データ構造とアルゴリズム I

九州大学大学院システム情報科学研究院
情報学部門
横尾 真
E-mail: yokoo@inf.kyushu-u.ac.jp
http://agent.inf.kyushu-u.ac.jp/~yokoo/

自己紹介

- 1986年東京大学大学院工学系研究科 電気工学専門課程 修士課程修了
- 同年 日本電信電話株式会社 (NTT) 入社
- NTT情報通信処理研究所 (神奈川県横須賀市), NTTコミュニケーション科学基礎研究所 (京都府相楽郡) 等に勤務
- 人工知能，マルチエージェントシステムに関する研究に従事
- 1995年 博士 (工学)，東京大学工学系研究科 電子情報工学専攻
- 2004年4月より九州大学システム情報科学研究院 教授，2012年より主幹教授

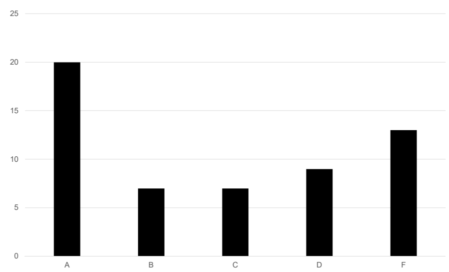
成績に関して

- 小テスト，期末試験，(レポート?) の成績で判断
- 出席は取るが，試験の成績が良ければ，出席率は問わない (小テストは受験するように!)
- 不合格ぎりぎりぐらいの場合は出席率も考慮するかも知れない

GPA制度(2007年度より導入)

- A：90-100点：4：特に優れている
- B：80-89 点：3：優れている
- C：70-79 点：2：普通である
- D：60-69 点：1：一応の学修成果があり、単位は認める
- F：59点以下：0：不合格

2016年度前期の成績



注意事項

- この講義はC課程向けで，A/B課程の学生の受講は推奨されていない（らしい）
- 一方，2016年度の後期のデータ構造とアルゴリズム I の講義を履修してF判定の人は，この講義を「再履修」しなくても，「再試験」を受ける権利がある（内容／時刻／場所は，この講義の定期試験と同じ）

予定 (I)

4/14: 第1回：イントロダクション
4/21: 第2回：アルゴリズムの解析
4/28: 第3回：関数の増加
5/2 (火，金曜日の講義日):
第4回：漸化式，確率論的解析
5/12: 小テスト
5/19: 第5回：乱択アルゴリズム
5/26: 休講
6/2: 前半まとめテスト

7

予定 (II)

6/9: 第6回：ヒープソート
6/16: 第7回：クイックソート
6/23: 第8回：線形時間ソーティング
6/30: 小テスト
7/7: 第9回：基本データ構造
7/14:第10回：ハッシュ表
7/21:第11回：2分探索木
7/28: 定期試験

8

講義の目的

目的：データ構造とアルゴリズムの基礎を身につける
- 電気情報工学科としての最低限の教養
- 身についていないと困る (アルゴリズムとデータ構造II, 卒論，大学院進学／就職後)

9

講義について

- パワーポイントのスライドを用いる
- 教科書 (近代科学社，コルメン，ライザーソン，リベスト，シュタイン著，アルゴリズムイントロダクション 第一巻，第三版) に準じて講義を進める
  - 旧版を持っているなら買い換えなくてもよい
- スライドはホームページで後日公開する
  - google等で"横尾　九州大学"
- 詳細なノートを取る必要はない (講義の内容に集中!)

10

自習方法

- 講義中で紹介したアルゴリズムに関して動作を良く理解し，使えるようになること
  - 小さな例題で，手を動かしてトレースする
    - トランプのカード等を使うのも良い
  - 自分で計算機に実装して動かす
    - プログラムする言語は好きなもの／慣れているものを選べば良い
    - 実装自体は簡単（インタラクションなし）
    - フリーの処理系は多い (GCC/G++, JAVA)

