Дедлайн: 28.11.2023 22:00

Инструкция

Задания в рамках домашней работы подразделяются на два блока:

1. Блок А «*Аналитические задачи*» – решения задач А1-А3 оформляются в письменном в виде в любом удобном формате (TeX, скан, фото и др.).
2. Блок Р «*Задачи на разработку*» – решения задач Р1-Р3 загружаются в систему Яндекс.Контест и проходят автоматизированное тестирование.

Блок А			**Блок Р**			**Итого**
А1	**А2**	**А3***	**Р1**	**Р2**	**Р3**	
15	30	20	5	5	6	61 (81*)

Задачи, помеченные *, не являются обязательными для решения (относятся к *бонусным*). Подтверждение представленных решений бонусных заданий обязательно сопровождается *устной защитой.*

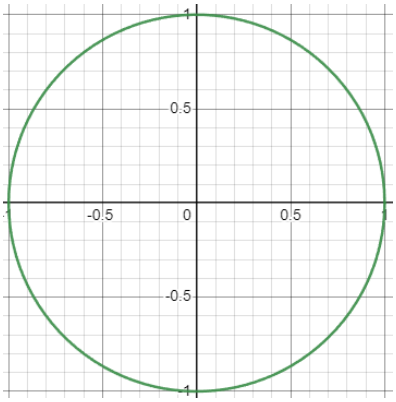
Плагиат влечет за собой обнуление результатов у всех вовлеченных лиц.

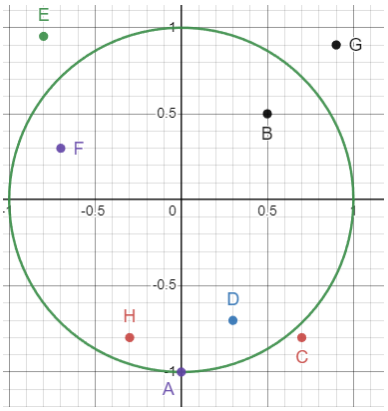
Удачи!

Блок А. Анализ результатов стохастического алгоритма типа «Монте-Карло». Реализация и сравнительный анализ гибридных сортировок

Задание А1 (15 баллов) Приблизительное вычисление числа π

В этой задаче требуется провести ряд экспериментов по вычислению приблизительного значения числа π с использованием стратегии стохастического алгоритма типа «Монте-Карло» на основе многократной генерации случайных величин. Рассмотрим круг единичного радиуса с центром в начале координат $(0,0)$. Выполним случайную генерацию некоторого количества точек N, координаты которых по x и y попадают в интервал $[-1,1]$ (см. на рисунке ниже).





Нам известно, что точная площадь круга с радиусом r составляет πr^2, а площадь квадрата (с тем же центром), в который этот круг вписан составляет $2r \cdot 2r = 4r^2$. Составим отношение площадей: $\frac{\pi r^2}{4r^2} = \frac{\pi}{4}$.

В рамках нашего эксперимента: точная площадь круга заменяется на M – количество попадающих в круг точек из общего числа случайно сгенерированных, а площадь квадрата заменяется на N. Составим отношение таких оценок площадей: $\frac{M}{N} = \frac{\pi}{4}$. Тогда, приблизительное значение числа π рассчитывается следующим образом: $\pi = \frac{4M}{N}$.

1. (6 баллов) Разработайте программу для вычисления приблизительного значения числа π по алгоритму, представленному выше. Оформите случайную генерацию координат точек в виде отдельной функции.
2. (7 баллов) Проведите ряд экспериментов для оценки точности вычисления значения числа π в зависимости от разного количества случайно сгенерированных точек N, которое меняется от 100 до 5000 с шагом 100. Представьте результаты проведенных экспериментов в следующем виде:
    a. График 1, который отображает, как изменяется приближенное значение числа π, полученное в разработанной программе, в зависимости от общего числа точек N.
    b. График 2, который отображает, как изменяется относительное отклонение (в %) приближенного значения числа π от точного в зависимости от общего числа точек N.
    c. Исходные данные, использованные для построения графиков.

    Для удобства представления, графики можно разделить на несколько.
3. (2 балла) Опишите полученные вами результаты и представьте выводы.

