Dynamické programování

Postup návrhu algoritmu dynamickým programováním:

1. Rozmyslet rozdělení na (překrývající se) podproblémy.
2. Sestavit rekurzivní algoritmus.
   - Typicky hledáme optimální hodnotu, pro niž sestavíme rekurzivní vztah.
   - V tomto kroku se nestaráme o složitost!
3. Určit správné pořadí počítání podproblémů, které zajistí, že se každý spočítá právě jednou. (*bottom-up* přístup)
   - Alternativou je *top-down with memoization*, ale většinou preferujeme *bottom-up*. Proč?
4. Chceme-li kromě optimální hodnoty znát i její realizaci: Navrhnout způsob, jak řešení sestavit.
   - Použijeme napočítané hodnoty podproblémů, někdy je třeba mírná modifikace.

Dynamické programování

Nejmenší počet kroků k jedničce

Máme dáno kladné celé číslo n. V každé iteraci můžeme provést jednu z následujících operací:

1. odečíst od n jedničku;
2. pokud je n sudé, vydělit n dvěma;
3. pokud je n dělitelné třemi, vydělit n třemi.

Jaký je nejmenší počet iterací takový, že se z n stane číslo 1?

- Použijeme přístup: „V každé iteraci použijeme tu operaci, která hodnotu čísla n nejvíc sníží.“ Je toto korektní přístup?

Dynamické programování

Nejmenší počet kroků k jedničce

Máme dáno kladné celé číslo n. V každé iteraci můžeme provést jednu z následujících operací:

1. odečíst od n jedničku;
2. pokud je n sudé, vydělit n dvěma;
3. pokud je n dělitelné třemi, vydělit n třemi.

Jaký je nejmenší počet iterací takový, že se z n stane číslo 1?

- Použijeme přístup: „V každé iteraci použijeme tu operaci, která hodnotu čísla n nejvíc sníží.“ Je toto korektní přístup?

- Chceme použít dynamické programování:
  1. podproblémy
  2. rekurzivní vztah pro optimální hodnotu
  3. pořadí počítání podproblémů
  4. sestavení řešení

Dynamické programování

Zalamování řádků v odstavci

Mějme text skládající se z n slov, šířka $i$tého slova je w_i pixelů. Chceme text vypsat do odstavce tak, aby byl zarovnán do bloku a na každém řádku byly co nejmenší mezery (mezery musí být minimálně 1 pixel). Maximální délka řádku je m pixelů. Každému řádku kromě posledního je určena penalizace

$$p_{i,j} = (m - j + i - \sum_{k=i}^{j} w_k)^3$$

kde i a j jsou indexy prvního a posledního slova na řádku. Cílem je minimalizovat součet penalizací všech řádků (kromě posledního).

- Chceme použít dynamické programování:
  1. podproblémy
  2. rekurzivní vztah pro optimální hodnotu
  3. pořadí počítání podproblémů
  4. sestavení řešení

Dynamické programování

Zalamování řádků v odstavci

Mějme text skládající se z n slov, šířka $i$tého slova je w_i pixelů. Chceme text vypsat do odstavce tak, aby byl zarovnán do bloku a na každém řádku byly co nejmenší mezery (mezery musí být minimálně 1 pixel). Maximální délka řádku je m pixelů. Každému řádku kromě posledního je určena penalizace

$$p_{i,j} = (m - j + i - \sum_{k=i}^{j} w_k)^3$$

kde i a j jsou indexy prvního a posledního slova na řádku. Cílem je minimalizovat součet penalizací všech řádků (kromě posledního).

- Chceme použít dynamické programování:
  1. podproblémy
  2. rekurzivní vztah pro optimální hodnotu
  3. pořadí počítání podproblémů
  4. sestavení řešení

K zamyšlení: Co kdyby penalizace byla lineární místo kubické?

Dynamické programování

Firemní večírek

Firma ZřetězenýSeznam.cz pořádá firemní večírek k příležitosti 42. narozenin jejího zakladatele a CEO. Zaměstnanci firmy jsou uspořádáni do stromové hierarchie. Specialisté z HR oddělení znají pro každého zaměstnance z jeho míru zábavnosti f_z. Dále pak zjistili, že není vhodné, aby se večírku účastnil libovolný zaměstnanec zároveň se svým přímým nadřízeným. Samozřejmě je ovšem nutné, aby se večírku účastnil sám CEO (kořen stromu). Sestavte seznam hostů, který bude respektovat tato omezení a zároveň maximalizovat součet jejich zábavnosti.

- Jak zformulujeme zadání formálně?

Dynamické programování

Firemní večírek

Firma ZřetězenýSeznam.cz pořádá firemní večírek k příležitosti 42. narozenin jejího zakladatele a CEO. Zaměstnanci firmy jsou uspořádáni do stromové hierarchie. Specialisté z HR oddělení znají pro každého zaměstnance z jeho míru zábavnosti f_z. Dále pak zjistili, že není vhodné, aby se večírku účastnil libovolný zaměstnanec zároveň se svým přímým nadřízeným. Samozřejmě je ovšem nutné, aby se večírku účastnil sám CEO (kořen stromu). Sestavte seznam hostů, který bude respektovat tato omezení a zároveň maximalizovat součet jejich zábavnosti.

- Jak zformulujeme zadání formálně?

- Chceme použít dynamické programování:
  1. podproblémy
  2. rekurzivní vztah pro optimální hodnotu
  3. pořadí počítání podproblémů
  4. sestavení řešení

Dynamické programování

Obarvení stromu

Mějme kořenový binární strom s n vrcholy a přirozené číslo $k \leq n$. Chceme obarvit k vrcholů stromu tak, aby každý vrchol měl obarveného co nejbližšího předchůdce. Obarvení chápeme jako podmnožinu vrcholů stromu K. Pro každý vrchol definujeme cenu obarvení jako:

$$cena(v, K) = \begin{cases} 0 & \text{pokud } v \in K \\ \infty & \text{pokud } v \text{ je kořen a } v \notin K \\ 1 + cena(parent(v), K) & \text{jinak} \end{cases}$$

Celková cena obarvení pak je $cena(K) = \max_v cena(v, K)$.
Chceme najít obarvení s minimální cenou.

Poznámka: Bude nám stačit řešení se složitostí $\mathcal{O}(n^2k^2)$.

Dynamické programování

Rekonstrukce textu bez mezer
Máme text, ze kterého zmizely všechny mezery a interpunkce. Chceme zrekonstruovat jeho (alespoň jednu možnou) původní podobu. K dispozici máme slovník, kterého se pomocí funkce $isWord(w)$ můžeme dotazovat, zda je daný řetězec platné slovo.

Dynamické programování

Hra o největší součet
Dva hráči hrají následující hru: Na stole je řada n karet s čísly. V každém kole hráč vezme jednu kartu buď z levého nebo z pravého konce řady. Hráči se střídají. Na konci vyhrává ten hráč, jehož součet čísel na kartách je větší.
Chceme určit, který z hráčů vyhraje, za předpokladu, že oba hrají co nejlépe.

- Uvažujme strategii: „Vždy vezmu větší ze dvou krajních karet.“ Je tato strategie optimální?