Сабиров И.В.

Пояснительная записка к тестовому заданию

Постановка задачи

В качестве тестового задания было предложено реализовать программу, имитирующую работу жесткого диска, и оценить среднее время, затрачиваемое на операцию чтение/запись.

При моделировании были сделаны допущения:

- носитель имеет один диск с одной рабочей поверхностью;
- скорость перемещения головки между соседними дорожками постоянна для любой пары дорожек;
- количество секторов на дорожке не зависит от того, является ли дорожка внешней или внутренней, и является постоянной величиной;
- контроллер жесткого диска имеет ограниченный буфер, работа с которым должна производится по одному из следующих алгоритмов:
  - FCFS;
  - SSF;
  - Elevator algorithm;
- запросы на чтение/запись являются случайными событиями, образующими стационарный поток Пуассона;
- передача запросов на чтение/запись из интерфейса в буфер, из буфера на выполнение и возврат результата осуществляются мгновенно.

FCFS (First Come, Firts Served) – модель обработки, при которой запросы выполняются по одному в порядке их поступления в очередь.

SSF (Shortest Seek First) – модель, при которой обрабатывается запрос к ближайшему цилиндру (относительно текущего положения).

Elevator algorithm – модель, при которой фиксируется направление «движения» (к внутренней дорожке или к внешней) и сканирование очереди запросов выполняется с учетом этого направления. Следует иметь в виду, что нумерация дорожек производится начиная с внешней, тогда если зафиксировано направление к внутренней дорожке, то следующим обрабатываемым запросом станет тот, который обращен к ближайшему цилиндру (относительно текущего положения), номер которого больше номера текущего цилиндра. Если такие запросы отсутствуют, то направление меняется на противоположное.

Исходные данные и реализация модели

В качестве источника исходных данных было выбрано описание продуктовой линейки жестких дисков Seagate Barracuda [1]. В модели были приняты следующие константы [1, сс. 11-14]:

- число оборотов диска в минуту – 7200;
- количество цилиндров – 16383;
- размер сектора – 512 байт.

Существует оценка параметра «Track-to-track» [1, с. 16]: операция чтения - 1 мс, операция записи - 1.2 мс. Этот параметр представляет собой среднее время всех возможных переходов между любыми двумя цилиндрами диска для чтении или записи, соответственно. Будем предполагать, что число операций чтения и записи равно, тогда справедливо положить

$$\nu = 2\frac{tract\_to\_track}{N_c}, \qquad \text{здесь } tract\_to\_track = \frac{1+1.2}{2} \text{ – средняя скорость чтения/записи,}$$

N_c – количество цилиндров, а ν – постоянная скорость перемещения головки между двумя соседними дорожками.

Исходя из допущения, что работа чтение/запись производится только с одной стороной одного диска, среднее количество секторов на дорожке может быть определено по формуле

$$N_s = \frac{S_d}{N_c\, S_s},$$

здесь, как и выше, N_c – количество цилиндров, а S_d и S_s - размеры диска и сектора в байтах соответственно. Таким образом, если на вход программы подается число, определяющее объем диска в гигабайтах, следует иметь в виду, что 1 Гбайт $= 10^9$ байт [1, с. 2], тогда результирующая формула для определения среднего количества секторов может быть записана в виде

$$N_s = \frac{V}{N_c\, S_s}10^9, \qquad V \text{ – объем диска в ГБ.}$$

Эта формула справедлива для всякой дорожки диска в силу допущения о постоянстве количества секторов на всех дорожках.

Вероятность возникновения за некоторое время k событий в стационарном процессе Пуассона определяется формулой [2]

$$P\{\xi = k\} = \frac{a^k}{k!}e^{-a}, \quad k = 0, 1, 2, \ldots; \qquad a = \lambda\tau, \tag{1}$$

здесь τ – рассматриваемый интервал времени, а λ – интенсивность потока. Вероятность P возникновения хотя бы одного события в рассматриваемом интервале τ:

$$P = 1 - P\{\xi = 0\} = 1 - e^{-\lambda\tau},$$

отсюда согласно (1) следующее событие наступит через

$$\tau = -\frac{1}{\lambda}\, ln(r),$$

здесь r - случайная величина с равномерным распределением в интервале $(0; 1)$.

В реализации программы для определения временного интервала τ между событиями и номеров дорожек и секторов запросов на чтение/запись использовались функции генерирования псевдослучайных чисел. При этом, если в случае с определением временного промежутка оказалось достаточно стандартной функции «rand()», то для номеров секторов, а в общем случае и для дорожек (может быть в будущем), отрезка [0;32767] не достаточно. Поэтому для последних был использован стандартный класс «std::default_random_engine», который генерирует числа в отрезке [1;2147483646].

