Лабораторная работа №6

Анализ алгоритмов при использовании различных структур данных

Дисциплина: Операционные системы

Вариант: 5

Алгоритм: удаление дубликатов (формирование уникальной коллекции)

---

1. Состав бригады

- **Рябчиков А.Д.**
- **Лойко С.И.**

---

2. Цель работы

Исследовать влияние различных структур данных на эффективность выполнения алгоритма. Провести экспериментальное измерение времени работы алгоритма **удаления дубликатов** (получение коллекции, содержащей только уникальные элементы) при использовании:

- ArrayList
- LinkedList
- ArrayDeque

---

3. Постановка задачи

1. Реализовать алгоритм удаления дубликатов из коллекции.
2. Провести испытания при использовании трёх структур данных: ArrayList, LinkedList, ArrayDeque.
3. Измерить время выполнения на наборах данных увеличивающегося размера.
4. Для статистической устойчивости выполнить многократные повторы измерений и вычислить среднее значение времени.
5. Представить результаты в виде таблиц и сделать выводы по эффективности.

---

4. Теоретические сведения

Удаление дубликатов реализовано следующим образом: выполняется последовательный проход по исходной коллекции и поддерживается множество уже встреченных значений `HashSet`. Если элемент встречается впервые, он добавляется в результирующую коллекцию; повторные появления пропускаются. Порядок первых вхождений сохраняется за счёт последовательного обхода.

Оценка сложности:

- **Средняя временная сложность:** `O(N)` (операции `HashSet.add()` в среднем выполняются за `O(1)`)
- **Дополнительная память:** `O(N)` (для хранения множества уже встреченных элементов и результирующей коллекции)

---

5. Экспериментальные данные

Каждое значение — **среднее время выполнения (мс)** по серии повторений (усреднение по многим прогонам).

5.1. Результаты для ArrayList

Размер данных	Среднее время (мс)
100000	3
250000	10
500000	36
1000000	79
2000000	206

5.2. Результаты для LinkedList

Размер данных	Среднее время (мс)
100000	4
250000	12
500000	45
1000000	143
2000000	348

5.3. Результаты для ArrayDeque

Размер данных	Среднее время (мс)
100000	3
250000	10
500000	37
1000000	89
2000000	232

---

6. Аналитическая оценка и сопоставление с экспериментом

Теоретически алгоритм удаления дубликатов через `HashSet` должен демонстрировать **приблизительно линейный рост времени** при увеличении размера входных данных. Экспериментальные результаты в целом подтверждают эту тенденцию: при росте объёма данных время увеличивается кратно размеру, без квадратичного «взрыва», характерного для сортировок сравнениями уровня $O(N^2)$.

Наблюдения по структурам данных

1. **ArrayList и ArrayDeque показывают близкие результаты** на всех размерах. Это ожидаемо, поскольку обе структуры обеспечивают эффективный последовательный обход и добавление в результирующую структуру.
2. **LinkedList заметно медленнее** на больших объёмах. Основные причины — более высокая накладная стоимость узлов списка (объекты + ссылки) и менее эффективная работа с памятью/кэшированием при последовательном обходе по сравнению с массивоподобными структурами.

---

7. Обсуждение результатов

1. Алгоритм удаления дубликатов через `HashSet` существенно эффективнее подходов с попарным сравнением, так как снижает асимптотику до среднего $O(N)$.
2. Разница между структурами данных проявляется сильнее на больших размерах входных данных (1–2 млн элементов).
3. Наиболее устойчивые по скорости структуры в эксперименте — **ArrayList** и **ArrayDeque**.
4. **LinkedList** демонстрирует наибольшие затраты времени на обработку, особенно при росте объёма данных.

---

8. Выводы

- Реализован и исследован алгоритм удаления дубликатов (формирование уникальной коллекции с сохранением порядка первых вхождений) для ArrayList, LinkedList и ArrayDeque.
- Экспериментальные данные согласуются с теоретической оценкой средней сложности `O(N)`.
- Наиболее эффективные структуры по измерениям: **ArrayList** и **ArrayDeque**.
- LinkedList показал худшую производительность на больших объёмах данных, что связано с накладными расходами структуры и особенностями доступа к памяти.