11

アルゴリズム設計

- 単なるプログラミングとは異なる
- アルゴリズムは言語や処理系に依存しない一般的なもの
- 良いアルゴリズムは広範囲に，非常に長い期間に渡って用いられる
- アルゴリズム設計は高度な知的作業
  - 面白い！
  - ただし，解析には数学的な知識が必要
  - 面白い課題が解けるようになるまでの勉強が大変

12

アルゴリズム設計の例

- 効率を気にしなければ，答えを見つける方法はすぐ思い付くが，効率的なアルゴリズムを見つけるのが難しい問題：
  安定結婚問題
- 問題の表現方法に工夫が必要な問題（ちゃんと表現できればすぐ解ける）：
  宣教師と人喰い人種
- どうやって答えを導くか自体が難しい問題：
  3人ケーキ分配問題

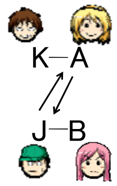
13

安定結婚問題

- 男性，女性がそれぞれ4人ずついる
- 各男性は4人の女性に対して，各女性は4人男性に対して，好みの順番が決まっている
- 好みの順番は，当然，人によって異なる（たまたま同じかも知れない）
- 簡単のため，同点はないものとする
- 8人から，不安定なペアが存在しないように，ペアを4組作りたい

14

不安定なペアとは?



- 女性：Alice, Becky, Carol, Daisy
- 男性：John, Ken, Lee, Mike
- Aliceにとって，J > K > L > M
- Johnにとって，A > B > C > D
- もしAliceのペアがKenで，JohnのペアがBeckyだと，AliceとJohnは，今のペアと別れてペアとなった方が二人ともより幸福
- このようなペアを不安定なペアと呼ぶ
- 不安定なペアを含まない組合せを安定マッチングと呼ぶ

15

安定マッチングを見つける

単純なアルゴリズム

- 男性の順序を，J, K, L, Mで固定
- 女性A, B, C, Dの並び替えをすべて生成し，J, K, L, Mと組み合わせて，安定かどうかをチェックする
- 最悪，4!の組合せをチェックすることになる
- 人数が増えると絶望的
- もっと効率的に安定マッチングを見つける方法はないか？

16

効率的なアルゴリズム (Gale & Shapley)

- 男性／女性は，独身，婚約中のどちらか
- 初期状態では全員独身
- 独身の女性が残っていれば，以下の処理を繰り返し適用
  - 独身の女性は，これまでにまだプロポーズをしていない男性のうちで，最も好みの男性にプロポーズする（男性が婚約中でも気にしない）．一人の男性には一回しかプロポーズできない
  - 男性は，現在婚約中の女性よりも，より良い相手がプロポーズしてきたら，現在の婚約を解消して，最も好みの女性と改めて婚約する
- 独身の女性がいなくなれば，現在婚約中のペアでマッチングを決定

17

Gale & Shapleyアルゴリズムの性質

- 各女性は，一人の男性に一回しかプロポーズできないので，繰り返しは高々4^2回 (n人ならn^2回) 程度で終了する

任意の時点で以下が成立

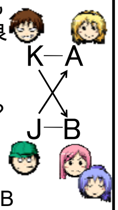
- 各女性にとって，婚約中の相手よりも望ましい男性には，すでに断られている
- 各男性にとって，今まで断った女性よりも，より望ましい女性と婚約している

よって，終了時のマッチングは安定である

18

Gale & Shapleyアルゴリズムの性質（続き）

- 女性にとっては，まだプロポーズしていない中で最も好みの男性にプロポーズするのが最適（正直が最良の策）
- 男性にとっては? 現在自分にプロポーズしている女性の中で，最も好みの女性を選ぶのが本当に最適?
  - 最適とは限らない！
  - 三人，JはA>B>Cで，B, Cがプロポーズしている．AはKにプロポーズしている．KはB>A，AはK>J，BはJ>K
  - JがBを選ぶとJ,Bのペアが決定，もしBを断ると，BはKにプロポーズして，Aが断られて自分に来る！
- 常に安定なマッチングを生成し，男性／女性の両方で正直が最良の策となるアルゴリズムは存在しない