Каждый предложенный алгоритм работы с буфером контроллера порождает свой класс, имитирующий работу жесткого диска и позволяющий оценить среднее время операции чтения/записи. С точки зрения проектирования эту задачу можно разрешить применением наследования или реализацией класса-шаблона с последующей специализацией функций работы с буфером контроллера.

Был выбран промежуточный вариант. Каждый производный класс реализует собственный алгоритм работы с буфером контроллера с помощью выделенного для этого виртуального метода «get_io_task_from_que()». Для остальных методов, так или иначе обращающихся к очереди контроллера, реализованы шаблонные функции, специализации которых используются в каждом классе самостоятельно. UML диаграмма представлена на Рис. 1.

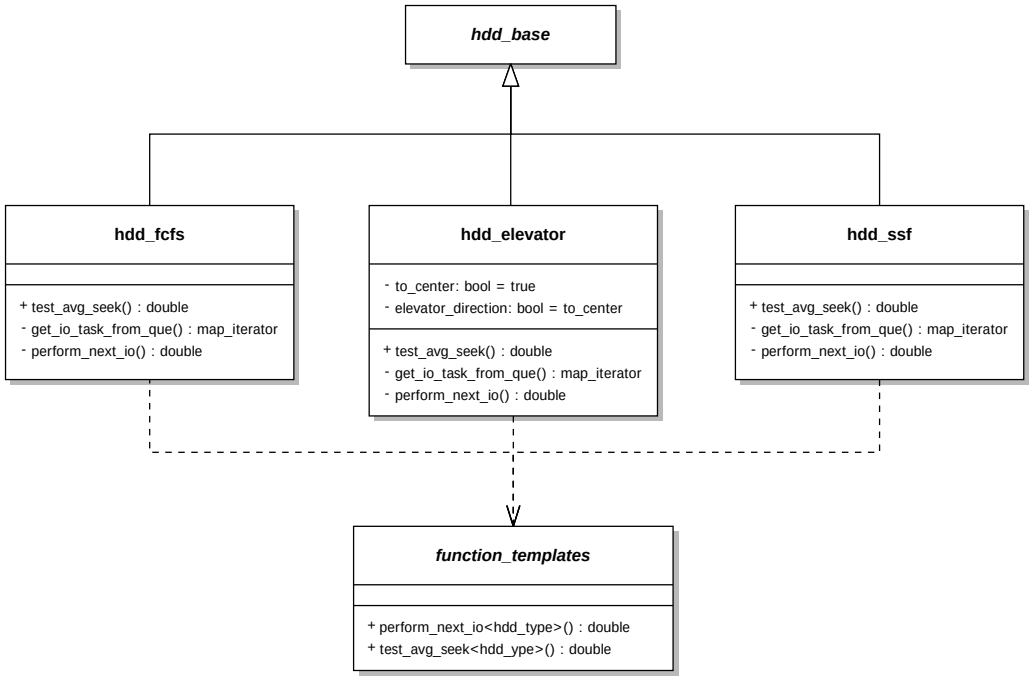

Рис. 1: UML диаграмма классов

Результаты экспериментов

Некоторые результаты численных экспериментов представлены ниже.

На Рис. 2 проиллюстрирована зависимость среднего времени выполнения запросов чтение/запись от интенсивности потока Пуассона. Размер буфера контроллера был взят равным 300 запросам.

