Advance Operating Systems - HW1

Page Replacement Algorithms and Evaluation

Student Name: 符新祐

Student ID: M133040019

Table of Contents

(1) Introduction

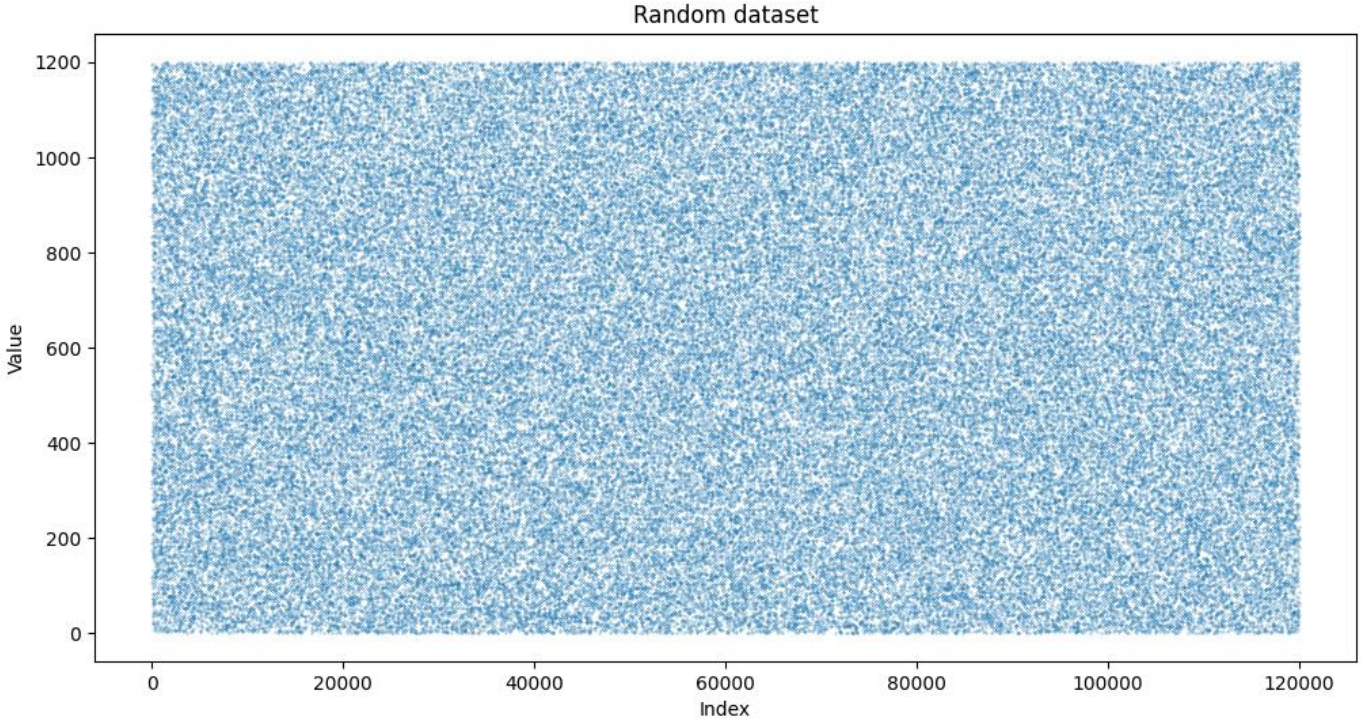
HW1 requires us to implement page replacement algorithms taught in class. In this report, I will demonstrate the performance of different algorithm (including FIFO, optimal, enhanced second-chance, and a self-purposed algorithm farthest neighbor) under different reference string dataset (including random, locality, and a customized dataset gaussian-distributed locality), as well as analyze each of their specific features.

(2) Reference string dataset

The reference string indicates the memory references under process demand. We simulated the reference string by generating 120000 integers range from 1-1200. This part briefly explains the generating method and illustrate the dataset.

(2.1) Random

Random dataset is generated by uniformly choosing 120000 integers from [1,1200]:



Graph 2.1. x-axis indicates the index, y-axis indicates the data value. We can see that the data point is scattered uniformly throughout the plot

(2.2) Locality

Locality dataset is generated by:

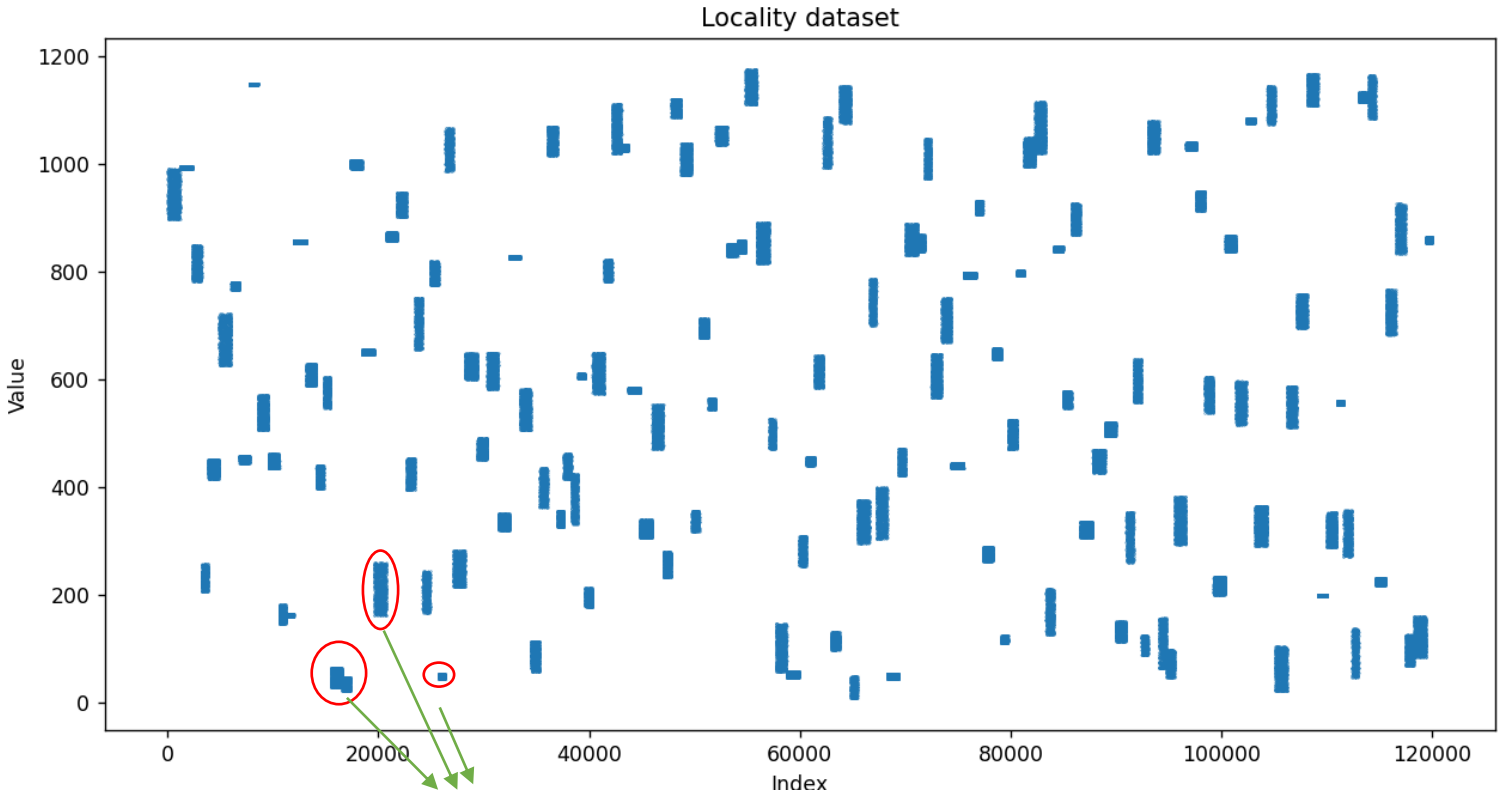
(1) Choose the locality segment length from 1/200~1/100

of the total length 120000

(2) Choose the locality value range from [5, 100]

(3) Move the locality to a random value position (y-axis)

Result:



Graph 2.2. We can see localized reference segments that vary in size (value from [5, 100] and index from [600, 1200]) with uniformly scattered positioning. Each segment represents a procedure call.

