# I482F　実践的アルゴリズム特論
# 13,14回目：近似アルゴリズム

上原隆平
(uehara@jaist.ac.jp)

# ソートの下界の話

- 比較に基づく任意のソートアルゴリズムは$\Omega(n \log n)$ 時間の計算時間が必要である
  - 証明（概略）
    - $k$ 回の比較で区別できる場合の数は高々 $2^k$ 種類しかない
    - $n$ 個の要素の異なる並べ方は $n!$ 通りある
    - したがって少なくとも

$$2^k \geq n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$

が成立していなければならない。両辺の対数をとれば以下を得る。

$$k \geq \log \sqrt{2\pi n}\left(\frac{n}{e}\right)^n = n \log n - O(n) + \frac{1}{2}\log n + O(1)$$

# 超高速ソートの話

- 以下の特殊なソートを考える:
  - 入力: a[1]…a[n] で、それぞれの a[i] の値は 1〜10
  - 以下のアルゴリズムAは $O(n)$ 時間で動作する(!?)
    1. 配列 b[1]=b[2]=…=b[10]=0; // b[i] は a[j]=i を満たす要素の個数
    2. for i=1,2,…,n do b[a[i]]++;
    3. for j=1,2,…,10 do "j をb[j]個出力する".

このソートAは比較に基づいていない!!

Radix sort, bucket sort などと呼ばれるソートと同様のアイデア

データが「整数」など「特殊」な場合はこちらの方が速い!!

データに
<u>暗黙の仮定</u>
があれば
利用できる
かもしれない

# 典型的なNP完全問題KNAP…を高速に解く方法(?)

- KNAP:

  Input: アイテムの配列 a[1], …, a[$n$], 大きさ $k$

  Output: $\sum_{i \in S} a[i] = k$ を満たす集合$S \subseteq \{1,\ldots,n\}$が存在するか?

- アルゴリズムB: それぞれの $a[i]$ が正整数なら以下で解ける
  1. b[1]=b[2]=…=b[k]=0;
  2. for i=1,2,…,n do
     1. for j=1,2,…,k do
        1. if b[j]>0 then b[j+a[i]]=1;
  3. if b[t]>0 then "yes" else "no".

一般には $k$ の値が $n$ に対して
指数関数的に大きくなりうるので、
実は多項式時間アルゴリズムではない!!

b[]をリストにすれば、実数などでも
動作する。

- Bの実行時間は O($nk$) 時間

データが「整数」など「特殊」な場合や、
とりうる値の組合せの数(b[]の要素数)が
$n$ の多項式で押さえられるときは速い!!

# 典型的なNP完全問題KNAP…を高速に解く方法(?)

- KNAP:

  Input: アイテムの配列 a[1], …, a[$n$], 大きさ $k$

  Output: $\sum_{i \in S} a[i] = k$ を満たす集合$S \subseteq \{1,\ldots,n\}$が存在するか?

- 演習問題 : 次のアルゴリズムB'は何を計算しているのか?

  1. b[1]=b[2]=…=b[k]=0;
  2. for i=1,2,…,n do
     1. for j=1,2,…,k do
        1. if b[j]>0 then b[j+a[i]]=b[j+a[i]]+1;
  3. if b[t]>0 then “yes” else “no”.

# 近似アルゴリズム(Approximation Algorithm)

- 近似アルゴリズムの枠組:
  - 「Yes/No」タイプの決定問題を「最適化問題」に改造して考える。
    （注意）最適化問題は「最小化問題」と「最大化問題」がある
- 例:
  - VC（頂点被覆）: 大きさ最小の頂点被覆を見つける
  - MaxSAT: 与えられた論理式のうち、なるべく多くの項を “True” にする
  - 巡回セールスマン問題: すべての都市をめぐる最小コストの経路を探す
    （グラフを「辺に重み（コスト）のついたグラフ」にして、全経路を通れることにする）
  - KNAP: 大きさ $k$ 以下の組合せの中で最大のものを見つける
- 近似アルゴリズムの良さは「近似率」（と計算時間）で測る:
  - 最適な解を O* として、アルゴリズムの出力を O とすると、
    最小化問題の近似率=O/O*≧1
    最大化問題の近似率=O*/O≧1
  - 近似率はいつでも1以上で、近似率1のアルゴリズムは誤差のないアルゴリズム。

# 近似アルゴリズム(Approximation Algorithm)

- アルゴリズムの良さを「近似率」(と計算時間)で測る:
  - $\mathcal{NP}$困難問題(を最適化問題に翻訳したもの)は典型的には以下の3つのタイプに分類できる

1. 定数倍近似すら困難なもの(クラス$\mathcal{APX}$: この授業では扱わない)
   1. “$\mathcal{P}$=$\mathcal{NP}$が成立しない限り定数倍近似は存在しない問題”など