Задание A2 (30 баллов) Гибридная сортировка MERGE+INSERTION SORT

Известно, что MERGE SORT является оптимальным алгоритмом сортировки на основе сравнений и имеет сложности $O(n \log n)$, а INSERTION SORT имеет сложность $O(n^2)$.

Однако на небольших размерах массивов INSERTION SORT выполняет сортировку быстрее, чем MERGE SORT за счет лучшей оценки константы, а также снижения накладных расходов, которые связаны с рекурсией.

В рамках этой задачи требуется сравнить эмпирические оценки временной сложности двух реализаций алгоритма MERGE SORT:

- стандартной (с выделением дополнительной памяти) и
- гибридной, в которой на малых размерах массивов выполняется INSERTION SORT.

Для проведения экспериментов по оценке и сравнению временной сложности MERGE и MERGE+INSERTION SORT **в зависимости от того, для какого размера массива вызывается INSERTION SORT**, необходимо подготовить тестовые данные:

1. Массивы, которые заполнены случайными значениями в некотором диапазоне.
2. Массивы, которые отсортированы в обратном порядке по невозрастанию.
3. Массивы, которые «почти» отсортированы. Их можно получить, обменяв местами небольшое количество пар элементов в полностью отсортированном массиве.

Зафиксируем следующие параметры для подготовки тестовых данных:

- размеры массивов – от 500 до 4000 с шагом 100 и
- диапазон случайных значений – от 0 до 3000.

Для удобства подготовки тестовых данных, сгенерируйте для каждого случая массив максимальной длины (4000), из которого выбирайте подмассив необходимого размера (500, 600, 700, … элементов).

Измерение времени работы рассматриваемых алгоритмов производится с помощью встроенной библиотеки `std::chrono`. Например:

```
auto start = std::chrono::high_resolution_clock::now();
mergeSort(A, l, r);
auto elapsed = std::chrono::high_resolution_clock::now() - start;
long long millisec =
std::chrono::duration_cast<std::chrono::milliseconds>(elapsed).count();
```

Обратите внимание, что для более точного замера времени потребуется закрыть остальные программы и отключиться от сети (желательно). Кроме того, необходимо провести замеры времени несколько раз (количество на ваше усмотрение) и усреднить результаты.

1. Проведите измерения времени работы стандартной реализации алгоритма MERGE SORT и представьте результаты в виде трех (групп) графиков для каждой категории тестовых данных.
2. Проведите измерения времени работы гибридного алгоритма MERGE+INSERTION SORT в зависимости от того, для какого размера массива вызывается INSERTION SORT: 5, 10, 20 и 50 элементов. В рамках этих случаев представьте результаты в виде трех (групп) графиков для каждой категории тестовых данных.
3. Опишите полученные вами результаты и представьте выводы о сравнении временных затрат рассматриваемых алгоритмов.

Помимо графиков и пояснений, приложите к ответу:

1. Реализацию генератора тестовых данных.
2. Реализацию гибридного алгоритма MERGE+INSERTION SORT.
3. Исходные данные, использованные для построения графиков.

Задание A3* **(20 баллов)** Гибридная сортировка QUICK+HEAP SORT

Постановка этой задачи аналогична A2, только теперь необходимо провести эксперименты по сравнению временной сложности QUICK и QUICK+HEAP SORT **в зависимости от того, для какого размера вызывается HEAP SORT**.

Используйте тот же набор тестовых массивов, который был подготовлен в задаче A2.

Блок P. Реализация различных алгоритмов сортировки

Полные спецификации заданий доступны в системе Яндекс.Контест.

Задание P1 (5 баллов) HEAP SORT

В рамках данной задачи требуется реализовать сортировку одномерного целочисленного массива по неубыванию с применением бинарной max-кучи.

Задание P2 (5 баллов) COUNTING SORT

В рамках данной задачи требуется реализовать сортировку подсчетом по неубыванию для одномерного целочисленного массива.

Задание P3 (6 баллов) 256-RADIX SORT

В рамках данной задачи требуется реализовать поразрядную сортировку одномерного целочисленного массива по неубыванию. В качестве основания системы счисления для обработки исходных данных используйте 256, т.е. разрядом будет выступать байт.