Математика и информационные технологии

С. М. Дудаков

Факультет прикладной математики и кибернетики

2021-10-31

С чего начиналась вычислительная техника?

Задачи обработки статистических данных (перепись населения)

Задачи расшифровки сообщений (Вторая мировая война)

Расчёты математических моделей (ядерный взрыв)

Современное применение

Хранение и обработка самой разной информации

- Географическая
- Экономическая
- Текстовая
- Мультимедиа (изображение, звук)

Информационные технологии

- Передача данных
- Хранение данных
- Обработка данных

Математические модели и методы применяются при решении всех этих задач

Разрешимые и неразрешимые задачи

Алгоритмическая разрешимость — принципиальная возможность построения алгоритма для решения задачи

- Существуют задачи, для решения которых не существует алгоритмов

Разрешимые и неразрешимые задачи

Алгоритмическая разрешимость — принципиальная возможность построения алгоритма для решения задачи

- Существуют задачи, для решения которых не существует алгоритмов
- Бóльшая часть задач неразрешима

Алгоритмические задачи

Проблема остановки

Определить, останавливается ли программа при заданных входных данных (не зацикливается)

Алгоритмические задачи

Проблема остановки

Определить, останавливается ли программа при заданных входных данных (не зацикливается)

Проблема тотальности

Определить, останавливается ли программа при любых входных данных (никогда не зацикливается)

Алгоритмические задачи

Проблема остановки

Определить, останавливается ли программа при заданных входных данных (не зацикливается)

Проблема тотальности

Определить, останавливается ли программа при любых входных данных (никогда не зацикливается)

Первая задача алгоритмически неразрешима, вторая — на один шаг «неразрешимее»

Замены в словах

Есть несколько вариантов замен, например,
$aa \mapsto bab$, $ba \mapsto aab$, $aba \mapsto bb$

Можно ли с помощью них из одного слова получить другое?

$aaa \mapsto baba \mapsto aabba \mapsto aabaab \mapsto abbab \mapsto abaabb \mapsto bbabb \mapsto baabbb \mapsto bbabbbb$

Числовые задачи

Диофантовы уравнения

Определить имеет ли произвольное алгебраическое уравнение с несколькими неизвестными целые решения

Числовые задачи

Диофантовы уравнения

Определить имеет ли произвольное алгебраическое уравнение с несколькими неизвестными целые решения

Элементарная арифметика

- По утверждению о натуральных числах, определить, верно оно или нет

Числовые задачи

Диофантовы уравнения

Определить имеет ли произвольное алгебраическое уравнение с несколькими неизвестными целые решения

Элементарная арифметика

- По утверждению о натуральных числах, определить, верно оно или нет
- Задача «неразрешимее» всех предыдущих на бесконечное число шагов

Формулировка

Пусть программа $\mathfrak{A}$ (транслятор) преобразует каждую программу $\mathfrak{B}$ в некоторую программу $\mathfrak{C}$.

Тогда существует такая программа $\mathfrak{B}_0$, что полученная в результате преобразования программа $\mathfrak{C}_0$ работает точно так же, как $\mathfrak{B}_0$.

Самовоспроизводящиеся программы

Можно ли написать программу, которая при запуске напечатает свой исходный текст?

Самовоспроизводящиеся программы

Можно ли написать программу, которая при запуске напечатает свой исходный текст?

- Пусть программа $\mathfrak{A}$ каждую программу $\mathfrak{B}$ преобразует в программу
  `print "`$\mathfrak{B}$`"`
- Согласно теореме, существует программа $\mathfrak{B}_0$, такая, что $\mathfrak{B}_0$ работает так же, как
  `print "`$\mathfrak{B}_0$`"`
- Последняя программа печатает $\mathfrak{B}_0$.
- Значит $\mathfrak{B}_0$ печатает $\mathfrak{B}_0$.

Автоматическая генерация программ

Пусть программа $\mathfrak{G}$ автоматически генерирует новые программы по каким-то данным.

Автоматическая генерация программ

Пусть программа $\mathfrak{G}$ автоматически генерирует новые программы по каким-то данным.

Насколько оптимальными являются сгенерированные программы? Нельзя ли их оптимизировать?

Автоматическая генерация программ

Пусть программа $\mathfrak{G}$ автоматически генерирует новые программы по каким-то данным.

Насколько оптимальными являются сгенерированные программы? Нельзя ли их оптимизировать?

