum bætt tímaflækjun með því að nota svokallað *forstrengsfall* (e. *prefix function*) strengsins a.

- Við getum bætt tímaflækjun með því að nota svokallað *forstrengsfall* (e. *prefix function*) strengsins a.
- Forstrengsfallið f er gefið með
  $f(j) = \max\{k \in \mathbb{N} \colon k < |a|, a[1, k] = a[j - k + 1, j]\}$.

- Við getum bætt tímaflækjun með því að nota svokallað *forstrengsfall* (e. *prefix function*) strengsins a.
- Forstrengsfallið f er gefið með $f(j) = \max\{k \in \mathbb{N} \colon k < |a|, a[1, k] = a[j - k + 1, j]\}$.
- Sjáum fyrst að þetta fall uppfyllir $f(j + 1) \leq f(j) + 1$.

- Við getum bætt tímaflækjun með því að nota svokallað *forstrengsfall* (e. *prefix function*) strengsins a.
- Forstrengsfallið f er gefið með $f(j) = \max\{k \in \mathbb{N} \colon k < |a|, a[1, k] = a[j - k + 1, j]\}$.
- Sjáum fyrst að þetta fall uppfyllir $f(j + 1) \leq f(j) + 1$.
- Látum $k = f(j)$ og sjáum að ef $a[j + 1] = a[k]$ þá er $f(j + 1) = k + 1$.

- Við getum bætt tímaflækjun með því að nota svokallað *forstrengsfall* (e. *prefix function*) strengsins a.
- Forstrengsfallið f er gefið með $f(j) = \max\{k \in \mathbb{N}\colon k < |a|, a[1,k] = a[j-k+1,j]\}$.
- Sjáum fyrst að þetta fall uppfyllir $f(j+1) \leq f(j) + 1$.
- Látum $k = f(j)$ og sjáum að ef $a[j+1] = a[k]$ þá er $f(j+1) = k+1$.
- Ef $a[j+1] \neq a[k]$ þá þurfum við að minnka k þangað til við fáum jöfnuð.

- Við getum bætt tímaflækjun með því að nota svokallað *forstrengsfall* (e. *prefix function*) strengsins a.
- Forstrengsfallið f er gefið með $f(j) = \max\{k \in \mathbb{N} \colon k < |a|, a[1,k] = a[j-k+1,j]\}$.
- Sjáum fyrst að þetta fall uppfyllir $f(j+1) \leq f(j) + 1$.
- Látum $k = f(j)$ og sjáum að ef $a[j+1] = a[k]$ þá er $f(j+1) = k+1$.
- Ef $a[j+1] \neq a[k]$ þá þurfum við að minnka k þangað til við fáum jöfnuð.
- Við minnkum k með því að láta $k' = f(k-1)$.

```c
void prefix_function(char *a, int *b)
{
    int i, j, m = strlen(a);
    for (i = 0, j = b[0] = -1; i < m; b[++i] = ++j)
        while (j >= 0 && a[i] != a[j]) j = b[j];
}
```

- Takið eftir að hver ítrun innri lykkjanna svarar til einnar ítrunar ytri lykkjanna.

- Takið eftir að hver ítrun innri lykkjanna svarar til einnar ítrunar ytri lykkjanna.
- Svo innri lykkjan keyrir, í heildina, ekki oftar en ytri lykkjan.

- Takið eftir að hver ítrun innri lykkjanna svarar til einnar ítrunar ytri lykkjanna.
- Svo innri lykkjan keyrir, í heildina, ekki oftar en ytri lykkjan.
- Því er tímaflækjan í heildina $\mathcal{O}(\quad)$ fyrir streng af lengd n.

- Takið eftir að hver ítrun innri lykkjanna svarar til einnar ítrunar ytri lykkjanna.
- Svo innri lykkjan keyrir, í heildina, ekki oftar en ytri lykkjan.
- Því er tímaflækjan í heildina $\mathcal{O}(\,n\,)$ fyrir streng af lengd n.

- En hvernig getum við notað forstrengsfallið til að framkvæma strengjaleit?

- En hvernig getum við notað forstrengsfallið til að framkvæma strengjaleit?
- Gerum ráð fyrir að við séum að leit að streng p í stærri streng s.

- En hvernig getum við notað forstrengsfallið til að framkvæma strengjaleit?
- Gerum ráð fyrir að við séum að leit að streng p í stærri streng s.
- Látum þá $a = p +$ "α" $+ s$, þar sem $+$ táknar samskeytingu strengja og α er stafur sem er hvorki í p né s.

- En hvernig getum við notað forstrengsfallið til að framkvæma strengjaleit?
- Gerum ráð fyrir að við séum að leit að streng p í stærri streng s.
- Látum þá $a = p +$ "α" $+ s$, þar sem $+$ táknar samskeytingu strengja og α er stafur sem er hvorki í p né s.
- Við látum svo f vera forstrengsfall strengsins a.

- En hvernig getum við notað forstrengsfallið til að framkvæma strengjaleit?
- Gerum ráð fyrir að við séum að leit að streng p í stærri streng s.
- Látum þá $a = p +$ "α" $+ s$, þar sem $+$ táknar samskeytingu strengja og α er stafur sem er hvorki í p né s.
- Við látum svo f vera forstrengsfall strengsins a.
- Við höfum þá að $f \leq |p|$ og $f(j) = |p|$ þá og því aðeins að $j > |p|$ og hlutstrengurinn í s sem byrjar á vísi $j - |p| - 1$ og er $|p|$ af lengd er sá sami og p.

```c
void prefix_function(char *a, int *b)
{
    int i, j, m = strlen(a);
    for (i = 0, j = b[0] = -1; i < m; b[++i] = ++j)
        while (j >= 0 && a[i] != a[j]) j = b[j];
}

void kmp(char *s, char *p, int *r)
{
    int i, j, n = strlen(s), m = strlen(p), b[n + m + 2];
    char a[n + m + 2];
    strcpy(a, p), strcat(a, "\n"), strcat(a, s);
    prefix_function(a, b);
    for (i = 0; i < n; i++) r[i] = i < n - m + 1 && b[i + 2*m + 1] >= m;
}
```

- Gerum ráð fyrir að $|s| = n$ og $|p| = m$.

- Gerum ráð fyrir að $|s| = n$ og $|p| = m$.
- Þá er $|a| = n + m + 1$, svo tímaflækjan er $\mathcal{O}($ $)$.

- Gerum ráð fyrir að $|s| = n$ og $|p| = m$.
- Þá er $|a| = n + m + 1$, svo tímaflækjan er $\mathcal{O}(n + m)$.