19

安定結婚問題の応用事例

- 研修医と病院のマッチング
- 学校選択制
- 卒業研究の研究室の割り当て
- …

20

2012年ノーベル経済学賞

- ロイド・シャプレー & アルビン・ロス，マーケットデザインおよびマッチングに関する理論

21

ゲーム理論

- 安定結婚問題等で，プレイヤの戦略的行動の影響を解析するために用いられる理論がゲーム理論
- ゲーム理論は，ジョン フォン・ノイマンを創始者とする応用数学の一分野
- ミクロ経済学と計算機科学の両方の分野で研究が行われている
- 数学なので，仮定が正しいなら理論的な帰結は確実に正しい
- 現実の問題に適用する際には，仮定が妥当かどうかの検証が必要

22

例：Gale & Shapleyアルゴリズム

- 男女100名ずつのお見合いパーティーでGale & Shapleyアルゴリズムでペアを構成
- 理論的帰結は，安定なマッチングが得られ，女性にとっては正直が最良の策
- 上記が本当に成立するか? 現実には成立するかどうか怪しい前提が隠れていないか?

23

例：Gale & Shapleyアルゴリズム (続き)

- 前提：男性は，女性に関して，あらかじめ与えられた選好順序を持ち，その順序に従ってプロポーズしてきた女性から，婚約する女性を選ぶ
- 自分が男性だとして，以下の状況を考える
  - 最初の状態では，Aliceの方をBeckyより，ごくわずかに好んでいた
  - Beckyのみが最初からプロポーズしてくれていた
  - 一方，Aliceは他の98人の男性から断られ，99人目に自分にプロポーズしてきた
  - ここでAliceを選べるか?

24

例：Gale & Shapleyアルゴリズム (続き)

- 「自分を一番好きだと思っている (あるいは一番好きと言ってくれる) 人が好き」という感情は自然
- しかし，「一番好き」と言ってくれる人を優先するようにアルゴリズム／メカニズムを構築すると，嘘でも「一番好き」と言ってしまう誘因／インセンティブを与えてしまう
- 真実を知りたければ，自分にとって都合の悪い真実でも受け入れる覚悟が必要

25

卒研配属の重要性

- ほとんどの人は大学院の修士課程までは進学するので，合計6年間を大学で過ごす
- 前半の3年間と後半の三年間に，大きな違いがある（高校から大学への違いと同じくらい?）
- 大学は個人事業主（教員）の寄り合い所帯
- 各研究室によって，運営方針は全く異なる
- 研究室の選択は，大学の選択と同程度に重要
- 楽なところが（将来的に）良いとは言えない
- 自分の興味に合った，アクティブに活躍している研究室を選んだ方がよい
- 研究室に合う／合わないで，その後の人生が大きく変わる
  - 研究が楽しい → 就職もうまくいく／博士課程に進学
  - 研究がつまらない→就職がうまくいかない →引きこもり／留年

26

電気情報工学科での卒論配属

- 教員数は80名から90名ぐらい
- 多分，学生からは馴染みがない教員も多い
- 教授／准教授のペアで研究をしているところも多い
  - 実質，どちらに配属されても，ほとんど差が無い
  - しかし，人気が偏ったりする --- 調査不足
- Web等の情報で，どんな研究をしているかチェックすべき
  - 最低限，教員のホームページぐらいはチェックする
  - 学生からも教員を評価できる
  - とりあえずgoogle（なるべく英語）で検索
  - google scholarでの論文の被引用回数

27

卒論配属方式 (2010年まで)

- 学生側は自身の希望する研究室を第1希望から第n希望まで申告
- 希望順位優先方式
  1. 全学生の第1希望において定員を満たすまで成績順に配属
  2. 配属されていない学生と定員の残る研究室で次の希望順位について同様に繰り返す

28

従来方式の問題点

- 学生同士の読みあい
  - 自分より成績の良い学生の希望順位に応じて、自分の希望順位を変更
- 読みを間違えると，望ましくない結果になる
- 例：学生1の希望はA研究室，しかし，自分の成績だと第一希望で通る自信がもてず，B研究室を第一希望にした．しかし，蓋を開けてみると難関のA研究室は多くの学生が敬遠し，自分より成績が低い学生2が配属された．
  - 学生1にとっても，A研究室にとっても望ましくない