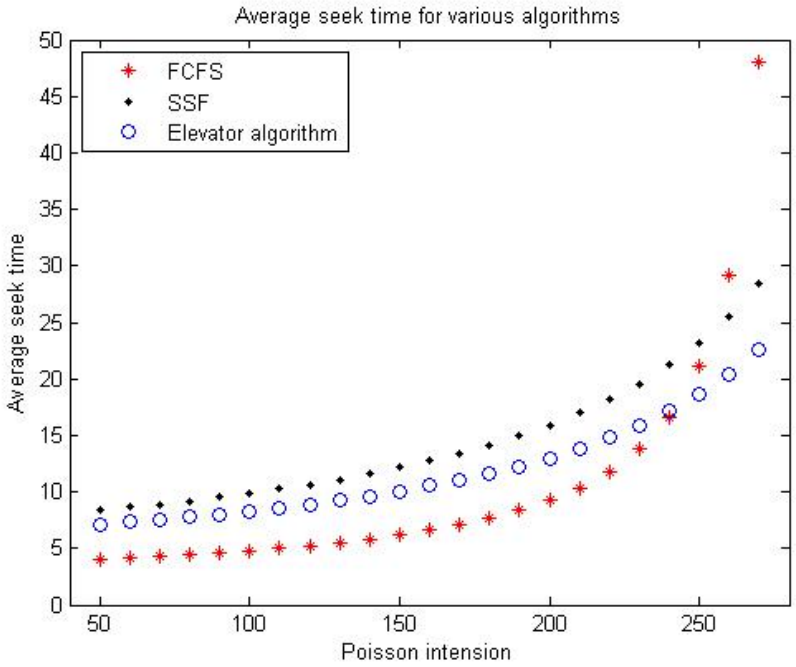

Рис. 2: Среднее время выполнения запросов

В таблице 1 представлено заполнение буфера контроллера каждым алгоритмом при разных значениях интенсивности потока Пуассона. Были приняты следующие обозначения: $\bar{t}$ – среднее время выполнения запроса чтение/запись (мс); q_{max} – максимальное количество запросов в буфере контроллера; $\overline{q}$ – среднее количество запросов в буфере контроллера; m – количество необработанных запросов в результате переполнения буфера контроллера.

Из таблицы 1 и Рис. 2 видно, что контроллер жесткого диска, работающий с буфером по алгоритму FCFS, менее надежен при большей интенсивности потока запросов. Среднее время обработки запроса становится многократно больше, чем в случае использования двух других алгоритмов; раньше появляются необработанные из-за переполнения буфера запросы.

Также является показательным среднее количество запросов в буфере контроллера. При одних и тех же значениях в разных алгоритмах, максимальное количество запросов в буфере контроллера в случае алгоритма FCFS больше, что говорит о менее экономном использовании ресурсов контроллера.

Однако при небольших значениях интенсивности потока запросов и при наличии необходимого объема памяти для буфера контроллера, использование алгоритма FCFS оправданно малой величиной времени обработки запроса. Это вполне естественно в силу того, что этот алгоритм не предполагает задержки запроса в буфере контроллера.

Показатели двух других алгоритмов близки, но в большинстве случаев лифтовый ал-

Таблица 1: Заполнение буфера контроллера

λ	FCFS				SSF				Elevator algorithm			
	$\bar{t}$	q_{max}	$\bar{q}$	m	$\bar{t}$	q_{max}	$\bar{q}$	m	$\bar{t}$	q_{max}	$\bar{q}$	m
50	4.01	8	1.21	0	8.39	10	1.43	0	7.12	9	1.37	0
60	4.14	8	1.26	0	8.64	10	1.54	0	7.31	9	1.45	0
70	4.29	9	1.32	0	8.91	10	1.65	0	7.53	10	1.55	0
80	4.44	9	1.38	0	9.20	12	1.76	0	7.76	10	1.65	0
90	4.61	10	1.45	0	9.52	14	1.89	0	8.01	11	1.76	0
100	4.80	11	1.52	0	9.87	14	2.03	0	8.28	13	1.87	0
110	5.02	12	1.60	0	10.24	15	2.18	0	8.58	15	1.99	0
120	5.26	14	1.68	0	10.66	17	2.34	0	8.91	15	2.13	0
130	5.53	15	1.78	0	11.11	16	2.51	0	9.26	15	2.27	0
140	5.84	18	1.89	0	11.60	19	2.70	0	9.64	16	2.43	0
150	6.20	21	2.01	0	12.14	21	2.91	0	10.06	20	2.59	0
160	6.61	22	2.15	0	12.73	20	3.13	0	10.53	19	2.78	0
170	7.09	23	2.31	0	13.39	24	3.38	0	11.05	19	2.98	0
180	7.67	26	2.49	0	14.13	27	3.66	0	11.63	23	3.21	0
190	8.37	27	2.71	0	14.95	27	3.97	0	12.27	23	3.46	0
200	9.23	30	2.98	0	15.88	27	4.31	0	12.99	27	3.73	0
210	10.33	33	3.31	0	16.96	31	4.71	0	13.83	27	4.05	0
220	11.78	35	3.75	0	18.16	32	5.15	0	14.77	28	4.41	0
230	13.75	41	4.33	0	19.59	35	5.67	0	15.86	31	4.81	0
240	16.59	54	5.16	0	21.22	40	6.27	0	17.12	34	5.29	0
250	21.05	63	6.45	0	23.14	44	6.96	0	18.63	39	5.84	0
260	29.11	84	8.77	0	25.49	46	7.82	0	20.42	44	6.50	0
270	48.00	123	14.17	0	28.43	52	8.87	0	22.56	61	7.30	0
280	144.22	300	41.59	92	31.92	62	10.14	0	25.40	54	8.32	0
290	912.58	300	260.98	121091	36.57	67	11.81	0	28.98	76	9.62	0
300	1005.2	300	287.32	356533	42.65	76	14.01	0	33.65	79	11.32	0

горитм немногим предпочтительнее. Оба алгоритма требовательны к объему памяти, предназначенному для буфера контроллера.

