Algorytmy aproksymacyjne

Dominik Lau

2 stycznia 2023

1 Podejścia w rozwiązywaniu problemów NP-trudnych

Prawie na pewno rozwiązanie - algorytm wielomianowy prawie zawsze znajdzie dobre rozwiązanie (tak często jak sobie tego zażyczymy)

Prawie zawsze szybko - algorytm w średnim wypadku wielomianowy, zawsze znajdzie rozwiązanie ale czasem będzie potrzebował czasu wykładniczego

Prawie optymalne rozwiązanie - działa w czasie wielomianowym, zawsze zwraca jakiś wynik, niezawsze poprawny (oddalony od właściwego o pewną wielkość)

2 Definicje

2.1 Algorytm k-absolutnie aproksymacyjny

Jest to taki algorytm A, który dla pewnych danych I gwarantuje

$$|A(I) - OPT(I)| \leq k$$

2.2 Algorytm k(względnie)-aproksymacyjny

Jest to taki algorytm A, który dla pewnych danych I gwarantuje dla problemu minimalizacyjnego

$$\frac{A(I)}{OPT(I)} \leq k$$

a dla maksymalizacyjnego

$$\frac{OPT(I)}{A(I)} \leq k$$

2.3 Schemat aproksymacyjny

dla problemu minimalizacyjnego algorytm A jest schematem aproksymacyjnym jeśli dla każdego $\varepsilon > 0$ zachodzi

$$\frac{A(I)}{OPT(I)} \leq 1 + \varepsilon$$

po ludzku: mamy algorytm aproksymacyjny, którego dokładność możemy regulować (tylko, że czym większa dokładność, tym większa złożoność algorytmu)

PTAS - schemat wielomianowy względem rozmiaru problemu, np. $O(n^{\frac{1}{\varepsilon}})$
FPTAS - schemat wielomianowy względem rozmiaru i $\frac{1}{\varepsilon}$, np. $O((\frac{1}{\varepsilon})^2 n)$

3 Przykłady algorytmów aproksymacyjnych

NaiveColor - przybliżone kolorowanie krawędzi

```
def NaiveColor(G):
   c = 1
   for e in E(G):
      C = set{1..c}
      C.usun_nielegalne_kolory(e)

      if C.empty():
         c+=1
         color(e) = c
      else:
         color(e) = pick_any(C)
```

dla każdej krawędzi przeglądamy maksymalnie 2Δ kolorów innych krawędzi (a z tw.Vizinga $\Delta \leq \chi' \leq \Delta + 1$), więc pomylimy się góra dwukrotnie zatem jest to algorytm 2-aproksymacyjny

NaiveCover - przybliżone pokrycie wierzchołkowe

```
def NaiveCover(G):
   C = set()
   while not E.empty():
      {u,v} = pick_any(E)
      C.add(u) # !!!
      C.add(v) # !!!
      E.remove_incident(u)
      E.remove_incident(v)
   return C
```

dla wybranej krawędzi umieszczamy w pokryciu oba wierzchołki (zamiast jednego), więc w najgorszym przypadku umieścimy w pokryciu dwa razy za dużo wierzchołków - algorytm jest 2-aproksymacyjny

kolorowanie krawędzi z tw. Vizinga

```
def ChromaticIndex(G):
    return Delta(G)
```

jest to algorytm 1-absolutnie aproksymacyjny (bo czasem się pomylimy o jeden kolor) oraz $\frac{4}{3}$-aproksymacyjny

kolorowanie wierzchołkowe grafów planarnych

```
def ChromaticNumberPlanar(G):
    if empty(G): return 1
    if bipartite(G): return 2
    return 4
```

jest to algorytm 1-absolutnie aproksymacyjny - czasem mylimy się o jeden kolor (jeśli $\chi = 3$) oraz $\frac{4}{3}$-aproksymacyjny

przykłady FPTAS
Problem plecakowy: $O(n(logn + \frac{1}{\varepsilon^2}))$ oraz $O(nlog(\frac{1}{\varepsilon}) + \frac{1}{\varepsilon^4})$
Problem sumy podzbioru: $O(n + \frac{1}{\varepsilon^3})$

4 Dowodzenie nieistnienia PTAS