29

卒研配属手続き (2011年より)

- Gale & Shapley アルゴリズムに準じた方式に変更
- 横尾研究室が方式設計，配属プログラムの開発を担当
  - 学生が安定結婚問題における女性，研究室が男性に対応
  - 1対1ではなく，多対1のマッチング
  - 研究室側の好みはすべて共通（成績順）
  - 成績順位は共通・課程必修科目の成績によって決定

30

卒研配属手続きの概要（I）

第一ステップ：各学生を、それぞれ希望順位1位の研究室に割り振る

i. 定員枠を超えなかった場合は、希望学生をその研究室に仮にマッチさせる（あくまでこの段階では仮であることに注意）。

ii. 定員枠を上回った場合は、希望学生を成績の高い順から定員枠まで仮にマッチさせる。

31

卒研配属手続きの概要 (II)

第二ステップ：第一ステップでもれた学生を、それぞれの第二希望の研究室に割り振る。
個々の研究室において、第二ステップで移動した学生と、既に仮にマッチしている学生を合わせて、第一ステップと同様にマッチングを行う。すなわち、定員枠以内であれば全員を仮にマッチとし、定員枠を超えた場合は、成績順に定員枠まで仮にマッチとする。

第二ステップでもれた学生を、それぞれの第三希望の研究室に割り振る。以下、もれた学生が一人もいなくなるまで同様の手順を続ける

最終的に、それぞれの研究室と仮にマッチしていた学生を正式配属とする。

32

卒研配属手続きの概要 (III)

定員枠は2, 学生を表す数字は順位
1, 2, 3:A, B, Cの順
4, 5:　B, C, Aの順
6:　　C, B, Aの順

第 1 ステップ
A 教授: 1, 2, 3
B 教授: 4, 5
C 教授: 6
赤字は仮マッチ

第 2 ステップ
A 教授: 1, 2
B 教授: 3, 4, 5
C 教授: 6

第 3 ステップ
A 教授: 1, 2
B 教授: 3, 4
C 教授: 5, 6
配属決定！

各ステップで、前回の仮マッチの結果は一切考慮していないことに注意。前のステップで定員枠からもれても不利にならない。

33

卒研配属手続きの性質

- 正直に自分の希望を提出するのが最適！
  - 希望順位をいじっても一切得をすることがない。
  - 「本当はこの研究室に行きたいが、自分の成績ではちょっと難しいかも...」という場合でも，希望して後で不利になることはない。
  - なるべく多くの希望を記述した方がよい（1次希望調査で決定しないと不利になる）。
- 「理不尽な結果」が絶対に生じない。
  - 「理不尽な結果」とは、自分が配属された研究室よりも、自分の希望順位が高い研究室に、自分より成績が下位の学生が配属されていること。
  - 理不尽な結果が生じない範囲で、自分にとって最も希望の高い研究室に配属されることが保証される。
- 学生全員にとって最高の結果が達成できる！

34

卒研配属手続きの詳細

- 2010年度までは各研究室の定員は教授3，准教授1で固定
- 2011年度より，教授は最大3, 最小2, 准教授は最大2, 最小1の範囲で、学生諸君の希望を重視して柔軟に定員枠を決定するように変更
- Gale & Shapleyアルゴリズムは最小配属人数を保証しない
- 研修医配属等においても同様の課題が生じる（僻地や離島に希望者がゼロだと困る）
- 正直が最良の策であることを保証しながら最小配属人数を保証する新しいアルゴリズムを横尾研で開発
- 研究室側の選好を入れることも可能

35

例題：宣教師と人喰い人種

- 宣教師が3人，人喰い人種が3人いる
- ボートで対岸に渡りたい
- ボートは一つだけ，一度に二人しか乗れない
- 人喰い人種の数が宣教師の数を超えると，宣教師は食べられてしまう！
- 無事に対岸に全員が渡れるだろうか?

