数理計画法E（第6学期）
第12回

担当：飯田勝吉（いいだかつよし）
iida@gsic.titech.ac.jp

分枝限定法（3）再掲

- ナップザック問題（元の問題）

目的関数： $7x_1+8x_2+x_3+2x_4 \rightarrow$ 最大化

制約条件： $4x_1+5x_2+x_3+3x_4 \le 6$

$x_i \in \{0,1\}$ 0か1のみを許す

- 連続緩和問題 ＝元の問題の0-1条件を緩和

目的関数： $7x_1+8x_2+x_3+2x_4 \rightarrow$ 最大化

制約条件： $4x_1+5x_2+x_3+3x_4 \le 6$

$0 \le x_i \le 1 \quad (1 \le i \le 4)$ 0～1の任意の実数を許す

連続緩和問題の解法 ～欲張り法～

- まず、アイテムを「単位重さあたりの利用価値」の順に並べる

$$\frac{c_1}{a_1} \ge \frac{c_2}{a_2} \ge \cdots \ge \frac{c_n}{a_n}$$

- ナップザック問題の場合
  - アイテムを入れるか入れないかの 0 or 1
- 連続緩和問題の場合
  - 重さ制限が許す場合は、アイテムを入れる (xi=1)
  - 重さ制限が許さない場合は、アイテムを部分的に入れる (0 < xi < 1)

組み合わせ最適化問題

- 欲張り法
  - アルゴリズムの途中過程で、近視眼的なローカル最適化を行うことで、問題全体のグローバル最適化を行おうとする方法の総称
  - 欲張り法の適用範囲は限られている
- 分枝限定法
  - 解空間の全ての候補を総当りで検索する際に、不必要な枝を先に刈り取ることで、計算量を削減する方法
- 動的計画法 (dynamic programming)
  - 問題全体の解を求めるために、小さい問題の最適解を求め、次第に拡張していく方法
- 近似解法 (heuristic methods)
  - 最適解の探索が困難な問題に利用する方法

列挙法

- 有限個の組み合わせ最適問題の解のすべてを列挙し、最適解を求める総当り探索を原則とする方法
  - 分枝限定法
    - 不必要な枝の探索をやめることで効率を向上
  - 動的計画法（dynamic programming)
    - 再帰的に部分問題の解から全体の解に拡張していく方法

動的計画法1

- 最適性の原理
  - ある問題とそれを分解した部分問題を考えると、元の問題の最適解は部分問題の最適解を含む
- 動的計画法
  - 最適性の原理を利用して組み合わせ最適化問題を解く手法
  - 例：ダイクストラ法
    - あるネットワークの最短路木Tにおいて、Tの根から最遠点Pを除いた木T'は、そのネットワークからPを除いた部分ネットワークの最短路木である。
    - 最短路木T'を求めるため、T'の根から最遠点P'を除いたネットワークにおける最短路木T''を求める・・・

動的計画法2

- [問題4.3] [資源配分問題] 余剰資金を3つの投資対象に投資したい。各投資先に投資したときに見込まれる利益は下表のとおりである。資金が5単位あるときの利益を最大化する投資配分法を求めよ。

投資額(単位)	1	2	3	4	5
投資先1	2	4	6	8	10
投資先2	1	2	5	9	11
投資先3	3	4	5	6	7

動的計画法3

- 資源配分問題の定式化
  - 投資先の個数=n
  - 余剰資金=Z
  - 投資先 i に xi 単位投資したときの利益= $f_i(x_i)$

目的関数： _____ → 最大化

制約条件： _____

_____

動的計画法4

- 資金配分問題の最適性
  - 投資先1～nに対してZ単位の資金を投資したときに得られる最大利益をFn(Z)と定義

$$F_n(Z) = \max_{0 \le x_n \le Z} \left(f_n(x_n) + F_{n-1}(Z - x_n) \right) \quad (n > 1 \text{のとき})$$
$$F_1(Z) = \max_{0 \le x_1 \le Z} f_1(x_1) \quad (n = 1 \text{のとき})$$

