Aproximační algoritmy

Aprox. algoritmy jsou vhodné tam, kde je nalezení optimálního řešení „beznadějné“ (časově příliš náročné), typicky u NP-těžkých optimalizačních úloh (optimalizačních verzí NP-úplných rozhodovacích problémů). Mají následující tři vlastnosti:

1. konstruují suboptimální řešení

2. poskytují odhad kvality zkonstruovaného řešení vzhledem k optimu

3. běží v polynomiálním čase (jinak nejsou zajímavé)

Příklad maximalizační úlohy (optimalizační verze KLIKY):

Pro daný neorientovaný graf najdi největší (počtem vrcholů) kliku (úplný podgraf).

Po aproximačním algoritmu chceme garanci typu f(APROX) ≥ ¾ f(OPT), kde f(X) je v tomto případě počet vrcholů (tj. velikost kliky) v řešení X, OPT je optimální řešení a APROX je řešení vydané aproximačním algoritmem.

Příklad minimalizační úlohy (optimalizační verze ROZ):

Pro dané úkoly a daný počet strojů najdi nejkratší rozvrh.

Po aproximačním algoritmu chceme garanci typu f(APROX) ≤ 2 f(OPT).

Definice: Poměrová chyba aproximačního algoritmu je definována jako poměr (podíl) f(APROX) / f(OPT) pro minimalizační úlohy a f(OPT) / f(APROX) pro maximalizační úlohy. Relativní chyba je pak definována jako |f(APROX) – f(OPT)| / f(OPT).

Naivní aproximační algoritmus FRONTA pro optimalizační verzi ROZ: bere úkoly postupně podle jejich čísel a každý úkol vždy umístí na stroj, který je volný nejdříve.

Značení: OPT = optimální rozvrh, Q = rozvrh zkonstruovaný algoritmem FRONTA, délka(OPT) = o, délka(Q) = q

Věta: Pokud m je počet strojů, tak $q \leq ((2m - 1) / m)o$ a tento odhad již nelze zlepšit.

Důsledek: Aproximační algoritmus FRONTA má poměrovou chybu 2.

Důkaz:

1. Těsnost odhadu: Pro každé m zkonstruujeme zadání, pro které platí v dokazované nerovnosti rovnost, a to následujícím způsobem

   $x_1 = x_2 = \ldots = x_{m-1} = m-1$ ($m-1$ úkolů délky $m-1$)

   $x_m = x_{m+1} = \ldots = x_{2m-2} = 1$ ($m-1$ úkolů délky 1)

   $x_{2m-1} = m$ (1 úkol délky m)

2. Platnost nerovnosti: Nechť j je úkol končící jako poslední v rozvrhu Q (končící v čase q) a nechť t je okamžik zahájení úkolu j. Potom žádný procesor nemá prostoj před časem t a platí $mq \leq (2m - 1)o$.

Lepší aproximační algoritmus USPOŘÁDANÁ FRONTA pro optimalizační verzi ROZ: pracuje stejně jako FRONTA, ale na začátku úkoly setřídí do nerostoucí posloupnosti podle jejich délek.

Značení: OPT = optimální rozvrh,
U = rozvrh zkonstruovaný algoritmem USPOŘÁDANÁ FRONTA,
délka(OPT) = o, délka(U) = u

Věta: Pokud m je počet strojů, tak $u \leq ((4m - 1) / 3m)o$ a tento odhad již nelze zlepšit.

Důsledek: Aproximační algoritmus USPOŘÁDANÁ FRONTA má poměrovou chybu 4/3.

Důkaz: Těsnost odhadu: Pro každé liché m zkonstruujeme zadání, pro které platí v dokazované nerovnosti rovnost, a to následujícím způsobem

$x_1 = x_2 = 2m-1$ (2 úkoly délky $2m-1$)

$x_3 = x_4 = 2m-2$ (2 úkoly délky $2m-2$)

$x_{2m-3} = x_{2m-2} = m+1$ (2 úkoly délky $m+1$)

$x_{2m-1} = x_{2m} = x_{2m+1} = m$ (3 úkoly délky m)

Lemma: Pokud pro všechny úkoly platí $x_i \geq 1/3o$ pak $u = o$.

Dokončení důkazu: Nechť j je úkol končící jako poslední v rozvrhu U (končící v čase u). Pokud $x_j > 1/3o$ tak použijeme Lemma, v opačném případě je důkaz velmi podobný jako pro algoritmus FRONTA.

Pravděpodobnostní (randomizované) algoritmy

pravděpodobnostní algoritmus dělá (na rozdíl od deterministického algoritmu) náhodné kroky, např. k některým krokům používá hodnoty získané z generátoru náhodných čísel

tím pádem dvě různá spuštění téhož pravděpodobnostního algoritmu na stejných datech mají (s velkou pravděpodobností) různý průběh

pravděpodobnostních algoritmů je mnoho typů, zde zmíníme jen dva a to algoritmy typu Las Vegas a typu Monte Carlo

Algoritmy typu Las Vegas

výsledek je vždy správný, náhodnost ovlivňuje pouze dobu běhu algoritmu, tj. po jaké cestě se algoritmus ke správnému výsledku dobere

Příklad: randomizovaný QuickSort – od deterministické verze se liší náhodnými výběry pivota při každém dělení posloupnosti, což poskytuje následující výhody

- dává dobrý průměrný čas (tj. $O(n \log n)$) i v případě, že data na vstupu nejsou náhodné permutace – žádný vstup není apriori špatný (pro každý deterministický výběr pivota existují apriori špatné vstupy)
- může být spuštěn paralelně v několika kopiích, výsledek je získán z kopie, kde výpočet skončí nejdříve (pro deterministickou verzi nemá takový postup žádný smysl)