Производитель гарантирует [1, сс. 12,13,16], что среднее время выполнения запросов на чтение и запись не превышает 8.5 и 9.5 мс соответственно. Имея в виду прежнюю договоренность о равном количестве запросов на чтение и запись, для сравнения следует принять величину 9 мс. Этой величине соответствует интенсивность потока 111 операций чтение/запись в секунду. Из таблиц 1, 2 видно, что только алгоритм SSF не удовлетворяет этому требованию. Возможно для этого алгоритма проблема может быть решена при использовании кеширования запросов, но исследование этого вопроса выходит за рамки данного тестового задания.

Таким образом, для дальнейших испытаний принято:

- интенсивность потока Пуассона - 111;
- буфер контроллера способен вместить 16 запросов.

Для разработанной модели помимо соответствия рассчитанных результатов справоч-

Таблица 2: Заполнение буфера контроллера (детализация)

λ	FCFS				SSF				Elevator algorithm			
	$\overline{t}$	q_{max}	$\overline{q}$	m	$\overline{t}$	q_{max}	$\overline{q}$	m	$\overline{t}$	q_{max}	$\overline{q}$	m
109	4.9948	12	1.5895	0	10.1998	15	2.1617	0	8.5485	15	1.9803	0
110	5.0175	12	1.5978	0	10.2410	15	2.1771	0	8.5809	15	1.9931	0
111	5.0405	12	1.6061	0	10.2813	15	2.1927	0	8.6120	15	2.0061	0
112	5.0635	12	1.6145	0	10.3216	15	2.2084	0	8.6456	15	2.0194	0
113	5.0870	12	1.6231	0	10.3636	15	2.2244	0	8.6764	15	2.0324	0
114	5.1108	12	1.6316	0	10.4069	15	2.2407	0	8.7087	15	2.0457	0

ной документации производителя важны такие объективные показатели модели как устойчивость и сходимость.

Расчеты, представленные в таблицах 1, 2 произведены для жесткого диска объемом 2 ГБ. В таблицах 3, 4 представлены те же показатели в интервале интенсивности потока $[109; 114]$ для дисков 10 ГБ и 200 МБ соответственно.

Таблица 3: Заполнение буфера контроллера (объем диска 10 ГБ)

λ	FCFS				SSF				Elevator algorithm			
	$\overline{t}$	q_{max}	$\overline{q}$	m	$\overline{t}$	q_{max}	$\overline{q}$	m	$\overline{t}$	q_{max}	$\overline{q}$	m
109	4.9980	11	1.5898	0	10.2127	14	2.1637	0	8.5528	13	1.9808	0
110	5.0207	11	1.5981	0	10.2527	14	2.1790	0	8.5862	13	1.9938	0
111	5.0438	11	1.6065	0	10.2930	15	2.1947	0	8.6169	13	2.0068	0
112	5.0667	11	1.6150	0	10.3337	15	2.2105	0	8.6495	13	2.0200	0
113	5.0902	11	1.6236	0	10.3742	15	2.2264	0	8.6812	13	2.0331	0
114	5.1137	11	1.6322	0	10.4173	15	2.2424	0	8.7145	13	2.0465	0

Таблица 4: Заполнение буфера контроллера (объем диска 200 МБ)

λ	FCFS				SSF				Elevator algorithm			
	$\overline{t}$	q_{max}	$\overline{q}$	m	$\overline{t}$	q_{max}	$\overline{q}$	m	$\overline{t}$	q_{max}	$\overline{q}$	m
109	4.9903	11	1.5888	0	10.2003	14	2.1625	0	8.5501	12	1.9806	0
110	5.0128	11	1.5969	0	10.2378	14	2.1777	0	8.5805	12	1.9933	0
111	5.0358	12	1.6054	0	10.2777	16	2.1931	0	8.6104	12	2.0060	0
112	5.0589	12	1.6138	0	10.3136	16	2.2082	0	8.6410	13	2.0190	0
113	5.0821	12	1.6222	0	10.3540	16	2.2240	0	8.6727	13	2.0320	0
114	5.1053	12	1.6307	0	10.3933	16	2.2396	0	8.7008	13	2.0446	0

Малые изменения рассчитанных показателей, представленных в таблицах 2 - 4 соответствуют небольшим изменениям интенсивности потока запросов и объема жестких дисков. Этот факт свидетельствует об устойчивости разработанной модели.

