УДК 007, 502.3

ВЫБОР ПОСЛЕДОВАТЕЛЬНОСТИ ШАГОВ ДЛЯ АЛГОРИТМА СОРТИРОВКИ ШЕЛЛА

Сычёв Пётр Павлович

Доцент;
Государственный университет «Дубна»;
Россия, 141980, Московская обл., г. Дубна, ул. Университетская, д. 19;
e-mail sychov.p.p@uni-dubna.ru.

В работе приведены результаты эмпирического исследования нескольких последовательностей шагов для алгоритма сортировки Шелла. Показана достаточно высокая эффективность таких последовательностей в сравнении с другими, хорошо известными последовательностями.

Ключевые слова: сортировка, сортировка Шелла.

Для цитирования:
Сычёв П. П. Выбор последовательности шагов для алгоритма сортировки Шелла // Системный анализ в науке и образовании: сетевое научное издание. 2023. № 1. С. 41–45. URL : https://sanse.ru/index.php/sanse/article/view/570.

CHOOSING A SEQUENCE OF STEPS FOR THE SHELL SORTING ALGORITHM

Sychov Petr P.

Associate professor;
Dubna State University;
19 Universitetskaya Str., Dubna, Moscow region, 141980, Russia;
e-mail: sychov.p.p@uni-dubna.ru.

The paper presents the results of an empirical study of several sequences of steps for the Shell sorting algorithm. A sufficiently high efficiency of such sequences is shown in comparison with other well-known sequences.

Keywords: Sort, ShellSort.

For citation:
Sychov P. P. Choosing a sequence of steps for the Shell sorting algorithm. *System analysis in science and education*, 2023;(1):41–45 (in Russ). Available from: https://sanse.ru/index.php/sanse/article/view/570.

Введение

Алгоритм сортировки Шелла (*ShellSort*) предложен в 1959 году [1] и занимает промежуточное положение между «медленными» алгоритмами сортировки с временной сложностью $O(N^2)$, где N – размер сортируемого массива (*InsertSort*, *BubbleSort* и т.д.) и «быстрыми» алгоритмами с временной сложностью $O(NlogN)$ (*QuickSort*, *SortMerge* и другие). Обычно временную сложность алгоритма *ShellSort* определяют как: $O(N^{\propto})$ где $1 < \propto < 2$. Значение $\propto$ зависит от выбора шагов алгоритма.

Алгоритм Шелла состоит в последовательном применении алгоритма сортировки вставками (*InsertSort*) с уменьшающемся шагом. Т.е. задается последовательность шагов (h_m h_{m-1} … h_1) такая что $h1=1$ и $h_k > h_{k-1}$. Массив, отсортированный с шагом p называется p-упорядоченным, т.е. все элементы этого массива, отстоящие друг от друга на расстояние, кратное p, упорядочены. Показано, что если p-упорядоченный массив отсортировать с шагом q, он останется p-упорядоченным, то есть он будет од-

новременно и p и q-упорядоченным. Поэтому к последней сортировке с шагом 1 массив уже значительно упорядочен и сложность последнего этапа будет пропорциональна N, а общая сложность алгоритма – лучше, чем N^2.

Хорошо изучены только некоторые последовательности шагов [2]. Генерация шагов можно производить либо по убыванию шагов, либо по их увеличению. Самый большой шаг должен быть сравним с размером массива, около $N/2$, чтобы на первом шаге сортировки в каждом подмножестве было только 2-3 элемента.

Методика

Для измерения характеристик различных последовательностей шагов была реализована функция сортировки массива целых чисел алгоритмом Шелла. Кроме массива чисел для сортировки в эту функцию передавался массив шагов, который использовался в алгоритме. Этот массив шагов готовился отдельно, в соответствии с выбранным алгоритмом генерации шагов, его размер зависел от размера сортируемого массива. Шаги располагаются в убывающем порядке.

