ОНЛАЙН-ОБРАЗОВАНИЕ

OTUS

Сравним For-ы или Stream-ы: что использовать?

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

OTUS

План занятия

- В чем суть спора?
- Измерения
- Идеология
- Вывод

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

OTUS

В чем суть спора?

«Староверы»:

циклы работают намного быстрее. Используем циклы.

«Хипстеры»:

в моде функциональный стиль. Используем streams.

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

OTUS

Измеряем скорость работы

Решаем задачу фильтрации массива и ArrayList.

Для измерения пользуемся JMH.

Измерения делались на
Intel® Core™ i7-8700K CPU @ 3.70GHz × 12

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

Результаты измерений

Зависимость времени работы от кол-ва элементов, мс

Тест	100_000	200_000	500_000	1_000_000	10_000_000	100_000_000
filterArrayFor	2.910	3.952	11.363	12.432	69.356	670.561
filterFor	4.325	6.486	12.393	15.527	79.114	790.759
filterArray Stream	3.656	4.972	9.089	12.929	93.983	991.561
filterStream	3.403	4.404	8.991	13.588	93.268	967.014
filterStream Unordered	3.490	6.879	9.049	12.858	94.165	960.990

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

OTUS

Первые выводы

1. Если надо обработать менее 1 млн. элементов, то разницу в скорости зафиксировать трудно.

2. Обработка массива в цикле быстрее примерно на 30%

3. Unordered stream работает не медленнее, но «хуже не будет»

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

OTUS

Более реалистичное измерение

Добавим процедуру чтения текстового файла на 100 т. строк
Зависимость времени работы от кол-ва элементов, мс

Тест	100_000	200_000	500_000	1_000_000	10_000_000	100_000_000
filterArrayFor	34.249	48.315	52.841	58.055	91.756	818.125
filterFor	39.893	48.908	48.122	55.326	106.619	857.030
filterArray Stream	33.266	46.101	39.541	50.167	124.710	1031.291
filterStream	32.713	43.475	36.621	46.467	117.754	998.899
filterStream Unordered	31.935	40.604	37.716	54.835	118.851	1000.393

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

Вторые выводы

1. Как и раньше:
   Если надо обработать менее 1 млн. элементов,
   то разницу в скорости зафиксировать трудно.
2. Обработка массива в цикле быстрее примерно на 26% (время сократилось с 30%)
3. Unordered stream работает не медленнее, но «хуже не будет»

<u>Предсказуемый результат:</u>
чем меньше доля операций над коллекциями в общем объеме операций, тем менее заметно замедление streams.

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

OTUS

А если с допингом?

У streams есть режим работы parallel.

Кто думает, что применение parallel
в соревнованиях – читерство ?

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

OTUS

Что об этом думает Brian Goetz?

I like to look at this as having chosen a design center that recognizes that
sequential is a degenerate case of parallel,
rather than treating parallel as the "weird bonus mode".

http://mail.openjdk.java.net/pipermail/lambda-dev/2014-February/011870.html

Т.е. последовательный режим –
частный случай параллельного.

OTUS

Parallel Strems

Parallel streams использует ForkJoin Pool (common).

Как это можно применить.

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

OTUS

Parallel Strems

Как убедиться,
сколько потоков будет создано?

Как вывести имя потока-worker-а?

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

Результаты измерений

Зависимость времени работы от кол-ва элементов, мс

Степень параллелизма – 10.

Тест	100_000	200_000	500_000	1_000_000	10_000_000	100_000_000
filterFor	4.325	6.486	12.393	15.527	79.114	790.759
filterStream	3.403	4.404	8.991	13.588	93.268	967.014
filterStream Parallel	6.813	15.560	20.155	25.786	47.363	262.555

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

OTUS

Параллельные выводы

1. Если надо обработать менее 1 млн. элементов, то параллельность только замедлит работу.

2. На *достаточно больших объемах работы* параллельность может ускорить работу в разы.

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

Что получается

Забываем про циклы и переходим на Streams ?

OTUS

Пример мутации

Забываем про циклы и переходим на Streams ?

Пример.

MutationArrayList

Идеология Strems

<Источник> ->

[операция1] ->

[операция2] ->

[операция3] ->

(результат).

Принципиальное правило – источник не должен меняться и не должно быть побочных действий.

Итоговые выводы

1. Параллельность имеет смысл только на больших объемах данных и при наличии вычислительных ресурсов.

2. О преимуществах циклов имеет смысл говорить только на больших объемах данных.

3. Если производительность (в том числе и нагрузка на gc) важнее выразительности, то циклы предпочтительнее.

Не стоит впадать в крайности,
под каждую задачу нужен свой инструмент.

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

OTUS

Вопросы?

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

OTUS

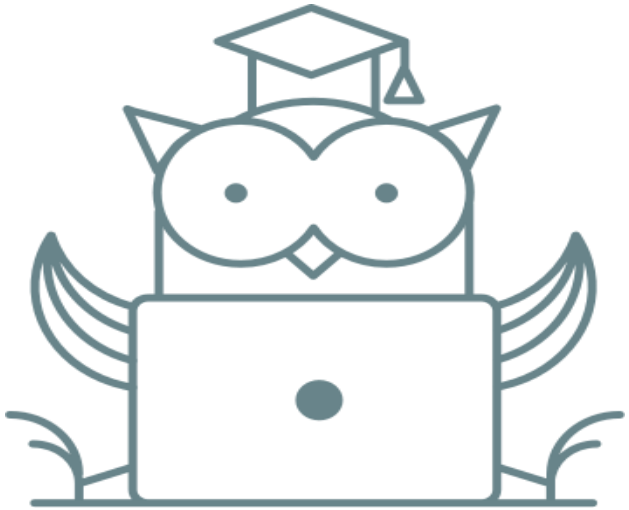
Опрос !!!

Пожалуйста, пройдите опрос.

Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer

OTUS

Спасибо за внимание!



Петрелевич Сергей
Oracle Certified Professional, Java SE 8 Programmer