На рисунках 3-5 представлены результаты теста на сходимость модели диска объемом 2 ГБ. Проведено несколько тестов с разным количеством запросов чтение/запись в потоке.

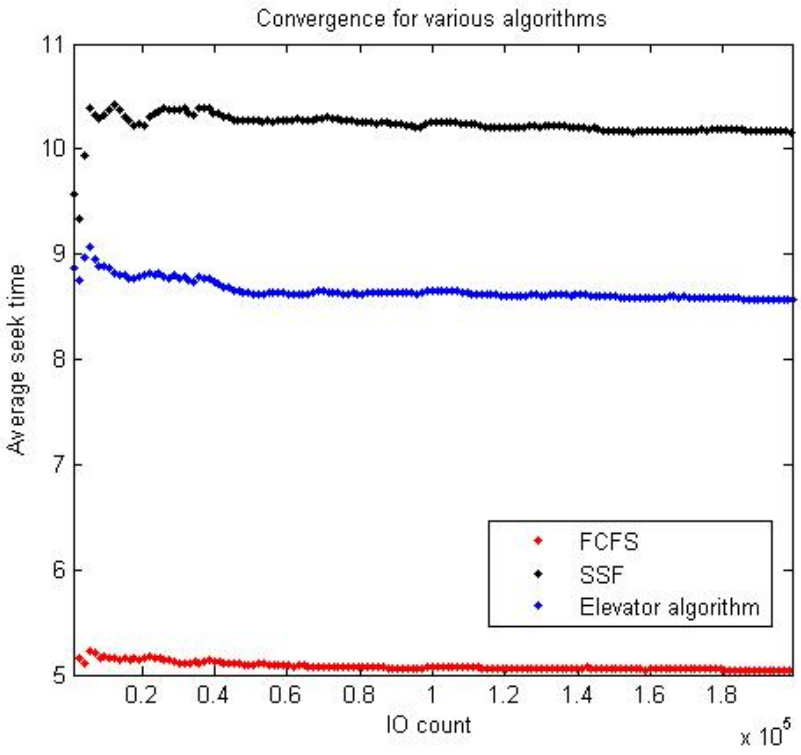

Рис. 3: Сходимость (объем диска 2 ГБ)

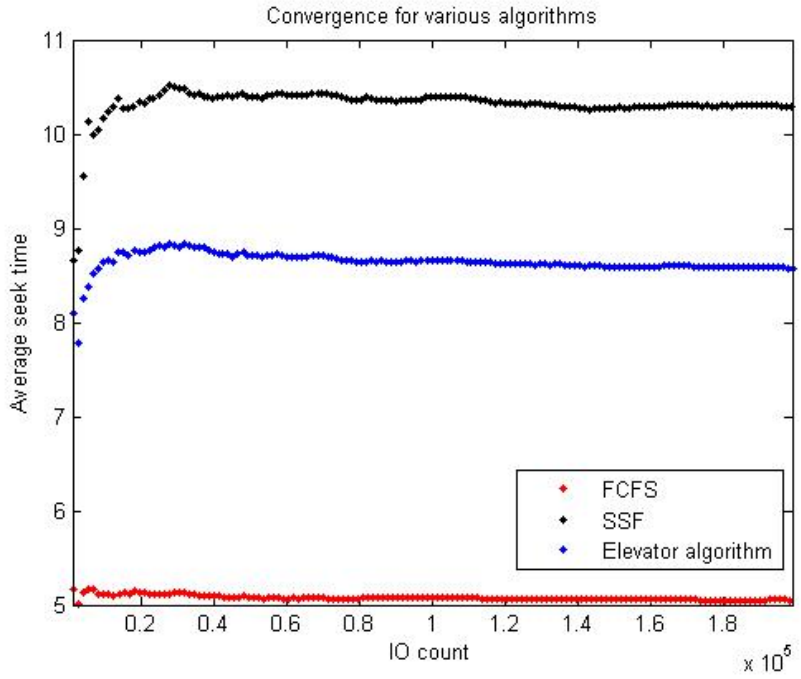

Рис. 4: Сходимость (объем диска 10 ГБ)