  - 資金配分問題
    - Fn(Z)とそれを実現する(x1,x2, x3)の組を求める問題
    - F1, F2, F3の順に決定していけば最適解を導出可能

動的計画法5

- 実際に問題4.3を動的計画法で解く
- (1) 0≦x1≦5より、F1(0), …., F1(5)を求める。F1(x1)は単調増加であるから

$$F_1(Z) = \max_{0 \le x_1 \le Z} f_1(x_1) = f_1(Z)$$

  - よって、F1(0)=0, F1(1)=2, F1(2)=4, F1(3)=6, F1(4)=8, F1(5)=10
- (2) $F_2(Z) = \max_{0 \le x_2 \le Z} \left(f_2(x_2) + F_1(Z - x_2) \right)$を利用してF2(0), …, F2(5)を求める

動的計画法6

- F2(0)=f2(0)+F1(0)=0
- F2(1)=max(f2(0)+F1(1), f2(1) + F1(0))
  =max(0+2, 1+0) = 2
- F2(2)=max(f2(0)+F1(2),f2(1)+F1(1), f2(2) + F1(0))
  =max(0+4, 1+2, 2+0) = 4
- F2(3)=max(f2(0) + F1(3), f2(1) + F1(2), f2(2)+F1(1), f2(3)+F1(0))
  =max(0+6, 1+4, 2+2, 5+0) = 6

動的計画法7

- F2(4)=max(f2(0)+F1(4), f2(1)+F1(3), f2(2)+F1(2), f2(3)+F1(1), f2(4)+F1(0))
  =max(___, ____, ____, ___, ___) = __
- F2(5)=max(_____)

=max(_____)= __

動的計画法8

- (3)$F_3(Z) = \max_{0 \le x_3 \le Z} \left(f_3(x_3) + F_2(Z - x_3) \right)$を利用して F3(0), …, F3(5)を求める。
- F3(0) = _____
- F3(1) = _____
  = _____
- F3(2) = max(f3(0)+F2(2),f3(1)+F2(1)+f3(2)+F2(0))
  =max(0+4, 3+2, 4+0)=5

動的計画法9

- F3(4)=max(f3(0)+F2(4), f3(1)+F2(3), f3(2)+F2(2), f3(3)+F2(1), f3(4)+F2(0))
  =max(0+9, 3+6, 4+4, 5+2, 6+0)=9
- F3(5)=max(f3(0)+F2(5), f3(1)+F2(4), f3(2)+F2(3), f3(3)+F2(2), f3(4)+F2(1), f3(5)+F3(0))
  =max(0+11,3+9,4+6,5+4,6+2,7+0)=12

動的計画法10

- (4) F3の最大値はF3(5)=12で、このとき x3=1, F2(4)=9。F2(4)=9となるのは、x2=___, _____のとき。F1(0)=0となるのは、x1=0のとき
- したがって(x1,x2,x3)=(0, 4, 1)が最適解
- 最大利益＝12

動的計画法11

- 動的計画法
  - 分枝限定法のように不必要な枝の探索をしなくなるわけではない
  - 大きな問題の全探索を再帰的に行う方法
  - 空間計算量が大きくなる代わりに、時間計算量を短縮する方法（問題によっては大幅に短縮）
  - 最適化組み合わせ問題の多くに適用可能なため、研究上、利用することが多い

近似解法（1）

- 組み合わせ最適化問題
  - 厳密な最適解を求めるためには非現実的な時間が必要なものが多い
  - 時間計算量による分類
    - クラスP: 多項式時間で答えが求まる問題の集合
    - クラスNP: 答えを求めるには指数時間がかかるが、事前にその答えがあたえられているとき、その答えを検証するのは多項式時間でできる問題の集合
    - クラスNP困難：クラスNPに含まれるもんだいよりも難しいか、同等に難しい問題からなる集合（例：巡回セールスマン問題）