(3) Self-designed content

(3.1) Self-designed algorithm – Farthest neighbor algorithm

As locality mentioned in page replacement scheme, I observe that the "distance" between each memory reference may be a clue to locality. As graph 2.2 shown, reference in locality segment has very close distance to each other. Also, figure described in text book [1] shows a utilization of close pages in the locality spot. This make sense that when a page fault happens, choosing a victim that is farthest from the "hot" pages might be a good choice.

Farthest neighbor algorithm:

r = input reference

f = main memory frames

v = victim frame

```
if (page_fault) then
    for i in frames do
        frames_dist[i] = abs(frames[i] – reference)
    end for
    v = argmax(frames_dist)
end if
```

Explanation: when page fault happens, the algorithm calculates the distance between it and all the value inside main memory frames. The victim page is then selected by the frame with the greatest distance. The algorithm is named “farthest neighbor” in purpose to show contrast to k-nearest neighbor(knn) algorithm [2] which aims to find a “most-related” spot for an incoming data, in FN algorithm, we try to find a “most-unrelated” spot.

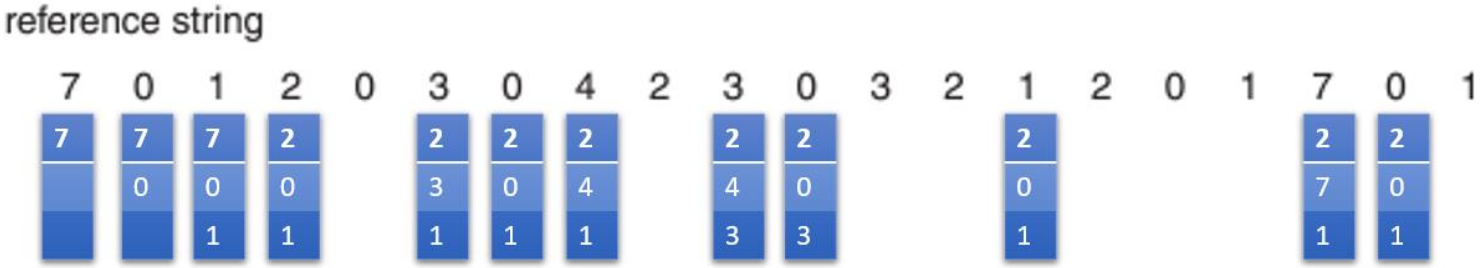


Figure 3.1 examples of FN algorithm with same reference string mentioned in [1]’s page replacement chapter

(3.2) Self-designed dataset – Gaussian-distributed locality with noise

Gaussian/normal distribution is a type of probability distribution with probability density function:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

One of the features of distance-based algorithm is that it “sense” the “density” of data. A larger total distance, as definition of standard deviation described, indicates a lower density of a data:

$$\sigma \equiv \sqrt{\mathrm{E}\left[(X-\mu)^2\right]} = \sqrt{\int_{-\infty}^{+\infty} (x-\mu)^2 f(x)\,\mathrm{d}x}$$

FN algorithm, which simply calculates absolute value distance, tends to

choose a point far away from the center. Applying a dataset that is more concentrated in the center supposedly benefits to FN algorithm's performance. Furthermore, noise is added to the dataset since FN will choose a victim close to one of the edges of locality segment.

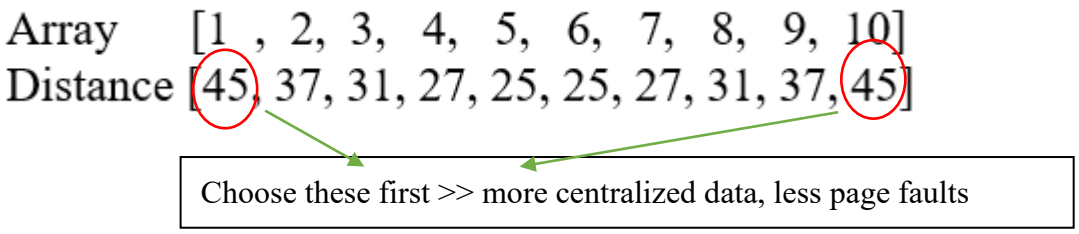


Fig 3.2. The distance to all neighbors is calculated by each point. We can find that the distance will be greater in the edge, so applying data with more centralized reference string may be a more reasonable approach

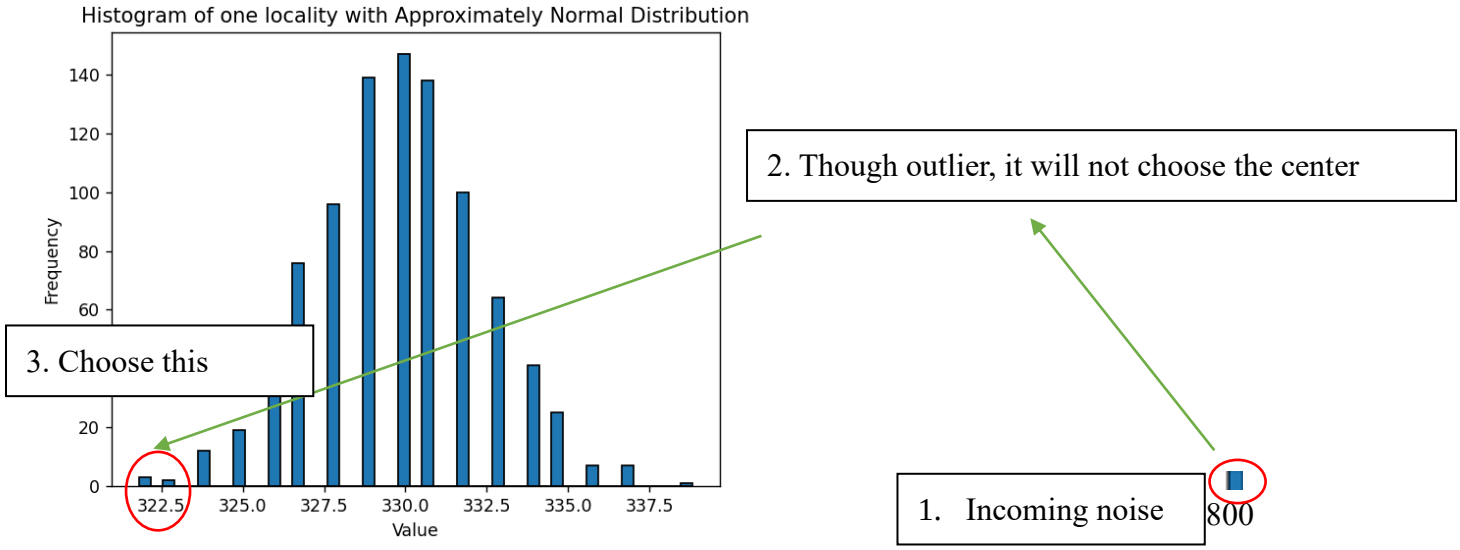


Fig 3.3. The noise redundancy of FN is shown in the figure. No matter what noise incoming, it will choose points far away from center hot pages.