Для каждого алгоритма генерации шагов сортировки генерировалось несколько массивов целых чисел разного размера, заполненных псевдослучайными значениями. Известно, что последовательность псевдослучайных чисел, генерируемых тем или иным генератором псевдослучайных чисел, полностью определяется начальным значением этого генератора. Чаще всего, в реальных программах, он инициируется текущим значением таймера, но его всегда можно задать явно. В нашем случае, для каждой серии прогонов использовалась одна и та же последовательность начальных значений.

Измерения

В таблице 1 приведены результаты тестовых прогонов программы для «медленных» последовательностей шагов. В первой строке приводится размер массива, в первом столбце – алгоритм генерации шагов. Элементы таблицы – усредненное по серии прогонов время работы функции сортировки для данного массива и данного набора шагов сортировки, округленное до целого числа миллисекунд.

Таблица 1. Результаты тестовых прогонов

Алгоритм	1000	2000	5000	10000	20000	50000	Линейный коэффициент
InsertSort	13	46	272	1092	4350	27131	1.97
$h_k = 2^k$	1	3	11	31	81	407	1.46
$h_k = 3^k$	1	3	11	36	77	262	1.46

Если временную сложность алгоритма *ShellSort* представить в виде $t = CN^{\alpha}$, то логарифмируя это выражение получим $logt = logC + \alpha logN$. Т.е. точки ($log\ t$, $log\ N$) должны аппроксимироваться прямой. Коэффициент ее наклона и будет представлять временную сложность в этом эксперименте. Коэффициент вычислялся по методу наименьших квадратов.

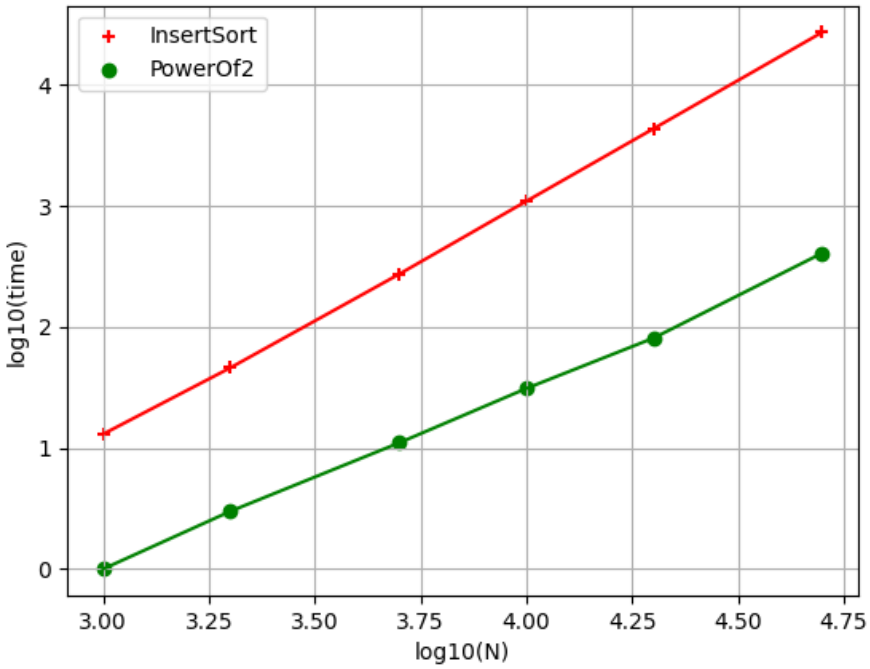

Рис. 1. График зависимости десятичного логарифма времени сортировки от размера массива

На рис. 1 приведен график зависимости десятичного логарифма времени сортировки от размера массива. Красный цвет соответствует обычной сортировке вставками, зеленый – последовательности шагов – степеней двух. График для последовательности шагов степеней трех не приводится, он практически совпадает с зеленым].

Временная сложность алгоритма сортировки вставками равна $O(N^2)$, что согласуется с результатом в таблице 1. Средняя временная сложность для двух других алгоритмов тоже известна и равна $O(N^{\frac{3}{2}})$ [2], что тоже соответствует результату в таб.1. Она относительно велика, так как последовательные шаги кратны друг другу, и при их использовании происходит слияние предыдущих подпоследовательностей, а не их перемешивание

Широко известные и часто используемые последовательности шагов приведены в таблице 2.