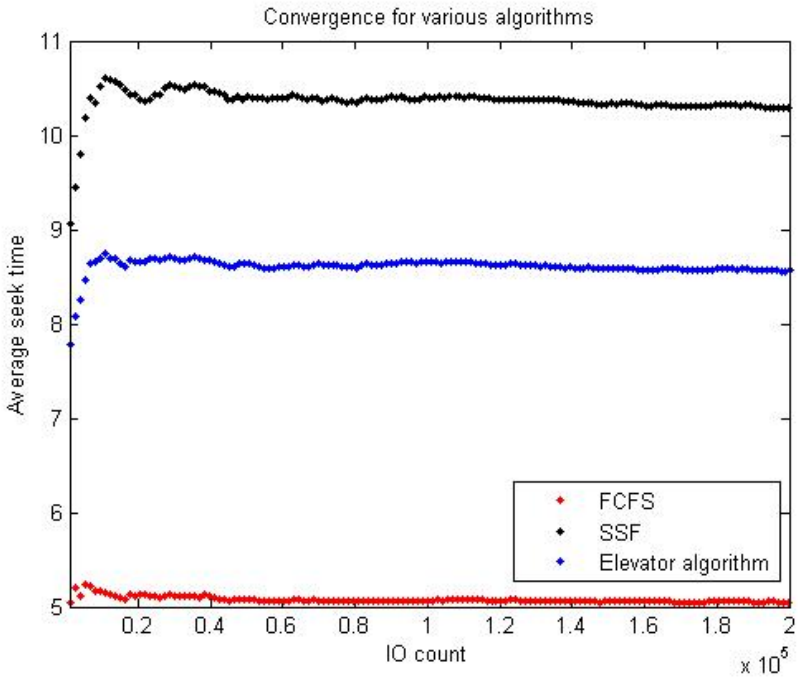

Рис. 5: Сходимость (объем диска 200 МБ)

Представлена зависимость рассчитываемого среднего времени выполнения запроса от количества этих запросов в потоке. Тесты были проведены для дисков объемом 10 ГБ, 2 ГБ, 200 МБ.

На всех рисунках 3-5 при малом количестве запросов видна осцилляция, которая затухает при увеличении количества запросов в потоке. Однако можно заметить, что при дальнейшем увеличении числа запросов вычисляемые значения стабилизируются не около прямой, параллельной оси абсцисс, а скорее около прямой, имеющей небольшой наклон к этой оси. Вероятнее всего, это явление связано с тем, что в модели используется псевдослучайное распределение.

Основываясь на данных о сходимости, можно выбрать число запросов в потоке для проведения множества экспериментов.

Также представляет интерес среднее время выполнения запроса чтение/запись к определенной дорожке диска. На рисунках 6-8 представлено распределение времени выполнения запроса чтение/запись по дорожкам диска объемом 10 ГБ при разных величинах потока запросов. Для демонстрации того, что величины потоков разные, помимо графиков алгоритмов представлен также график, иллюстрирующий количество запросов к каждой дорожке в данном эксперименте.

При принятом допущении о том, что число секторов на всех дорожках равно, для диска объемом 10 ГБ количество секторов на каждой дорожке будет 1192. Этим объясняется уменьшение разброса средних величин при увеличении количества запросов чтение/запись в потоке(в среднем от 35 до 3500 запросов к каждой дорожке).

Выбор между точностью модели и скоростью ее работы должен осуществляется исходя из требований заказчика. В данном случае такие требования отсутствуют, потому количество запросов в потоке было выбрано как половина общего количества секторов диска, объем которого задается пользователем модели.