Неоптимальность генерации

Всякий автоматический генератор программ $\mathfrak{G}$ неоптимален

Неоптимальность генерации

Рассмотрим программу $\mathfrak{A}$:

- Вход — программа $\mathfrak{B}$
- Генерировать с помощью $\mathfrak{G}$ новые программы, пока не получится программа $\mathfrak{C}$ длиннее $\mathfrak{B}$
- Выдать $\mathfrak{C}$

Неоптимальность генерации

Рассмотрим программу $\mathfrak{A}$:

- Вход — программа $\mathfrak{B}$
- Генерировать с помощью $\mathfrak{G}$ новые программы, пока не получится программа $\mathfrak{C}$ длиннее $\mathfrak{B}$
- Выдать $\mathfrak{C}$

По теореме о неподвижной точке существует программа $\mathfrak{B}_0$, которая будет эквивалентна $\mathfrak{C}_0$, выданной программой $\mathfrak{A}$, то есть $\mathfrak{G}$

Неоптимальность генерации

Рассмотрим программу $\mathfrak{A}$:

- Вход — программа $\mathfrak{B}$
- Генерировать с помощью $\mathfrak{G}$ новые программы, пока не получится программа $\mathfrak{C}$ длиннее $\mathfrak{B}$
- Выдать $\mathfrak{C}$

По теореме о неподвижной точке существует программа $\mathfrak{B}_0$, которая будет эквивалентна $\mathfrak{C}_0$, выданной программой $\mathfrak{A}$, то есть $\mathfrak{G}$

Но $\mathfrak{B}_0$ короче сгенерированной программы $\mathfrak{C}_0$

Формулировка

Примеры

zip/unzip, gzip/gunzip и т. д.

Формулировка

Примеры

zip/unzip, gzip/gunzip и т. д.

- Есть данные D
- Есть программа $\mathfrak{A}$ (архиватор), которая должна по D построить сжатое представление S
- Есть программа $\mathfrak{B}$ (разархиватор), которая должна по S восстановить D

Математическая формулировка

- Сжатое представление S — программа

Математическая формулировка

- Сжатое представление S — программа
- Разархиватор $\mathfrak{B}$ — это интерпретатор, который выполняет программу S

Математическая формулировка

- Сжатое представление S — программа
- Разархиватор $\mathfrak{B}$ — это интерпретатор, который выполняет программу S
- D — результат выполнения программы S интерпретатором $\mathfrak{B}$

Пример

- Сжимаемая информация — число
- Сжатое представление — арифметическое выражение

Пример

- Сжимаемая информация — число
- Сжатое представление — арифметическое выражение
- Разархиватор — программа, вычисляющая значение выражения

Пример

- Сжимаемая информация — число
- Сжатое представление — арифметическое выражение
- Разархиватор — программа, вычисляющая значение выражения
- Пусть $D = 999999999999999999999999999999$
- В качестве сжатого представления D можно взять $S = 10^{30} - 1$

Результат

Что делает архиватор?

Архиватор $\mathfrak{A}$ по D генерирует программу S

Результат

Что делает архиватор?

Архиватор $\mathfrak{A}$ по D генерирует программу S

Сгенерированные программы неоптимальны

Результат

Что делает архиватор?

Архиватор $\mathfrak{A}$ по D генерирует программу S

Сгенерированные программы неоптимальны

Все архиваторы сжимают «плохо»

Оптимальное сжатие

Существует оптимальный способ сжатия информации

Существует разархиватор для этого оптимального способа

Оптимальное сжатие

Существует оптимальный способ сжатия информации

Существует разархиватор для этого оптимального способа

Оптимальный архиватор построить невозможно

Что можно сжать?

Некоторые файлы сжимаются «хорошо», другие — «плохо»

Что можно сжать?

Некоторые файлы сжимаются «хорошо», другие — «плохо»

Бóльшую часть данных сжать нельзя

Что можно сжать?

Некоторые файлы сжимаются «хорошо», другие — «плохо»

Бóльшую часть данных сжать нельзя

Если есть бесконечный поток данных $\mathfrak{D}$, то существует бесконечно много точек, в которых данный поток можно немного сжать

Общее понятие

Примеры мер сложности

- Время работы
- Требуемая память

Общее понятие

Примеры мер сложности

- Время работы
- Требуемая память

Сложность — величина, характеризующая количество ресурсов необходимое для вычислительного процесса (существует точное определение)

Сложнорешаемые задачи

Для любой вычислимой функции A существуют задачи, сложность решения которых превосходит A.

Сложнорешаемые задачи

Для любой вычислимой функции A существуют задачи, сложность решения которых превосходит A.