Таблица 2. Последовательности шагов

Алгоритм	10^4	$2*10^4$	$5*10^4$	10^5	$2*10^5$	$5*10^5$	10^6	$2*10^6$	Коэфф.
$h_k = \frac{N}{2^k}$	14	30	90	200	460	1372	3190	7071	1.18
$h_k = \frac{N}{3^k}$	12	25	67	156	393	1084	2420	5504	1.17
$Fibbo$	16	35	100	230	541	1702	3902	8716	1.20
Sedgewick	10	22	60	131	270	731	1590	3363	1.09

Третий алгоритм в табл.2 – числа Фибоначчи, где очередной шаг равен сумме двух предыдущих, первые два равны 1 и 2.

Ряд нетривиальных последовательностей шагов предложено и изучено в работах [2,3,4]. Самой лучшей, из известных на сегодня, является последовательность шагов, предложенная Седгевиком [4].

$$h_k = \begin{cases} 9\left(2^k - 2^{\frac{k}{2}}\right) + 1, \text{если к} - \text{четное} \\ 8 \cdot 2^k - 6 \cdot 2^{(k+1)/2} + 1, \text{если } k \text{ нечетное} \end{cases}$$

Средняя временная сложность для этой последовательности равна $O(N^{\frac{7}{6}})$ [4]. Результаты приведены в последней строке таблицы 2. На рисунке 2 приведены графики для последовательности 2 (на легенде графика – *Div*3) и последовательности шагов Седгевика.