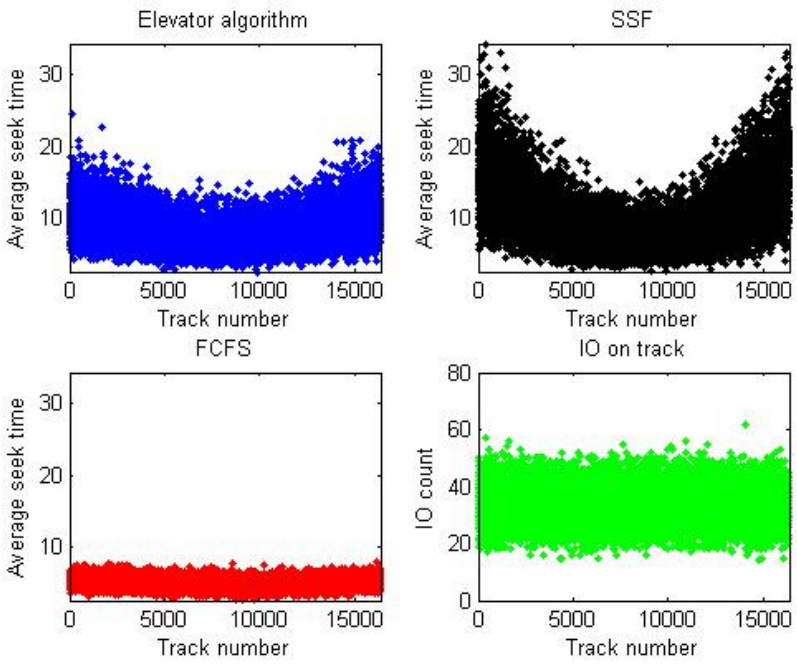

Рис. 6: Распределение запросов (число запросов в потоке $\approx 5 \cdot 10^5$)

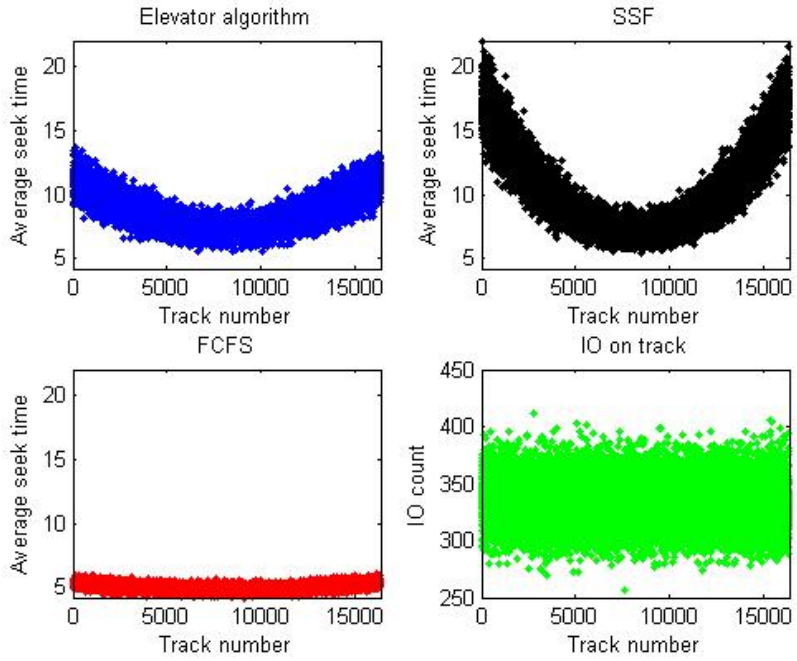

Рис. 7: Распределение запросов (число запросов в потоке $\approx 5 \cdot 10^6$)

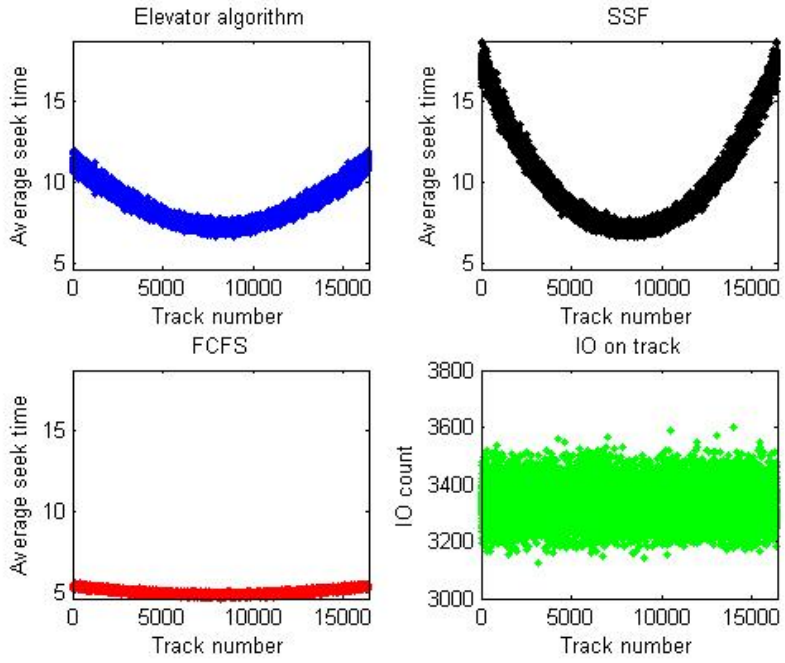

Рис. 8: Распределение запросов (число запросов в потоке $\approx 5 \cdot 10^7$)

На рисунке 9 представлено распределение соответствующее последнему допущению о величине потока. Для оценки изменения точности расчетов в таблице 5 представлены показатели алгоритмов при разных величинах потоков запросов чтение/запись.