GLN dataset is generated by:

(1) Choose the locality range and length as previous case.
(2) Generate Gaussian distribution in the range. Note that the result is a probability density function
(3) Clip and round the continuous floating point to the desired range
(4) Apply noise

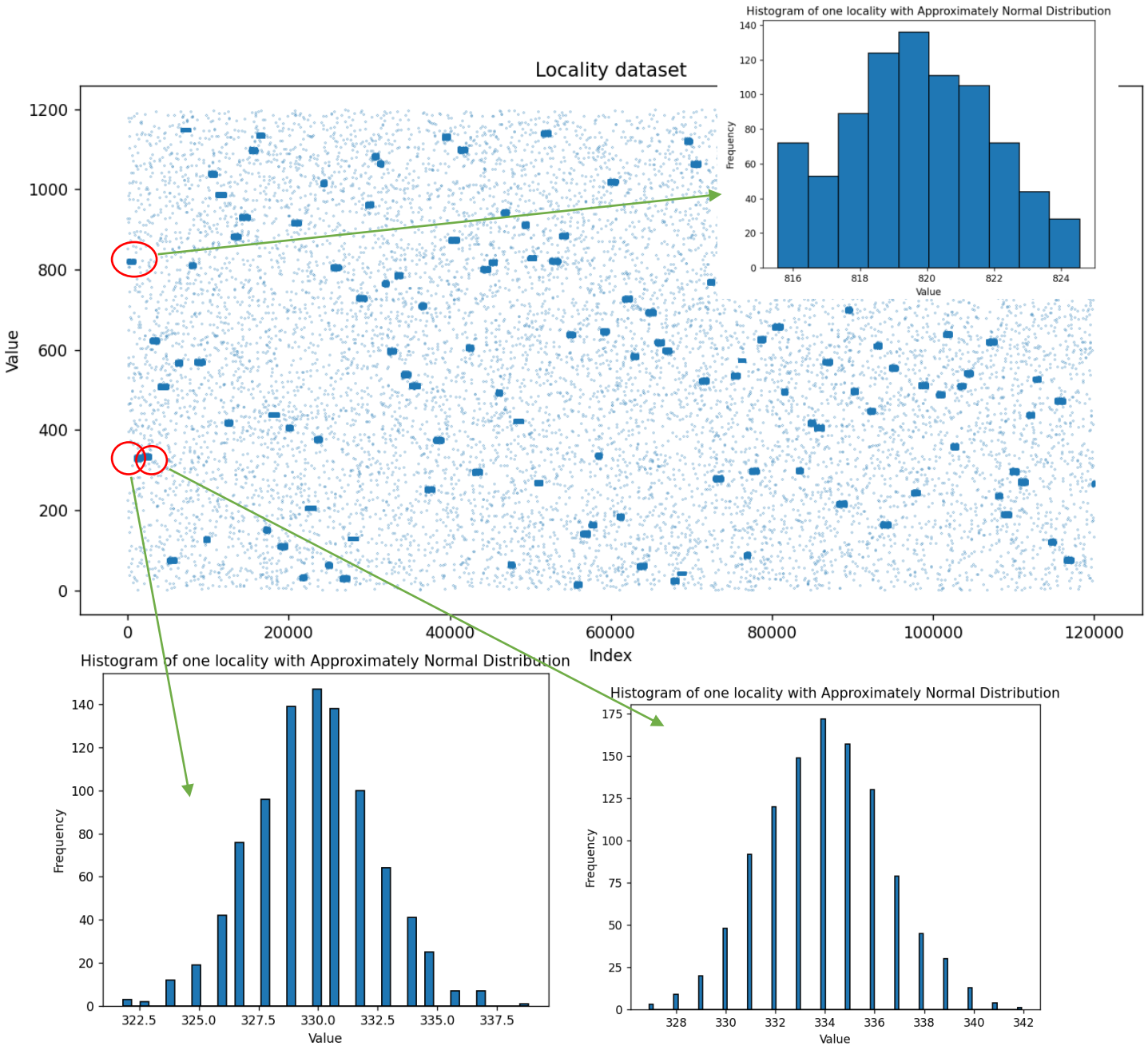


Fig 3.4. The GLN dataset. We can see locality as well as noise scattered in the plot. Histogram of the first three locality segments is also shown following normal distribution.

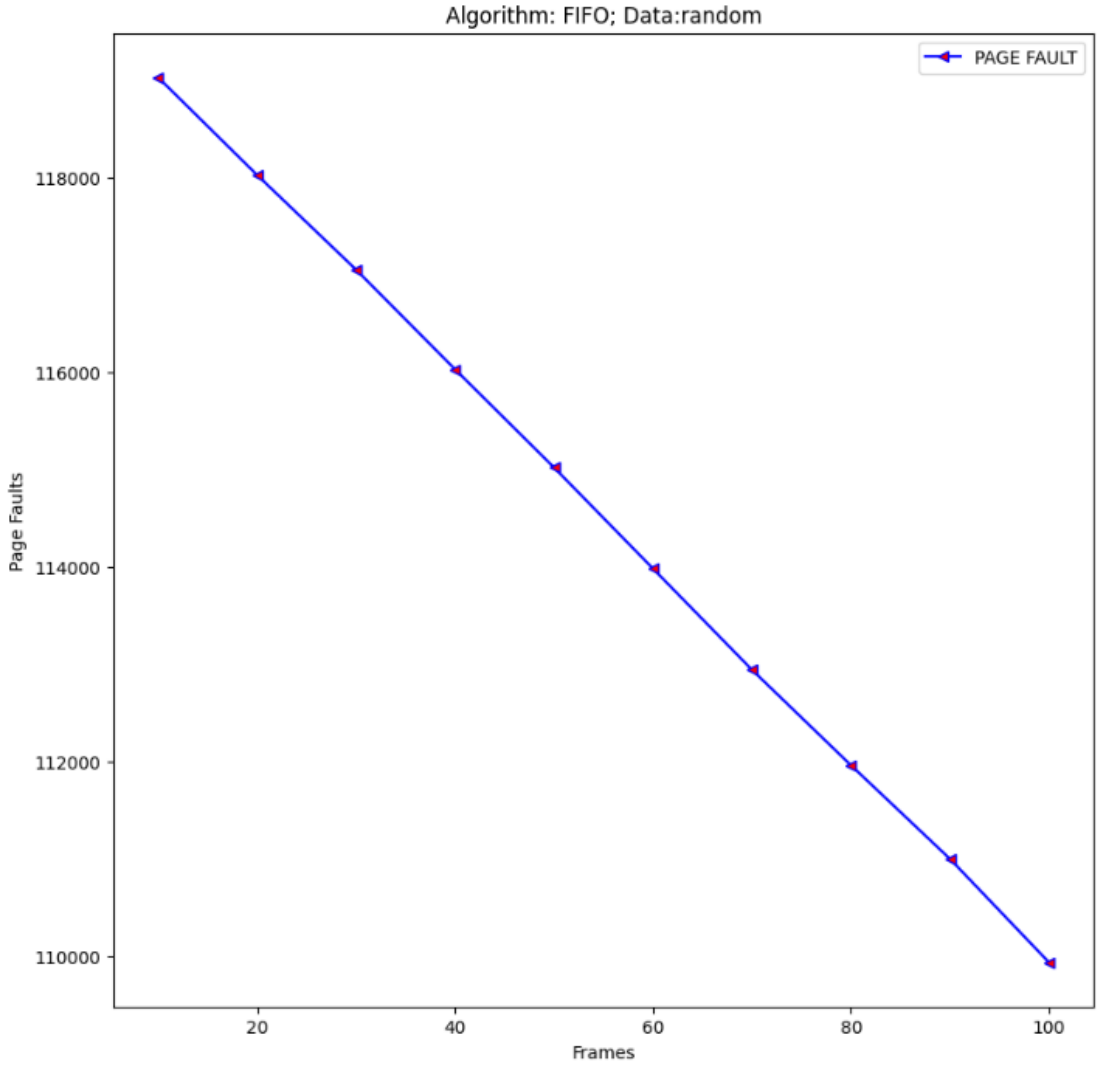
Remarks: Applying the dataset not only provides advantage to FN algorithm. The **unbiased** and **noise** features make a better simulation of real-world scheme.