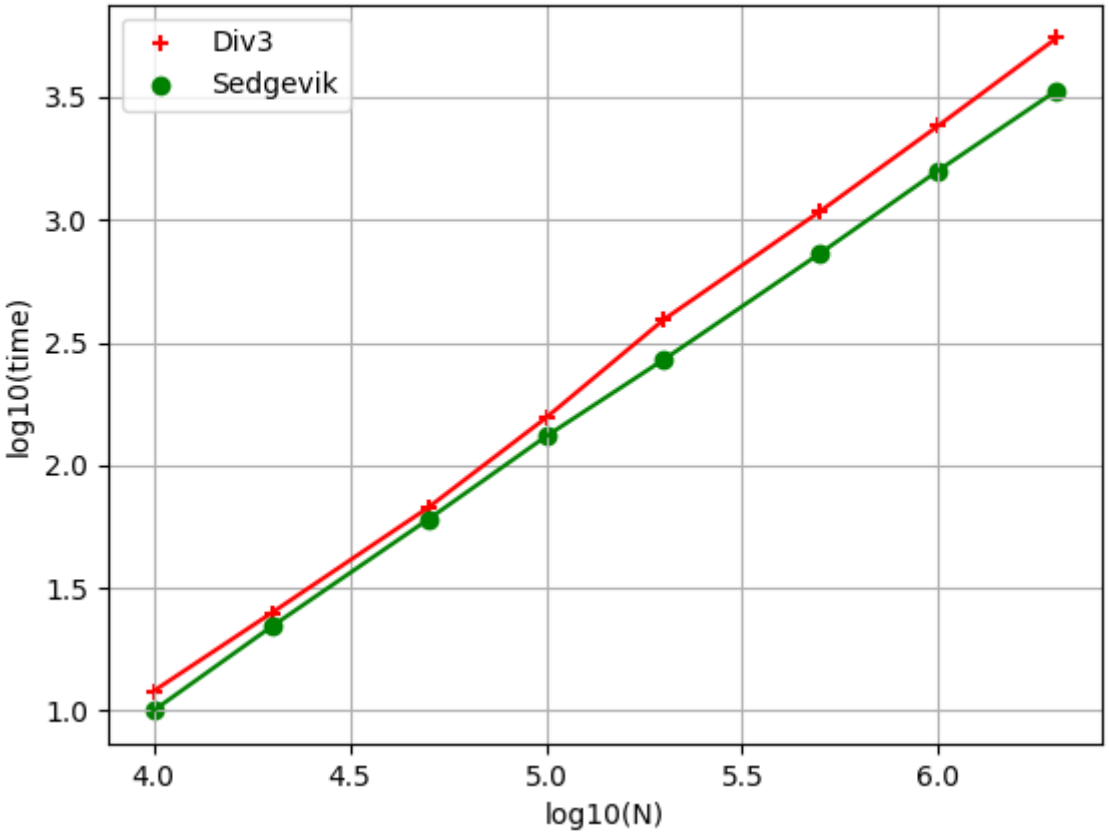

Рис. 2. Графики для последовательности 2

Рассмотрим последовательности, генерируемые формулой $h_{k+1} = \lfloor ah_k \rfloor + 1$, где a – некоторая действительная константа. Через $\lfloor x \rfloor$ обозначена операция извлечения целой части числа. В таблице 3 приведены результаты прогонов для некоторых значений параметра a.

Таблица 3. Результаты прогонов

Параметр	10^4	$2*10^4$	$5*10^4$	10^5	$2*10^5$	$5*10^5$	10^6	$2*10^6$	Коэфф.
1.9	11	24	66	144	297	802	1738	3593	1.09
2.0	12	28	85	185	432	1267	2690	7192	1.17
2.1	10	22	61	132	283	754	1547	3356	1.09
2.3	9	20	59	127	282	743	1633	3387	1.09
2.5	10	22	60	130	271	736	1619	3421	1.10

Хорошо видно, что за исключением целочисленного значения a = 2, результаты вполне сравнимы с последовательностью Седгевика. График не приводится, так как эти линии почти сливаются.

Другая интересная последовательность шагов генерируется формулой $h_{k+1} = \lfloor ah_k \rfloor$. где a – некоторая действительная константа. В таблице 4 приведены результаты для этой последовательности при разных значениях параметра a.

Таблица 4. Результаты

Параметр	10^4	$2*10^4$	$5*10^4$	10^5	$2*10^5$	$5*10^5$	10^6	$2*10^6$	Коэфф.
2.1	11	24	65	138	287	784	1641	3515	1.09
2.3	10	22	63	134	289	771	1690	3511	1.09
2.5	10	23	62	135	283	787	1705	3638	1.11
2.7	11	22	61	127	274	755	1652	3447	1.09

И последняя серия шагов, порождаемая формулой $h_{k+1} = \frac{h_k \cdot p}{q},\ h_1 = \frac{N}{2}$, где p, q – некоторые целые константы.

Таблица 5. Результаты

Параметры	10^4	$2 * 10^4$	$5 * 10^4$	10^5	$2 * 10^5$	$5 * 10^5$	10^6	$2 * 10^6$	Коэфф.
p=13 q=31	10	22	60	132	282	750	1622	3544	1.09
p=15 q=31	11	23	62	135	286	754	1610	3418	1.09
p=17 q=31	11	24	68	138	297	785	1680	3551	1.09
p=19 q=31	12	25	68	146	302	847	1775	3752	1.10

Видно, что эти серии показывают очень неплохие результаты в рассмотренных условиях.

Результаты

Рассмотренные семейства шагов сортировки для алгоритма сортировки Шелла показывают хорошие результаты в практических тестах. Сами формулы для этих серий гораздо проще, чем, например, серия Седгевика. Однако, нужно отметить, что эти серии теоретически не изучены, и средняя сложность для них получена только эмпирически.

Список источников

1. Shell D. L. A high speed sorting procedure. Communications of the ACM. 1956. Т. 2. № 7. С. 30–32. DOI: https://doi.org/10.1145/368370.368387.
2. Кнут, Д. Искусство программирования. Т 3: Сортировка и поиск. 3-е издание. Москва: Вильямс, 2017.
3. Plaxton C. Greg, Suel Torsten. Lower Bounds for Shellsort. Journal of Algorithms. 1997. Т. 23. № 2. С. 221–240. DOI: https://doi.org/10.1006/jagm.1996.0825.
4. Sedgewick R. A New Upper Bound for Shellsort. Journal of Algorithms. 1986. Т. 7. № 2. С. 159–173.