Таблица 5: О выбранной точности

П	T	FCFS				SSF				Elevator algorithm			
		$\overline{t}$	q_{max}	$\overline{q}$	m	$\overline{t}$	q_{max}	$\overline{q}$	m	$\overline{t}$	q_{max}	$\overline{q}$	m
$5 \cdot 10^7$	372	5.0427	13	1.6063	0	10.2852	16	2.1937	0	8.6160	15	2.0065	0
$1 \cdot 10^7$	84	5.0413	13	1.6059	0	10.2828	15	2.1931	0	8.6134	14	2.0060	0
$5 \cdot 10^6$	34	5.0442	11	1.6067	0	10.2929	15	2.1949	0	8.6172	13	2.0070	0
$5 \cdot 10^5$	3	5.0507	11	1.6066	0	10.2505	15	2.1913	0	8.6036	11	2.0061	0

В таблице 5: П – величина потока запросов чтение/запись, T – время работы программы в секундах, остальные обозначения имеют тот же смысл что и в таблицах 1-4.

Из рисунков 6-9 можно сделать вывод о причинах преимущества лифтового алгоритма над алгоритмом SSF. В силу того, что в алгоритме SSF выбирается наиболее короткий путь до следующего запроса, головка жесткого диска дольше находится именно возле средних номеров дорожек, а запросы, обращенные к начальным и конечным номерам дорожек вынуждены простаивать в буфере контроллера. По этой причине среднее время обработки «дальних» запросов увеличивается, что негативно сказывается на общем показателе среднего времени обработки запросов с помощью этого алгоритма.

Лифтовый алгоритм, оставаясь наиболее надежным, подвержен таким проблемам менее. Отсюда следует, что среди рассмотренных алгоритмов именно лифтовый является предпочтительным.

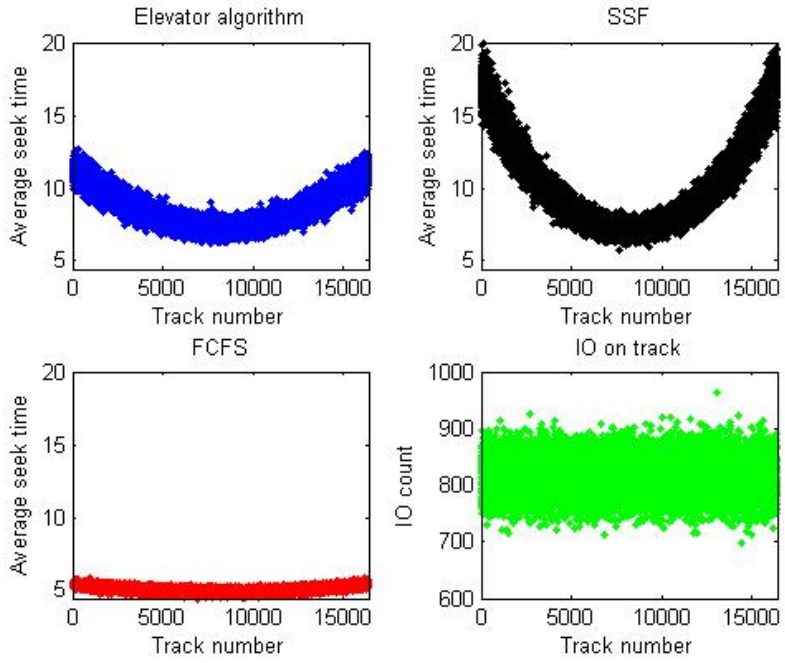

Рис. 9: Распределение запросов (число запросов в потоке $\approx 1 \cdot 10^7$)

Для ускорения процесса расчета была реализована многопоточность с помощью библиотеки boost::thread. Расчеты для каждого экземпляра класса с разными алгоритмами ведутся в параллельных потоках.

Исходные коды программы, вспомогательные скрипты, этот отчет и использованные в нем графические изображения доступны в репозитарии:

https://github.com/verentil/HDD_queue

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Barracuda: Product manual. Seagate Technology LLC, PN: 100686584, Oct. 2012.

2. Королюк В.С. Справочник по теории вероятностей и математической статистике. М.: Наука, 1985.