36

問題の抽象的な表現

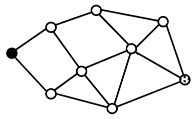
- 状態の集合
  - 例えば，全員こちら側にいる，全員向こう岸に渡った等が一つの状態
- 各状態に対して，近傍の状態が定義される（ボートで移動することによって遷移可能）
- 初期状態，目標状態

37

問題の抽象的な表現 (続き)

以下のように考えることにより，この問題はノード（節点）とリンク（辺）で構成されるグラフで表現することができる

- 各状態をノード
- 近傍の状態を結ぶリンク
- 目的は初期ノードからゴールノードに至る経路を見つけること



38

経路探索問題の応用事例

- カーナビの経路探索
- VLSIのレイアウト
- 移動ロボットの経路プランニング
- …

39

可能な状態の表現

- ボートの中で喰われることはない
- 川のこちら側の状態を決めれば，向こう岸の状態は一通りに決まる
- よって，こちら側の状態のみを表現すれば十分
- (m, c, b) の三つ組で表現可能
  - m: 宣教師の数 (0から3)
  - c: 人喰い人種の数 (0から3)
  - b: ボートがあれば1, なければ0
- 初期状態は (3, 3, 1), 目標状態は (0, 0, 0)

40

可能な状態

- 可能な状態数は 4×4×2=32通り
- 喰われてしまう状態を除く
  - m≠0なら，m≧cでないといけない
  - m ≠3なら，対岸で喰われないためには，3 – m ≧3 – c
- 初期状態から到達できない状態を除く
  - (0, 0, 1) --- どこからボートが来たの?
  - (3, 3, 0) --- ボートはどこに行った?
  - (0, 3, 0)，(3, 0, 1) --- 直前にボートに乗ったのは誰?

41

可能な状態 (続き)

- 結局，考慮すべき状態は16通りに絞られる
- (0, 0, 0)
- (0, 1, 0), (0, 1, 1), (0, 2, 0), (0, 2, 1), (0, 3, 1)
- (1, 1, 0), (1, 1, 1)
- (2, 2, 0), (2, 2, 1)
- (3, 0, 0), (3, 1, 0), (3, 1, 1), (3, 2, 0), (3, 2, 1)
- (3, 3, 1)

42

近傍関係

- ボートが出る／着くによって近傍が定義される
- (m, c, 0)なら，(m+1, c, 1), (m+2, c, 1), (m, c+1, 1), (m, c+2, 1), (m+1, c+1, 1)が近傍
- (m, c, 1)なら，(m-1, c, 0), (m-2, c, 0), (m, c-1, 0), (m, c-2, 0), (m-1, c-1, 0)が近傍
- 許されない状態，マイナスになったり，4以上になるものを除く

43

演習

- 状態のグラフを書いてみよう
- 最短経路を見つけよう
- (0, 0, 0)
- (0, 1, 0), (0, 1, 1), (0, 2, 0), (0, 2, 1), (0, 3, 1)
- (1, 1, 0), (1, 1, 1)
- (2, 2, 0), (2, 2, 1)
- (3, 0, 0), (3, 1, 0), (3, 1, 1), (3, 2, 0), (3, 2, 1)
- (3, 3, 1)

44

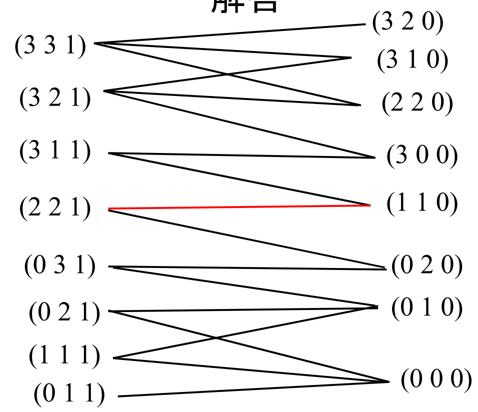
状態

(3 3 1)
(3 2 1)
(3 1 1)
(2 2 1)
(0 3 1)
(0 2 1)
(1 1 1)
(0 1 1)

(3 2 0)
(3 1 0)
(2 2 0)
(3 0 0)
(1 1 0)
(0 2 0)
(0 1 0)
(0 0 0)

45

解答



46