Пример

Если N — количество данных на входе и $A = 2^N$, то существуют задачи, требующие не менее 2^N времени для своего решения (или не менее 2^N ячеек памяти).

Теорема об ускорении

Для любой вычислимой функции A есть задачи такие, что если программа $\mathfrak{A}$ решает эту задачу, то всегда существует программа $\mathfrak{B}$, которая решает ту же самую задачу быстрее как минимум на A^{-1}

Теорема об ускорении

Для любой вычислимой функции A есть задачи такие, что если программа $\mathfrak{A}$ решает эту задачу, то всегда существует программа $\mathfrak{B}$, которая решает ту же самую задачу быстрее как минимум на A^{-1}

Пример

Пусть $A = \log_2 N$. Существует задача, и для всякой программы $\mathfrak{A}$, которая её решает за время T, найдётся программа $\mathfrak{B}$, решающая ту же задачу за время $\log_2 T$

Точно не установленная сложность

Существуют практически важные задачи, сложность которых до сих пор не известна (класс NP)

Точно не установленная сложность

Существуют практически важные задачи, сложность которых до сих пор не известна (класс NP)

Нет доказательства, что они требуют времени не менее чем 2^N (N — размер исходных данных для решения задачи)

Точно не установленная сложность

Существуют практически важные задачи, сложность которых до сих пор не известна (класс NP)

Нет доказательства, что они требуют времени не менее чем 2^N (N — размер исходных данных для решения задачи)

Не найдено алгоритмов, которые требуют меньшего времени

Примеры задач из NP

Разбиение чисел

Даны числа $X_1, \ldots, X_N$. определить, можно ли их разделить на две части, суммы чисел в которых равны

Примеры задач из NP

Разбиение чисел

Даны числа $X_1, \ldots, X_N$. определить, можно ли их разделить на две части, суммы чисел в которых равны

Задача об упаковке

Даны N вещей объёмами $X_1, \ldots, X_N$. Даны M контейнеров объёмами $Y_1, \ldots, Y_M$. Определить, можно ли разложить все вещи по контейнерам

Примеры задач из NP

Составление расписания работ

Нужно выполнить работы $R_1, \ldots, R_N$. Для каждой работы R_i известна её длительность T_i и какие работы должны быть выполнены до неё. определить, можно ли выполнить все работы за время T.

Примеры задач из NP

Составление расписания работ

Нужно выполнить работы $R_1, \ldots, R_N$. Для каждой работы R_i известна её длительность T_i и какие работы должны быть выполнены до неё. определить, можно ли выполнить все работы за время T.

Задача коммивояжёра

Нужно посетить несколько пунктов. Известно время пути между любыми двумя из них. Определить, можно ли объехать все пункты за заданное время T

Взаимосвязь

Сложность многих задач взаимосвязана

Взаимосвязь

Сложность многих задач взаимосвязана

Пример

Четыре предыдущих задачи имеют одну и ту же сложность

Взаимосвязь

Сложность многих задач взаимосвязана

Пример

Четыре предыдущих задачи имеют одну и ту же сложность

Если для одной из них мы найдём быстрый алгоритм, то и другие сможем быстро решить

Взаимосвязь

Сложность многих задач взаимосвязана

Пример

Четыре предыдущих задачи имеют одну и ту же сложность

Если для одной из них мы найдём быстрый алгоритм, то и другие сможем быстро решить

Если для одной из них докажем, что её нельзя быстро решить, то то же самое касается и остальных

Основные принципы

Кодирование информации с целью затруднить её восстановление без соответствующей санкции

Основные принципы

Кодирование информации с целью затруднить её восстановление без соответствующей санкции

Задача расшифровки должна иметь большую вычислительную сложность (требовать много времени, памяти и т. д.)

Основные принципы

Кодирование информации с целью затруднить её восстановление без соответствующей санкции

Задача расшифровки должна иметь большую вычислительную сложность (требовать много времени, памяти и т. д.)

Задачи шифровки и расшифровки должны быть независимы

Пример шифрования

Замена символов в строке.

$A \to F \to L \to B \to \cdots \to G \to A$

Пример шифрования

Замена символов в строке.

