集合知プログラミング
第5章　最適化

06T4073R 三上健太

はじめに

- 最適化が利用される問題
  →多数の可能な解が存在する問題
  →変数の組み合わせによって結果が大きく変わる問題
- 最適解を探すとき→多くの異なる解を試しスコアリングを行う
  →それぞれの解の質を決定
- 解探索の手法
  →無作為な推測を数千程度行い、どれが最適かを見る手法
  →改善できるであろうやり方で解を改変していく手法

グループ旅行

- グループ旅行のプランニング
  →簡単な式を使って答えを出すことができない→最適化
- 考慮すること
  ・各者のフライトスケジュール
  ・必要なレンタカー台数
  ・トータルコスト
  ・空港での待ち時間　etc...
- 条件
  ・アメリカ中の様々な土地から同じ場所に向かう旅行
  ・全員が同じ日に旅行先に到着、別の同じ日に旅行先を出発
  ・空港からは車をシェアする→空港で待機　　etc...

解の表現

- 家族がそれぞれどのフライトを使うか決める
  - 運賃の総額を抑えることは目的のひとつ
  しかし・・・他にも考慮(最小化)したい要素はたくさんある
  ex.空港での待ち時間、飛行時間の合計
  →どのように解を表現するかが重要
- 最適化関数→多種多様な問題に対処できる一般性の高いもの
  解の表現→シンプルなもの　ex.数字のリスト

解の表現(2)

- スケジューリングで考慮すべき点
  →1つのものを考慮しすぎると他がおろそかになる
  →スケジュールの多様な属性に重み付けをしてバランスよく。
  全てを考慮してベストなものを決める！

コスト関数

- 最適化アルゴリズム
  →コスト関数を最小化するインプットの集合を発見
- コスト関数
  →問題を最適化して解決する為の鍵となる関数。決定が難しい
  →解がどの程度悪いか示す値を返す必要がある
  →悪い解には大きな値を返すようにする

コスト関数(2)

- 多くの変数にわたる解：改善 or 改悪する要素の特定が難しい

ex.

・運賃：財政状況を考慮した重み付け平均を使っても良い
・発時刻：あまりにも早朝の便に対してはコスト追加
・レンタカーの貸出期間：借りた時間より遅いと追加料金
・飛行時間
・待ち時間

etc...

コスト関数(3)

- 複雑な問題の最適解
  →どの要素が重要か定めることが必要
  →定めてしまえばほぼどんな問題にも対応可
- コストを生じる変数を数字に変換する方法を決定する
  ex.飛行時間や空港での待ち時間をドルに換算する
  →全ての変数をドル(運賃の単位)に換算

- コスト関数
  →正しい数字の組み合わせを選んでコストを最小化する
- 理論的には全ての組み合わせを試すことが可能
  →かなり時間がかかる

ランダムサーチ(無作為探索)

- 無作為に推測を行い最適解を探索する
- 1000回試行すれば、ひどくない解に出くわす可能性もあるが…
  →優れた最適化手法とはいえない

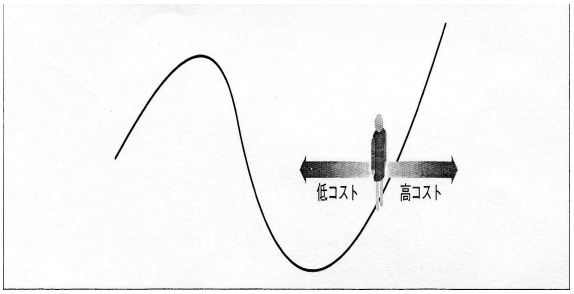
- 利点
  ・ベースラインにして他のアルゴリズムとの比較ができる

ヒルクライム

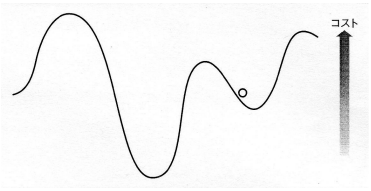
- ヒルクライム(傾斜登り)
  →無作為解から出発し、近傍解の中からより優れたものを探す
- ex.

ランダムなスケジュールを定める
↓
近傍のスケジュールを全て探索する
(ある人のスケジュールを前後に少しだけずらす)
↓
全ての近傍解についてコスト計算し、最小値を新しい解にする
↓
繰り返し(近傍解でコストが改善されなくなるまで)

ヒルクライム(2)



ヒルクライム(3)



- 全体的な最良解に達するとは限らない
  →最終解は局所最小
- 全体的な最良は大域最小
  →最適化アルゴリズムが見つけるべきはこれ！
- アプローチ法として無作為再出発ヒルクライムがある
  →無作為な出発点で何度もヒルクライムアルゴリズムを実行