2. 適当な定数に対して定数倍近似が可能なもの(2倍近似アルゴリズムなど)

3. 任意の正定数 ε>0 に対して以下の条件を満たすアルゴリズムが存在する:
   1. $n$ と $1/\varepsilon$ に対する多項式時間で動作する
   2. $(1+\varepsilon)$近似解を出す

   このアルゴリズム(群)を多項式時間近似スキーム
   (PTAS; Polynomial Time Approximation Scheme)と呼ぶ

# 近似アルゴリズム(Approximation Algorithm)

2. 2倍近似アルゴリズムの例
   - **頂点被覆問題**(VC)の最適化バージョン

     入力：無向グラフ$G=(V,E)$

     出力：最小の頂点被覆 $S$
   - アルゴリズムC:
     1. $S:=\Phi$;
     2. $G$ の辺 $e=\{u,v\}$を適当に1本選ぶ
     3. $u$ と $v$ を $S$ に入れる
     4. $u,v$ につながっている辺を $G$ からすべて削除する
     5. $G$ に辺が残っていれば 2 に戻る

なぜか2倍を切れる
かどうかが難しい
問題が多い

$S$は$G$の「頂点被覆」:
どの辺$\{u,v\}$に対しても
$u\in S$または$v\in S$が成立

線形時間で動作
するのは簡単なの
で省略

[定理13.1] アルゴリズムCの実行時間は$O(|V|+|E|)$時間である。また $G$ の最適な頂点被覆を $S^*$ とし、
アルゴリズムCの出力を $S$ とすると、以下が成立：

$$|S|/|S^*|\leqq 2$$

# 近似アルゴリズム(Approximation Algorithm)

## 2. 2倍近似アルゴリズムの例

- アルゴリズムC:
  1. $S$:=Φ;
  2. $G$ の辺 $e=\{u,v\}$を適当に1本選ぶ
  3. $u$ と $v$ を $S$ に入れる
  4. $u,v$ につながっている辺を $G$ からすべて削除する
  5. $G$ に辺が残っていれば 2 に戻る

$S$は$G$の「頂点被覆」:
どの辺$\{u,v\}$に対しても
$u \in S$または$v \in S$が成立

[定理13.1] $G$ の最適な頂点被覆を $S^*$ とし、
アルゴリズムCの出力を $S$ とすると、以下が成立: $|S|/|S^*| \leqq 2$

[証明] ステップ2で選ばれた辺 $e$ の集合を $C$ とおく。

$e$ はステップ4で削除されるため同じ辺が2度選ばれることはない。∴$2|C|=|S|$。

$C$ は頂点を互いに共有しない辺の集合で、$e \in C$ のそれぞれについて
少なくとも一方の端点は $S^*$ に入っていなければならない。∴$|C| \leqq |S^*|$

したがって $|S|/|S^*| \leqq 2|C|/|C|=2$ である。

# 近似アルゴリズム(Approximation Algorithm)

2. **2倍近似アルゴリズムの例**
   - アルゴリズムC:
     1. $S$:=Φ;
     2. $G$ の辺 $e=\{u,v\}$を適当に1本選ぶ
     3. $u$ と $v$ を $S$ に入れる
     4. $u,v$ につながっている辺を $G$ からすべて削除する
     5. $G$ に辺が残っていれば 2 に戻る

$S$は$G$の「頂点被覆」:
どの辺$\{u,v\}$に対しても
$u \in S$または$v \in S$が成立

[定理13.1] $G$ の最適な頂点被覆を $S^*$ とし、
アルゴリズムCの出力を $S$ とすると、以下が成立: $|S|/|S^*| \leqq 2$

[演習問題] アルゴリズムCは2倍近似アルゴリズムであることが証明された。

C の近似率2はこれ以上改善できないことを示せ。
具体的に、無限に多くの $n$ に対して、C の近似率がちょうど 2 であるような
$n$ 頂点グラフの例を示せ。

# 近似アルゴリズム(Approxima

[アイデア] 個々の値を
それに近い値に丸めて、
値の種類を減らす

3. 多項式時間近似スキームの例
- KNAPの最適化バージョン

Input: アイテムの配列 a[1], …, a[$n$], 大きさ $k$

Output: $\sum_{i \in S} a[i] = k_S < k$ を満たす集合$S \subseteq \{1,\ldots,n\}$で もっとも近いもの

- アルゴリズムD (アルゴリズムBも参照):
  1. L:=Φ; //実現できる和の近似値のリスト
  2. for i=1,2,…,
     1. L のそれぞれの要素 b に対して、b+a[i]≦$k$ ならそれを L に登録
     2. L のいくつかの要素を「丸め」て、不要なら捨てる
  3. L の中の $k$ 以下のもっとも大きな値 $k'$ を出力する