$A \to F \to L \to B \to \cdots \to G \to A$

Недостатки

- Легко расшифровать
- Каждый, кто может зашифровать сообщение, может и расшифровать

Современная основа

Сложнорешаемые задачи

- Логарифмирование в конечных кольцах: решение уравнения $A^x \bmod B = C$.
- Разложение составного числа на простые множители

Алгоритм RSA

- P, Q — больши́е простые числа, $X = PQ$, $ZY \bmod (P-1)(Q-1) = 1$
- Ключ для шифрования— два числа X и Y
- Ключ для расшифровки — два числа X и Z

Алгоритм RSA

- P, Q — больши́е простые числа, $X = PQ$, $ZY \bmod (P-1)(Q-1) = 1$
- Ключ для шифрования— два числа X и Y
- Ключ для расшифровки — два числа X и Z
- Шифрование — $S = D^Y \bmod X$
- Расшифровка — $D = S^Z \bmod X$

Алгоритм RSA

- P, Q — больши́е простые числа, $X = PQ$, $ZY \bmod (P-1)(Q-1) = 1$
- Ключ для шифрования— два числа X и Y
- Ключ для расшифровки — два числа X и Z
- Шифрование — $S = D^Y \bmod X$
- Расшифровка — $D = S^Z \bmod X$
- Для посторонних лиц — нужно разложить число X на простые множители

Шифрованый канал связи

- Нужно установить безопасное соединение между A и B
- Связь между A и B возможна только по открытому каналу
- Как предотвратить перехват информации третьими лицами?

Схема Эль-Гамаля

- A выбирает P, Q — бо́льшие числа, P — простое, $Q < P$, и отправляет их B

Схема Эль-Гамаля

- A выбирает P, Q — бо́льшие числа, P — простое, $Q < P$, и отправляет их B
- A берёт случайное число X и отправляет B число $X' = Q^X \bmod P$
- B берёт случайное число Y и отправляет A число $Y' = Q^Y \bmod P$
- Ключ шифрования — $K = Q^{XY} \bmod P$

Схема Эль-Гамаля

- A выбирает P, Q — бо́льшие числа, P — простое, $Q < P$, и отправляет их B
- A берёт случайное число X и отправляет B число $X' = Q^X \bmod P$
- B берёт случайное число Y и отправляет A число $Y' = Q^Y \bmod P$
- Ключ шифрования — $K = Q^{XY} \bmod P$
- A и B легко вычисляют ключ: $K = X'^Y \bmod P = Y'^X \bmod P$

Схема Эль-Гамаля

- A выбирает P, Q — больши́е числа, P — простое, $Q < P$, и отправляет их B
- A берёт случайное число X и отправляет B число $X' = Q^X \bmod P$
- B берёт случайное число Y и отправляет A число $Y' = Q^Y \bmod P$
- Ключ шифрования — $K = Q^{XY} \bmod P$
- A и B легко вычисляют ключ: $K = X'^Y \bmod P = Y'^X \bmod P$
- Для посторонних лиц нужно решить одно из уравнений: $X' = Q^X \bmod P$ или $Y' = Q^Y \bmod P$

Основные идеи

Что делать, если «быстрый» алгоритм точного решения задачи неизвестен

Основные идеи

Что делать, если «быстрый» алгоритм точного решения задачи неизвестен

Использовать алгоритмы с использованием случайных чисел, которые дают правильный результат со сколь угодно большой степенью достоверности

Пример

Проблема нуля

Дано арифметическое выражение с переменными $X_1, \ldots, X_N$. Определить, всегда ли это выражение равно 0.

Пример

Проблема нуля

Дано арифметическое выражение с переменными $X_1, \ldots, X_N$. Определить, всегда ли это выражение равно 0.

Традиционный алгоритм

- Раскрыть все скобки, привести подобные члены, если хотя бы один из них не сократился, то выражение не всегда равно 0.

Пример

Проблема нуля

Дано арифметическое выражение с переменными $X_1, \ldots, X_N$. Определить, всегда ли это выражение равно 0.

Традиционный алгоритм

- Раскрыть все скобки, привести подобные члены, если хотя бы один из них не сократился, то выражение не всегда равно 0.
- Способ требует большого количества времени и памяти (если N — длина исходного выражения, то может потребоваться 2^N ячеек памяти)

Проблема нуля

Вероятностный способ

Случайным образом сгенерировать значения для переменных $X_1, \ldots, X_N$. Вычислить значение выражения. Повторить несколько раз. Если всегда получается 0, то ответ — «да».

Тест на простоту

Определить, является ли число X простым?

Тест на простоту

Определить, является ли число X простым?

Традиционный способ

- Перебрать все числа $Y = 2, \ldots, X - 1$
- Проверить делится ли X на Y

Тест на простоту

Определить, является ли число X простым?

