ТЕОРИЯ И ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ

АЛГОРИТМЫ И СТРУКТУРЫ ДАННЫХ

Лекция 2. Вероятностные алгоритмы

Глухих Михаил Игоревич, к.т.н., доц.

mailto: glukhikh@mail.ru

Задача о найме

- Менеджер хочет нанять нового работника
- Агентство присылает ему по одному кандидату в день
- Он проводит с кандидатом собеседование
- И принимает окончательное решение, брать его на работу или нет (до опроса остальных кандидатов)
- Каждое собеседование стоит X
- Приём на работу нового работника дополнительно стоит Y
- В итоге мы хотим нанять самого лучшего кандидата
- Если за N дней мы опросили N кандидатов и наняли из них M, мы потратили NX + MY
- Наши затраты в худшем / лучшем / **среднем** случае?

Решение

- Мы принимаем кандидата на работу, если он лучше имеющегося работника (считаем, что на собеседовании мы это можем установить)
- Худший случай?
- Лучший случай?
- Средний случай?

Решение

- Мы принимаем кандидата на работу, если он лучше имеющегося работника (считаем, что на собеседовании мы это можем установить)
- Худший случай: C = NX + NY
  (по возрастанию квалификации)
- Лучший случай: C = NX + Y
  (по убыванию квалификации)
- Средний случай?

Решение

- Мы принимаем кандидата на работу, если он лучше имеющегося работника (считаем, что на собеседовании мы это можем установить)
- Худший случай: C = NX + NY
  (по возрастанию квалификации)
- Лучший случай: C = NX + Y
  (по убыванию квалификации)
- Средний случай: имеем случайную перестановку списка квалификаций (1, 2, …, N-1, N)
  - Индикаторная случайная величина Zi = 1, если i-й кандидат нанят, или =0, если он не нанят
  - Среднее E[Zi] = 1/i (1-й нанимается всегда, второй, если он лучше 1-го, 3-й, если он лучше 1-го и 2-го и так далее)
  - Сумма E[Zi] = Сумма 1/i = O(ln(N))

Задача о гардеробщике

- Гардеробщик принимает шляпы у N посетителей и затем возвращает их случайным образом
- Сколько посетителей в среднем получит обратно собственную шляпу?

Задача о гардеробщике

- Гардеробщик принимает шляпы у N посетителей и затем возвращает их случайным образом
- Сколько посетителей в среднем получит обратно собственную шляпу?
- РЕШЕНИЕ
  - Индикаторная случайная величина Zi = 1, если посетитель i получил свою шляпу, или 0 в противном случае
  - Все Zi имеют среднее E[Zi] = 1/N
  - Отсюда их сумма имеет среднее 1

Задача случайной перестановки

- Дано: List<T> / Array<T> inputList размера N с уникальными элементами, то есть inputList[i] != inputList[j]
- Требуется: сформировать outputList того же размера, в котором элементы inputList переставлены так, чтобы...

Задача случайной перестановки

- Дано: List<T> / Array<T> inputList размера N с уникальными элементами, то есть inputList[i] != inputList[j]
- Требуется: сформировать outputList того же размера, в котором элементы inputList переставлены так, чтобы…
  - outputList[i] == inputList[j] с вероятностью 1 / N при любых i и j ?

Задача случайной перестановки

- Дано: List<T> / Array<T> inputList размера N с уникальными элементами, то есть inputList[i] != inputList[j]
- Требуется: сформировать outputList того же размера, в котором элементы inputList переставлены так, чтобы…
  - outputList[i] == inputList[j] с вероятностью 1 / N при любых i и j ?
  - НЕДОСТАТОЧНО
  - ОПРОВЕРЖЕНИЕ: выбираем случайное число от 0 до N-1 и сдвигаем inputList по кольцу (вправо или влево) на данное количество элементов

Задача случайной перестановки

- Дано: List<T> / Array<T> inputList размера N с уникальными элементами, то есть inputList[i] != inputList[j]
- Требуется: сформировать outputList того же размера, в котором элементы inputList переставлены так, чтобы…
  - Все N! возможных перестановок были равновероятны!

Очевидное решение

- Дано: List<T> / Array<T> inputList размера N с уникальными элементами, то есть inputList[i] != inputList[j]
- Требуется: сформировать outputList того же размера, в котором элементы inputList переставлены так, чтобы…
  - Все N! возможных перестановок были равновероятны!
- Решение: много раз выбираем случайную пару (k, m), 0<=k<N, 0<=m<N, и переставляем элементы k и m местами
- Что можно сказать о его характеристиках?

Очевидное решение

- Решение: много раз выбираем случайную пару (k, m), 0<=k<N, 0<=m<N, и переставляем элементы k и m местами
- Что можно сказать о его характеристиках?
- Список из двух элементов: ОДИН раз либо переставили (50%), либо нет – то, что надо
- Список из трёх элементов – ???
  - 1 2 3 (3/9) (21/81) (135/729)
  - 2 1 3 (2/9) (12/81) (126/729)
  - 3 2 1 (2/9) (12/81) (126/729)
  - 1 3 2 (2/9) (12/81) (126/729)
  - 3 1 2 (0) (12/81) (108/729)
  - 2 3 1 (0) (12/81) (108/729)

Очевидное решение

- Решение: много раз выбираем случайную пару (k, m), 0<=k<N, 0<=m<N, и переставляем элементы k и m местами
- Что можно сказать о его характеристиках?
- Уже для трёх элементов мы НИКОГДА не получим равновероятных перестановок (по той простой причине, что 9^n не делится на 6 нацело)

Sort-based перестановка

- Дано: List<T> / Array<T> inputList размера N с уникальными элементами, то есть inputList[i] != inputList[j]
- Формируем priorityList размера N, каждый элемент которого – случайное число в диапазоне от 1 до N^3
- Сортируем inputList, используя priorityList как ключи
- Трудоёмкость: как у сортировки

Перестановка “на месте”

- Дано: List<T> / Array<T> inputList размера N с уникальными элементами, то есть inputList[i] != inputList[j]
- В цикле i = от 0 до N-1 меняем местами элементы #i и #(i+rand(n-i)) – иначе говоря, выбираем случайный элемент между #i и #n и меняем его местами с элементом #i
- Трудоёмкость: O(N)
- Доказательство корректности: см. [Кормен], глава 5

Итоги

- Рассмотрели
  - Вероятностный анализ поведения алгоритмов, индикаторные случайные величины
  - Алгоритмы случайной перестановки
- Далее
  - Обзор алгоритмов сортировки