[ポイント] Dで以下の2点が満たされればよい：
1. Lのサイズがいつでも $n$ の多項式
2. 出力 $k'$ がよい近似解を与える

# 近似アルゴリズム(Approximation Algorithm)

3. 多項式時間近似スキームの例
   - KNAPの最適化バージョン

     Input: アイテムの配列 a[1], …, a[$n$], 大きさ $k$

     Output: $\sum_{i \in S} a[i] = k_S < k$ を満たす集合$S \subseteq \{1,\ldots,n\}$で もっとも近いもの

   - アルゴリズムD (アルゴリズムBも参照):

[仮定]
- L が小さい順で並んでいるとする
- 正の定数εを固定する

[丸めの詳細]
- Lの中で $b_1 < b_2 \leqq (1+\varepsilon/2n)\, b_1$ なら $b_2$ を削除する

[定理13.2] アルゴリズムDはPTASである。
つまり任意の正定数εに対して以下が成立する:
1. $n$, $1/\varepsilon$ の多項式で動作する
2. 近似率は $(1+\varepsilon)$

[証明] 1,2 ともにちょっと計算が必要…

# 近似アルゴリズム(Approximation Algorithm)

- 補題13.1: 自然数列 $a_1=1, a_2, \ldots, a_n=k$ が $(a_{i+1})/a_i \geqq 1+t$ を満たすなら、次が成立: $n < \frac{1+t}{t}\ln k$

  [証明] $(1+t)^n < k$ より、$n < \log_{t+1} k$である。公式 $\frac{x}{1+x} \le \ln(1+x) \le x$ を適用すると以下を得る。

$$n < \log_{t+1} k = \frac{\ln k}{\ln(1+t)} \le \frac{(1+t)}{t}\ln k$$

- 補題13.2: 0<ε <1に対して $\left(1+\frac{\varepsilon}{2n}\right)^n \le 1+\varepsilon$

  [証明] 以下の公式をつかう。

  [公式1] $\lim_{n\to\infty}\left(1+\frac{x}{n}\right)^n = e^x$ (この式は $n$ に対して単調増加関数)

  [公式2] $|x| \leqq 1$ならば $1+x \le e^x \le 1+x+x^2$

  以上より $\left(1+\frac{\varepsilon}{2n}\right)^n \le e^{\varepsilon/2} \le 1+\frac{\varepsilon}{2}+\left(\frac{\varepsilon}{2}\right)^2 \le 1+\varepsilon$

# 近似アルゴリズム(Approximation Algorithm)

- KNAPの最適化バージョンの多項式時間近似スキーム

[定理13.2] アルゴリズムDはPTASである。
つまり任意の正定数εに対して以下が成立する:
1. $n$, $1/\varepsilon$ の多項式で動作する
2. 近似率は $(1+\varepsilon)$

[証明] 1) Lのサイズが $n$, $1/\varepsilon$ の多項式で抑えられればよい。
ここでＬの要素列 $b_1, b_2, \ldots$ は、$1 \leqq b_1$, $b_i \leqq k$, $b_{i+1}/b_i > (1+\varepsilon/2n)$ を満たす。
よって補題13.1より、

$$\text{Lのサイズ} < \log_{(1+\varepsilon/2n)} k = ((2n+\varepsilon) \log k)/\varepsilon < (2n \log k)/\varepsilon$$

となる。

# 近似アルゴリズム(Approximation Algorithm)

- KNAPの最適化バージョンの多項式時間近似スキーム

[定理13.2] アルゴリズムDはPTASである。
つまり任意の正定数εに対して以下が成立する:
1. $n$, $1/\varepsilon$ の多項式で動作する
2. 近似率は $(1+\varepsilon)$

[証明] 2) アルゴリズムの出力 $k'$ を構成する $a[]$ の要素集合が存在する。
この集合を $S'$ とする。
つまり次が成立する: $\sum_{a[]\in S'} a[] = k'$
ここで入力に対する
最適な集合を $S^*$ とし、$\sum_{a[]\in S^*} a[] = k^*$ とする。
$S^*$ のそれぞれの
$a[]$ に対しては、それが L に存在しているか、それを代替したものがあるはずである。代替されている場合、最悪だと $a[]$ は以下の値 $a'$ で代替されている。

$$\frac{a[]}{(1+\varepsilon/2n)^n} < \frac{a[]}{(1+\varepsilon/2n)^{n-1}} \le a' < a[]$$

よって
$k^*/(1+\varepsilon/2n)^n \leqq k'$ が成立し、補題13.2より $k^*/k' \leqq 1+\varepsilon$ を得る。