Традиционный способ

- Перебрать все числа $Y = 2, \ldots, X - 1$
- Проверить делится ли X на Y

Если число X большое (4096 бит, около 1200 десятичных цифр), то время будет неприемлемо большим $\approx 10^{1200}$

Тест Миллера-Рабина

Малая теорема Ферма

Для простого числа P выполнено $A^{P-1} \bmod P = 1$ для всех $A < P$

Тест Миллера-Рабина

Малая теорема Ферма

Для простого числа P выполнено $A^{P-1} \bmod P = 1$ для всех $A < P$

Следствие

- $\sqrt{A^{P-1} \bmod P} = A^{\frac{P-1}{2}} \bmod P = \pm 1$

Тест Миллера-Рабина

Малая теорема Ферма

Для простого числа P выполнено $A^{P-1} \bmod P = 1$ для всех $A < P$

Следствие

- $\sqrt{A^{P-1} \bmod P} = A^{\frac{P-1}{2}} \bmod P = \pm 1$
- Если $A^{\frac{P-1}{2}} \bmod P = 1$, то $A^{\frac{P-1}{4}} \bmod P = \pm 1$

Тест Миллера-Рабина

Малая теорема Ферма

Для простого числа P выполнено $A^{P-1} \bmod P = 1$ для всех $A < P$

Следствие

- $\sqrt{A^{P-1} \bmod P} = A^{\frac{P-1}{2}} \bmod P = \pm 1$
- Если $A^{\frac{P-1}{2}} \bmod P = 1$, то $A^{\frac{P-1}{4}} \bmod P = \pm 1$
- И т. д.

Тест Миллера-Рабина

Малая теорема Ферма

Для простого числа P выполнено $A^{P-1} \bmod P = 1$ для всех $A < P$

Следствие

- $\sqrt{A^{P-1} \bmod P} = A^{\frac{P-1}{2}} \bmod P = \pm 1$
- Если $A^{\frac{P-1}{2}} \bmod P = 1$, то $A^{\frac{P-1}{4}} \bmod P = \pm 1$
- И т. д.

Теорема Миллера-Рабина

Если P составное, то таких чисел A «мало» (менее четверти от всех)

Алгоритм Миллера-Рабина

Дано число X

Алгоритм Миллера-Рабина

Дано число X

Для случайных чисел $A_1, \ldots, A_k$ выполнить

- $Y = X$
- пока Y чётно выполнять
- если $A_i^Y \bmod X = -1$, вернуть «да»
- $Y = Y/2$; конец цикла
- если $A_i^Y \bmod X = \pm 1$, вернуть «да», иначе «нет»

Алгоритм Миллера-Рабина

Дано число X

Для случайных чисел $A_1, \ldots, A_k$ выполнить

- $Y = X$
- пока Y чётно выполнять
- если $A_i^Y \bmod X = -1$, вернуть «да»
- $Y = Y/2$; конец цикла
- если $A_i^Y \bmod X = \pm 1$, вернуть «да», иначе «нет»

Если хотя бы раз ответ «нет», то число составное, иначе простое

Определение

База данных

- Конечный набор конечных таблиц
- Служит для хранения информации

Пример базы данных

Расписание автобусов

Отпр	Назн	Вр отпр	Путь
Тверь	Москва	10:00	02:00
Москва	Новгород	11:30	08:00
Тверь	Торжок	14:15	01:00
Москва	Тверь	13:10	02:00
Москва	Кострома	11:00	15:00

Язык запросов

- Служит для извлечения информации из базы данных
- Классический язык — SQL

Язык запросов

- Служит для извлечения информации из базы данных
- Классический язык — SQL

Пример запроса

- Куда можно проехать с одной пересадкой?
- Запрос на SQL:

```
select p1.отпр, p2.назн
from расп p1, расп p2
where p1.назн = p2.отпр
```

Более сложный запрос

Вопрос

Куда можно проехать с несколькими пересадками?

Более сложный запрос

Вопрос

Куда можно проехать с несколькими пересадками?

На «классическом» SQL (до версии SQL'99) этот запрос записать нельзя

Более сложный запрос

Вопрос

Куда можно проехать с несколькими пересадками?

На «классическом» SQL (до версии SQL'99) этот запрос записать нельзя

В версии SQL'99 произошло расширение языка за счёт рекурсивных запросов

Насколько универсален SQL'99

Вопрос

Не может ли случиться так, что снова окажется: какие-то запросы невозможно записать?

Насколько универсален SQL'99

Вопрос

Не может ли случиться так, что снова окажется: какие-то запросы невозможно записать?

Ответ

Не может: на языке SQL'99 можно записать любые запросы, которые можно описать хотя бы на одном языке программирования

Конец

Спасибо за внимание!

Вопросы, пожалуйста...