近似解法（2）

- NP困難問題に大きなパラメータを入力すると、最適解を得るのに非現実的な時間がかかる
- そのような問題は、ヒューリスティック解法（発見的解法）により、近似最適解を得ることができる。
- たとえば、欲張り法によりナップザック問題、巡回セールスマン問題の近似解の探索が可能。

メタヒューリスティックス（1）

- Meta heuristics
- 特定の計算問題によらず、どの問題に対しても適用可能な汎用的な近似解法の枠組み
- 数値結果が必要で現実的に最適解を得られない問題で多く利用される

メタヒューリスティックス（2）

- 局所探索法（local seearch)
  - まず、初期解を与え、（欲張り法、ランダム等）
  - 現在の解の近くにある_____の目的関数を計算し、現在の解よりも目的関数を向上する解があれば、それを現在の解とし、反復する。
  - 終了条件（前回の解との差が閾値以下になるなど）を満たせば終了

メタヒューリスティックス（3）

- 局所探索法
  - 近傍の定義のしかた、複数ある近傍の中で次の解をどのように選ぶか、などによって計算量や近似最適解のよしあしが変わる
  - つまり、欠点は_____
- 欠点を補うため
  - 初期解を変化させて何度も局所探索法を繰り返す
  - あるいは、多少の改悪を認めて近似をよくしていく方法

メタヒューリスティックス（4）

- 焼きなまし法 (Simulated annealing法)
  - 探索の各繰り返しにおいて、現在の解の近傍からランダムに解を探索
  - 改良になるならば即、その解を現在の解とし、
  - 改悪になる場合は、改悪の度合いに応じて確率的にそれを次の最適解にするかどうかを決める
  - 繰り返しの回数を金属の焼きなましの温度と考え、初期段階では改悪の度合いの確率を高くし、徐々に確率を下げていく

メタヒューリスティックス（5）

- タブー探索法 (tabu search)
  - 局所探索法において、局所最適解に落ち込んでしまうことを避けるために、近傍解の中で現在の解より改良となる解が見つからない場合、最も改悪の量が少ない解を現在の解とする
  - しかし、これによっていくつかの解が繰り返し現れ、エンドレスになる可能性がある
  - そこで、すでに近似最適解として選ばれた解のリストを保存し、いまだ選んでいない解の中でもっともよい解へ移動する

メタヒューリスティックス（6）

- タブー探索法
  - すでに選ばれていてもう選ぶことのできない解のリスト＝_____
  - このリストは無制限に大きくなるので、LRUなどのアルゴリズムで古いものを新しいものと置き換えていく

メタヒューリスティックス（7）

- ほかの方法
- 遺伝的アルゴリズム(Genetic Algorithm)

- メタヒューリスティックスの特徴
  - 問題によって、よりよいアルゴリズムやそのパラメータが変わる
  - 理論的取り扱いが難しく、計算実験を繰り返し、経験的にチューニングする、職人の世界

組み合わせ最適問題のまとめ1

- 欲張り法
  - もっとも単純（naive）な方法
  - 最短木問題などで有効
  - しかし多くの問題で最適解の探索が不能
  - そのような問題の場合、近似解探索として利用可能
- 分枝限定法
  - 列挙法のひとつで、すべてを探索するのではなく、不必要な枝を刈りながら効率的に探索する方法
  - ナップザック問題などで有効

組み合わせ最適問題のまとめ2

- 動的計画法
  - 列挙法のひとつで、最適性の原理を利用し、再帰的に小さな部分問題の最適解を全体の最適解に拡大する方法
  - 多くの最適問題に適用可能
- ヒューリスティックス
  - 発見的方法により、近似解を得る方法
- メタヒューリスティックス
  - ヒューリスティックスアルゴリズムの中で、汎用的にさまざまな問題に適用可能な方法