1. DÚ Z ADS2

Práce s textem

Termín **úterý 24. října 08:00**. Všimněte si času – v pondělní noci můžete pracovat jak dlouho chcete, ale nezapomeňte to odeslat. Úkoly odevzdávejte emailem na adresu `bohm(at)iuuk.mff.cuni.cz`. Do předmětu napište **ADS2 HW1**. V emailu zmiňte přezdívku, pod kterou mohu zveřejnit průběžné bodové hodnocení (iniciály, přezdívku z IRC/ze skauta, náhodný řetězec délky 6 ...).

Prosím odevzdávejte digitálně psané dokumenty, buď ve formátu plaintext nebo v PDF. Při jakýchkoli problémech nebo otázkách se nebojte napsat email a vyřešíme to.

PŘÍKLAD PRVNÍ [**4 body.**] Budeme se nyní zabývat problémem nalezení NEJDELŠÍ SPOLEČNÉ PODPOSLOUPNOSTI. Rozdíl mezi *podřetězcem* a vybranou podposloupností je ten, že podposloupnost není souvislá, takže třeba `main` je podposloupností slova `martin`, ale rozhodně ne podřetězcem.

Zpět k problému: na vstupu dostaneme řetězec α (délky a) a řetězec β (délky b) a zajímá nás ten nejdelší řetězec, který je podposloupností jak α, tak β – kdyby jich bylo více stejně dlouhých, tak libovolný takový.

1. Důležitý první krok: nejprve dokažte, že problém je vůbec řesitelný polynomiálně! Stačí jakékoli řešení, hlavně když bude z vaší hlavy.
2. Vymyslete algoritmus, který problém vyřeší v čase $O(a \cdot b)$.

Jak na to? Projděte si na cvičení řešený příklad NEJDELŠÍ SPOLEČNÝ PODŘETĚZEC, a zkuste vyřešit podposloupnost podobným přístupem. Může pomoci si klást otázku: když budu jeden řetězec přidávat znak po znaku, jak po přidání nového znaku přepočítat řešení?

Zbylé dva problémy řeší na cvičení hodně propíraný problém HLEDÁNÍ VÍCE JEHEL V SENĚ, na který je užitečný algoritmus Aho-Corasicková (viz přednáška). Pro připomenutí:

HLEDÁNÍ VÍCE JEHEL V SENĚ: Na vstupu máme seno σ a několik jehel $\iota_1, \iota_2, \ldots, \iota_k$. Celková délka sena je s, celková délka všech jehel $j_1 + j_2 + \ldots + j_k = J$. Naším úkolem je vypsat dvojici (c, i), kdykoli na pozici c v seně končí jehla ι_i.

PŘÍKLAD DRUHÝ [**3 body.**] Pokud jste četli nebo slyšeli algoritmus Aho-Corasicková, tak víte, že jeho složitost je $O(s + J + V)$, kde V je počet výskytů v textů. V nějakém smyslu nás tedy zpomaluje, že musíme všechny výskyty vypsat.

Co když si tedy místo výpisu všech výskytů dáme za úkol vypsat jen celkový počet všech výskytů pro každou jehlu j_i? Výstup už jistě bude velikosti $O(J)$ – podaří se vám tedy navrhnout algoritmus, který doběhne v čase $O(s + J)$?

PŘÍKLAD TŘETÍ [**3 body.**] Zbylo z cvičení:

Navrhuji alternativní algoritmus k řešení HLEDÁNÍ VÍCE JEHEL V SENĚ. Opět máme trii postavenou ze slov v jehelníčku, a opět máme zpětné hrany jako v KMP. Místo „zkratkových hran“ v Aho-Corasickové si ale zjednodušíme implementaci tím, že si pro každý vrchol trie v předpočítáme množinu $M(v)$ indexů slov k ohlášení.

Dokažte, že pro slovník velikosti J může být součet $M(v)$ v celé trii dokonce $\omega(J)$ – takže předpočítat si množiny v čase $O(J)$ nestihneme. Připomínám, že $\omega(J)$ značí, že jich je asymptoticky ostře více, než J. Třeba $J^{3/2}$, $J \log J$ apod.

Pozor na to, že nám jde o součet velikostí množin, ne součet všech délek slov v těchto množinách. Když má $M(v)$ tři slova délky sto, tak $|M(v)|$ je stále 3.

Nezapomeňte také na to, že nestačí jeden konkrétní vstup, ale bude jich potřeba celá posloupnost s rostoucím J, aby váš výsledek byl skutečně asymptotický!