(4) Simulation Results

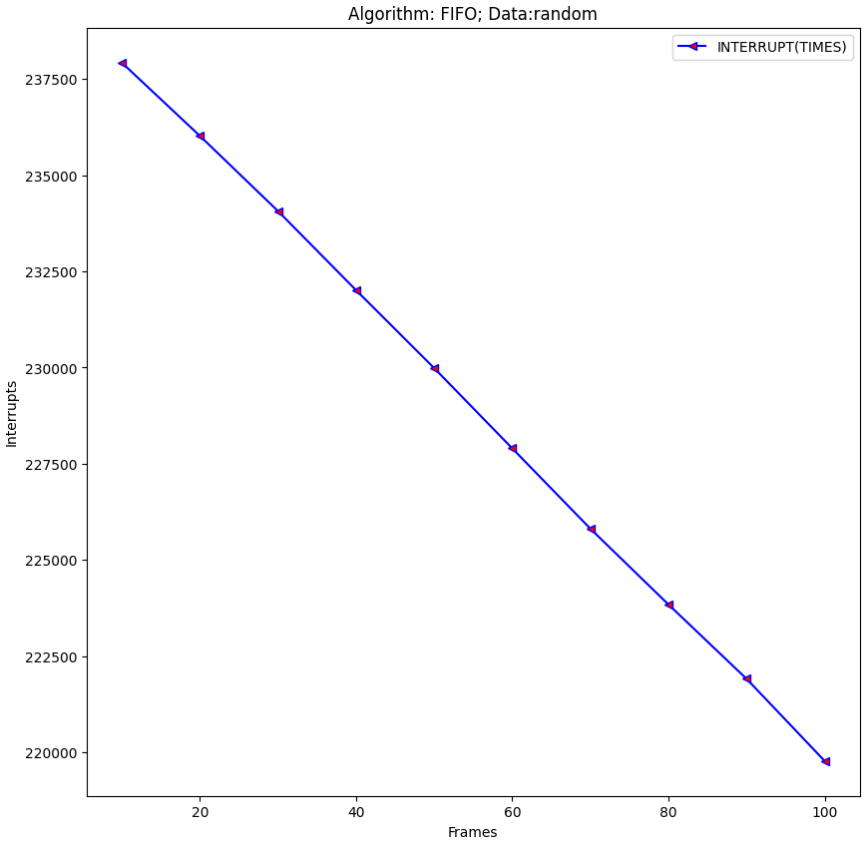
(4.1) Dataset: Random

4.1.1 FIFO algorithm

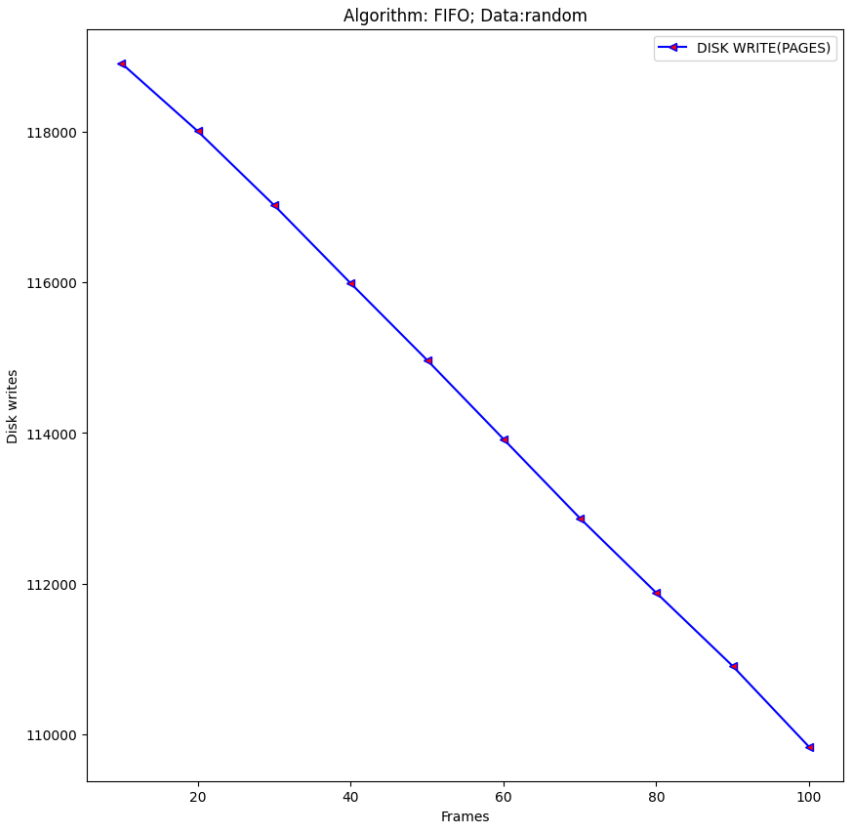
Page faults:



Interrupts:

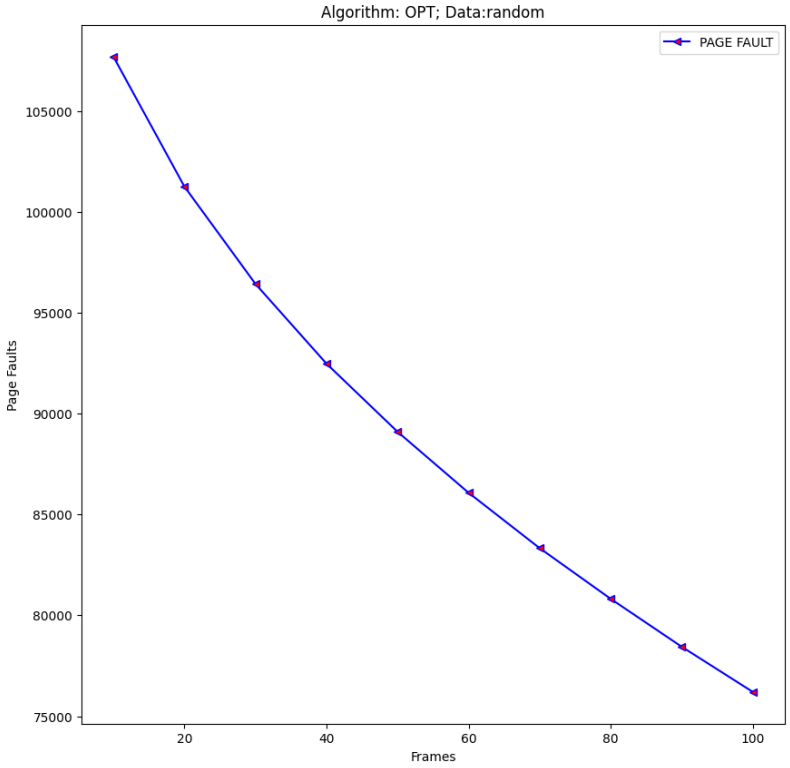


Disk writes:

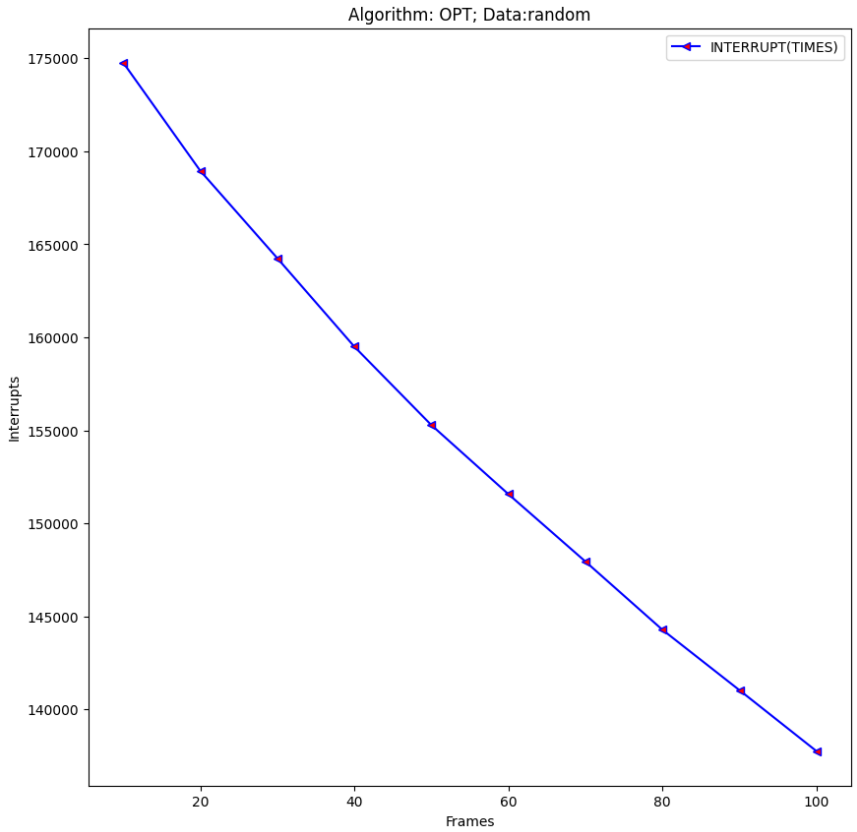


4.1.2 Optimal algorithm

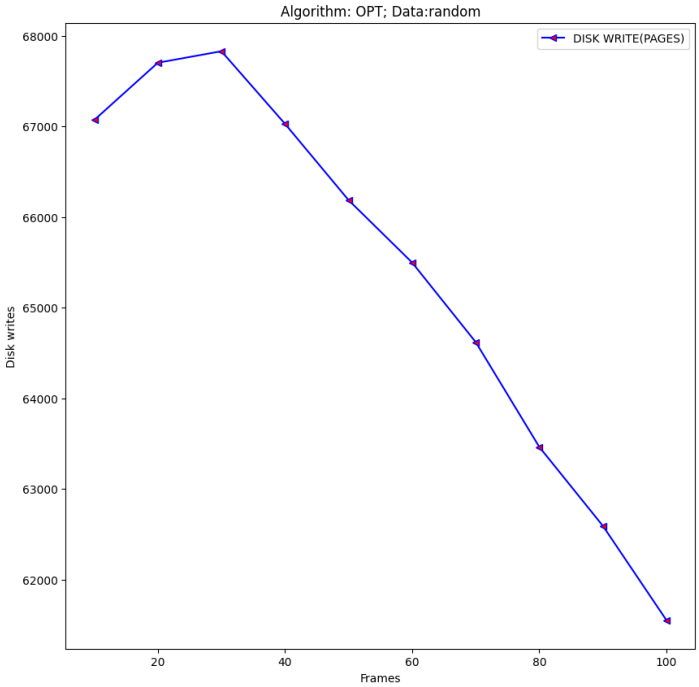
Page faults:



Interrupts:

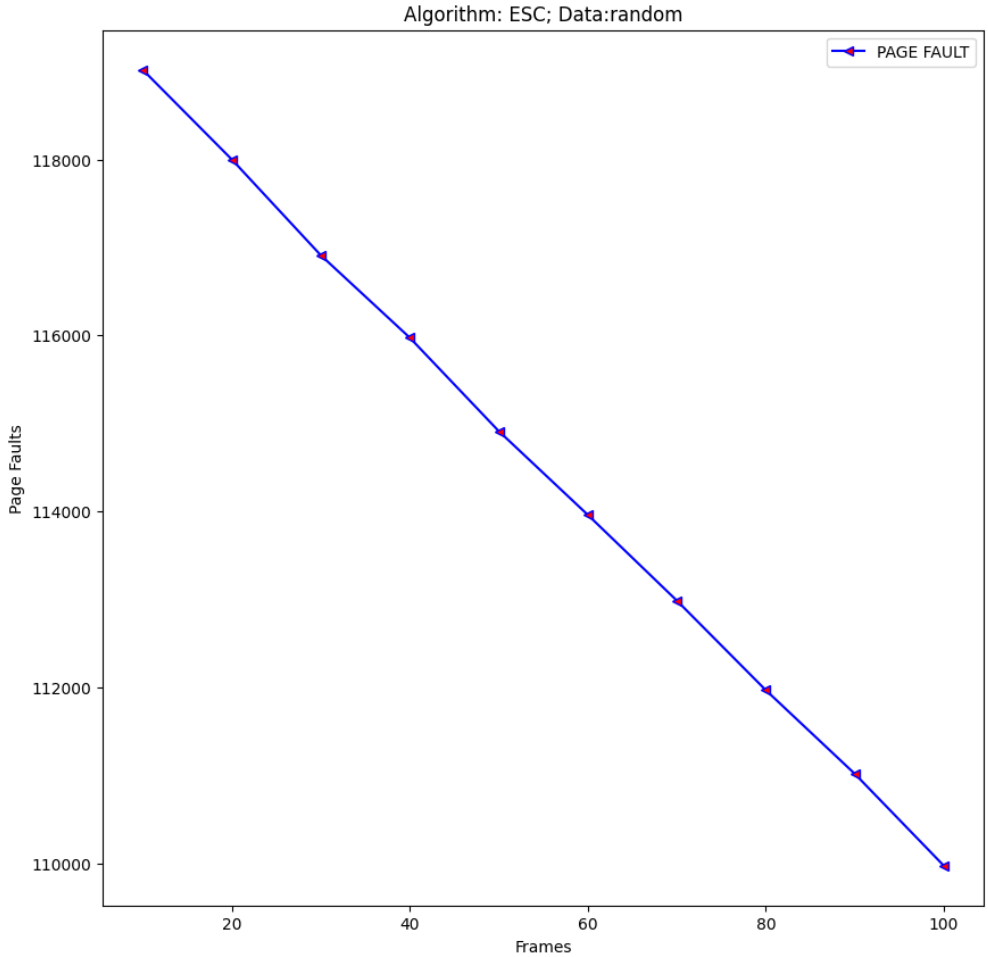


Disk writes:

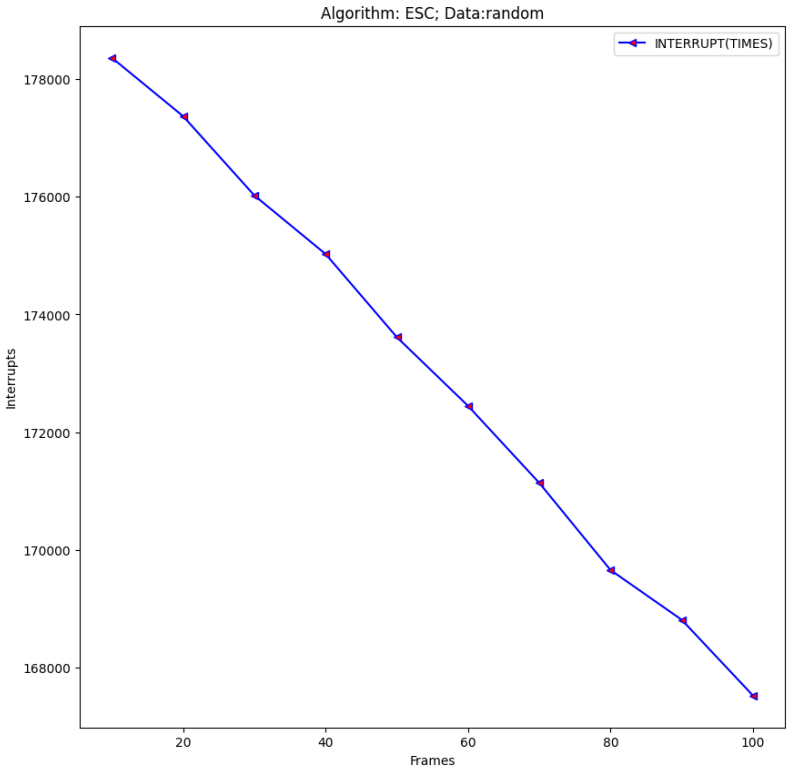


4.1.3 Enhanced second-chance algorithm

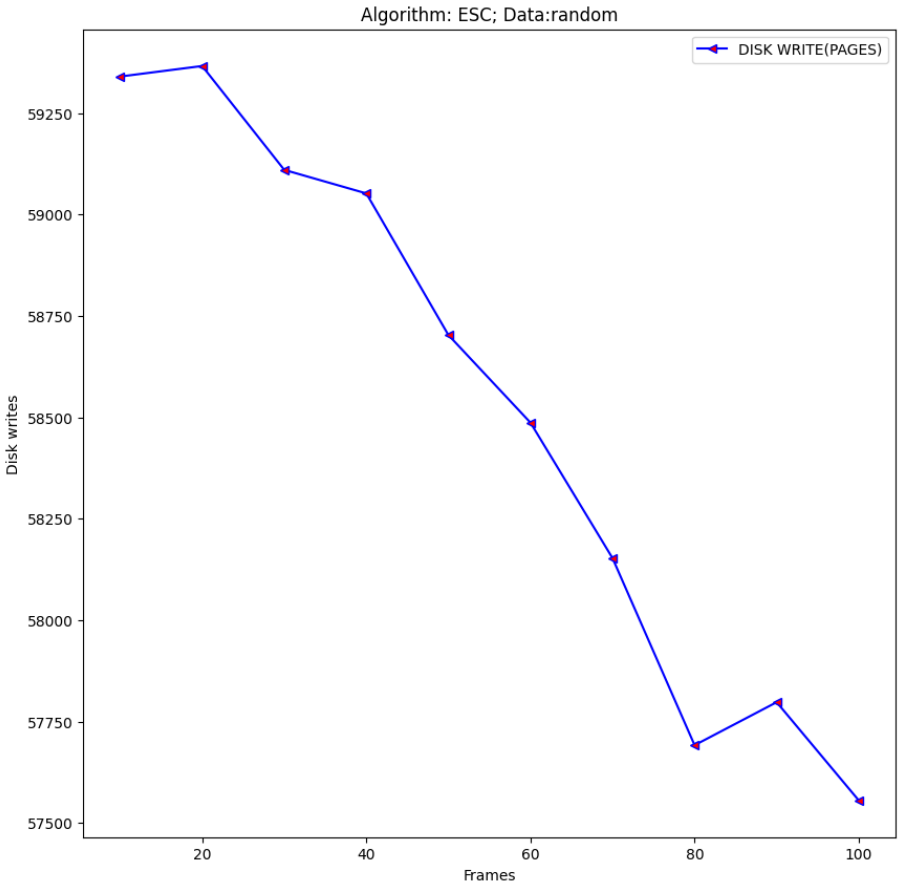
Page faults:



Interrupts:

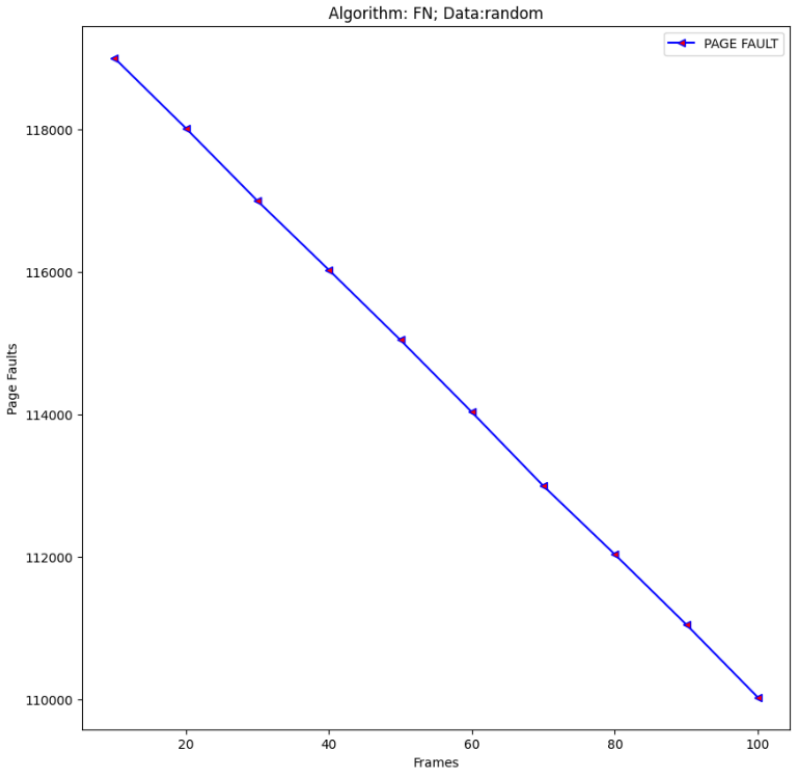


Disk writes:

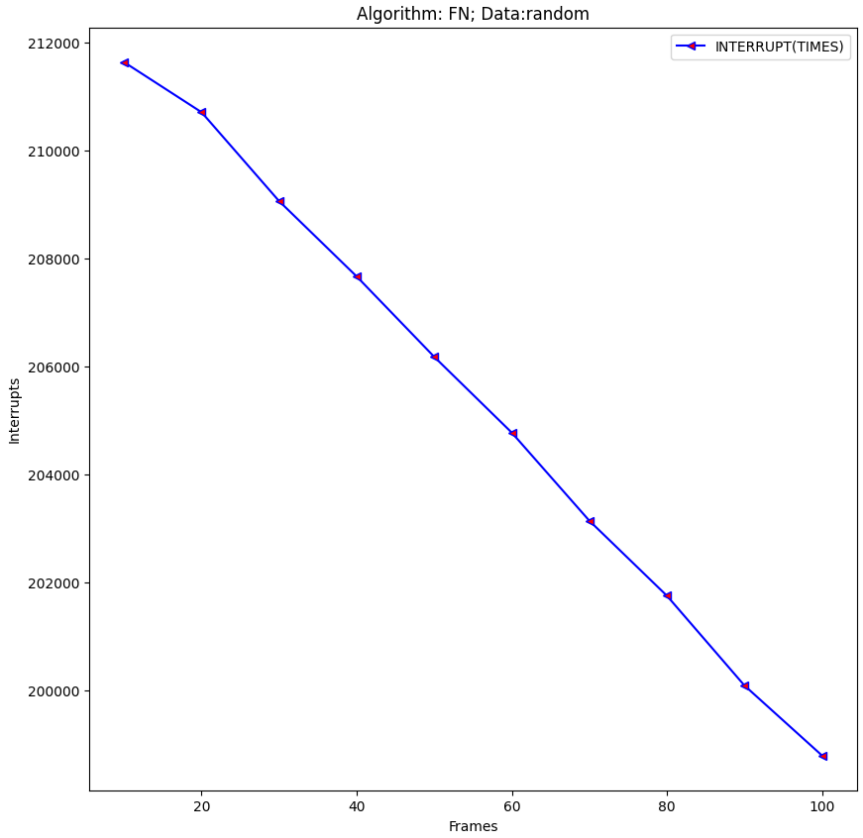


4.1.4 Farthest neighbor algorithm

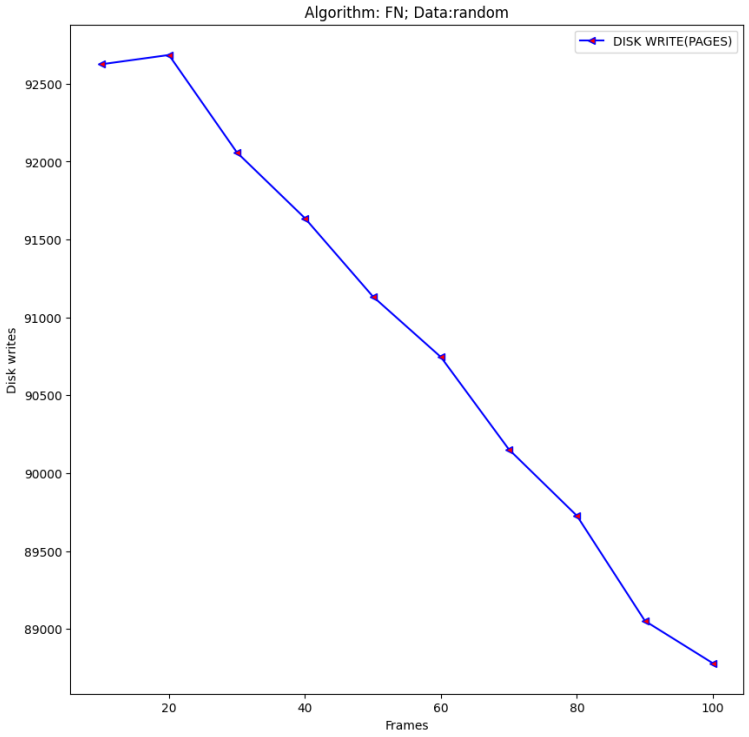
Page faults:



Interrupts:



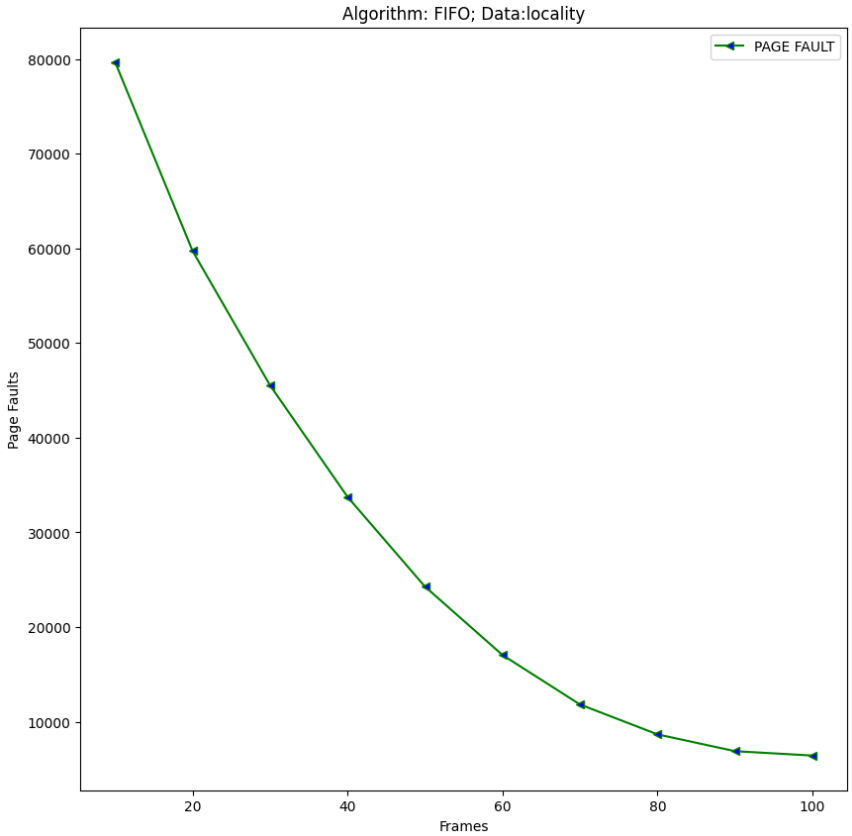
Disk writes:



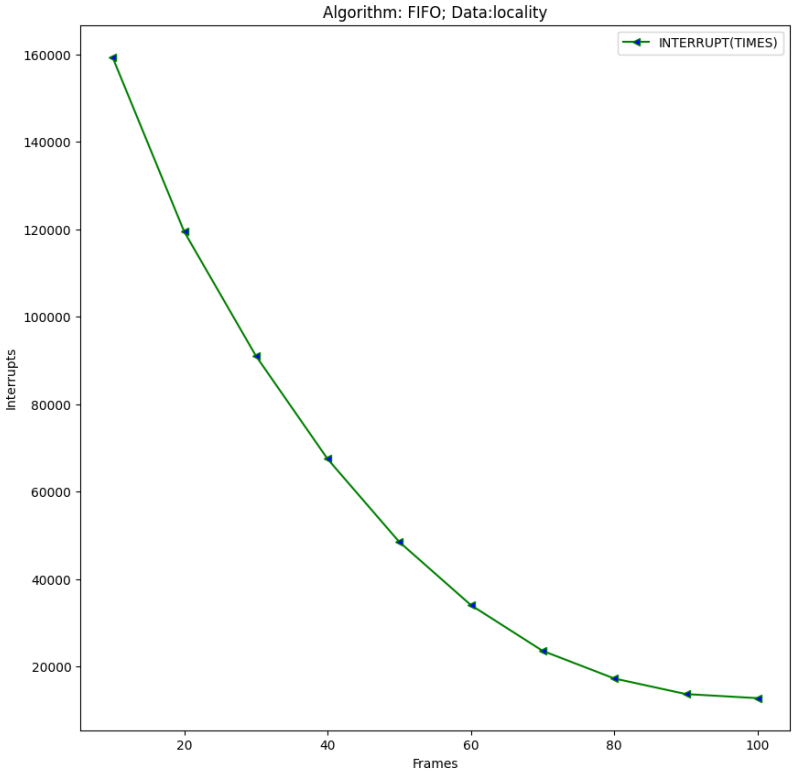
(4.2) Dataset: Locality

4.2.1 FIFO algorithm

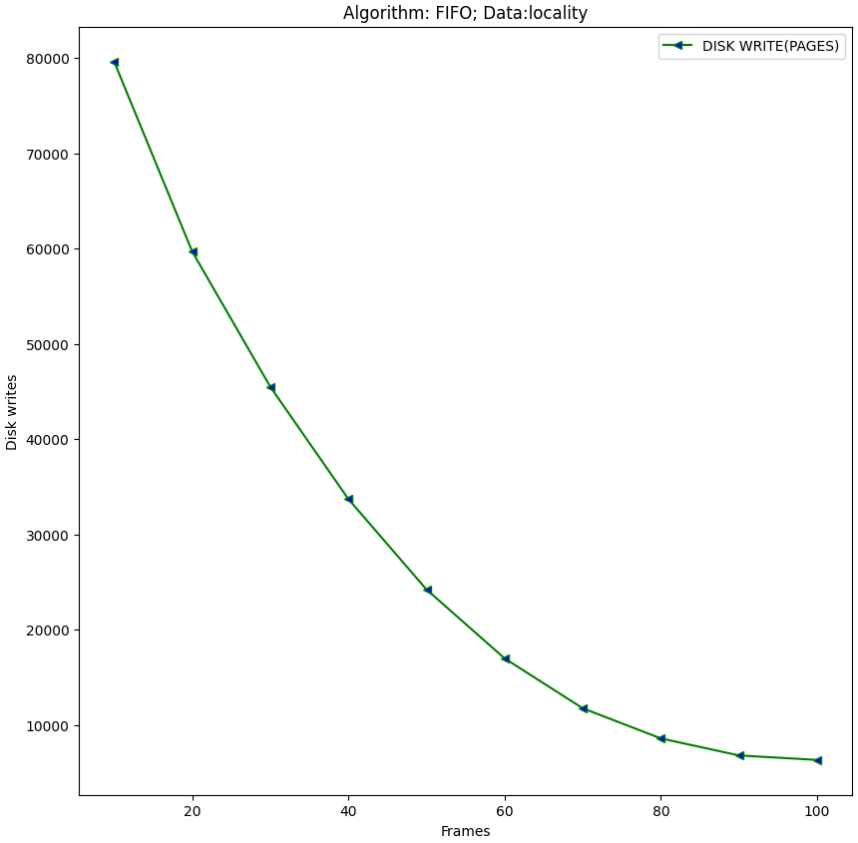
Page faults:



Interrupts:

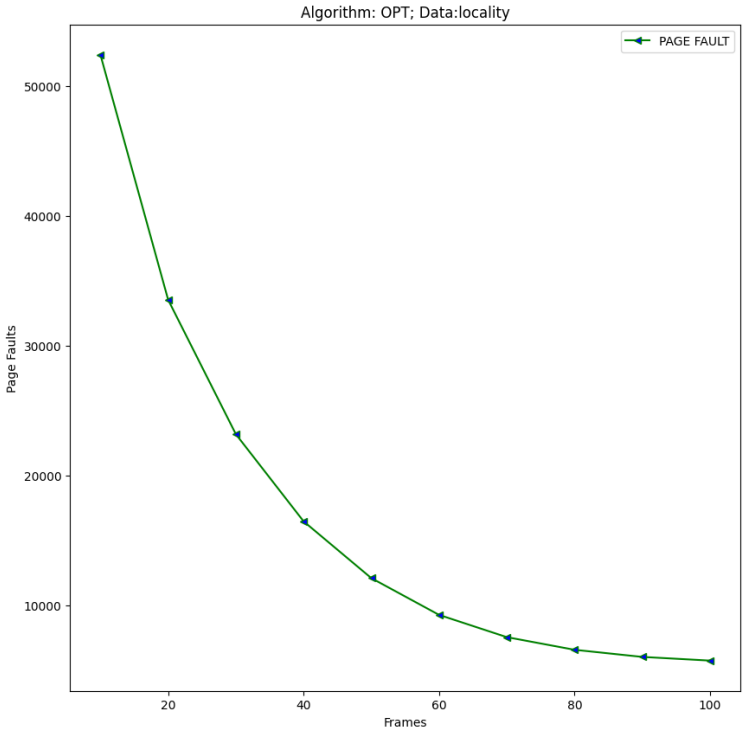


Disk writes:

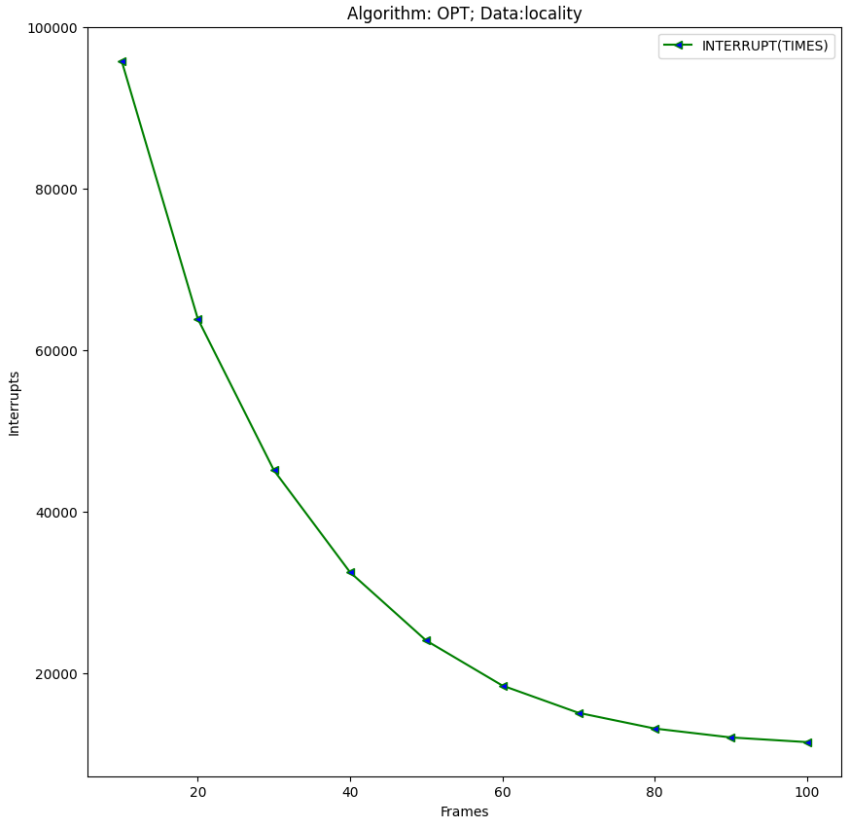


4.2.2 Optimal algorithm

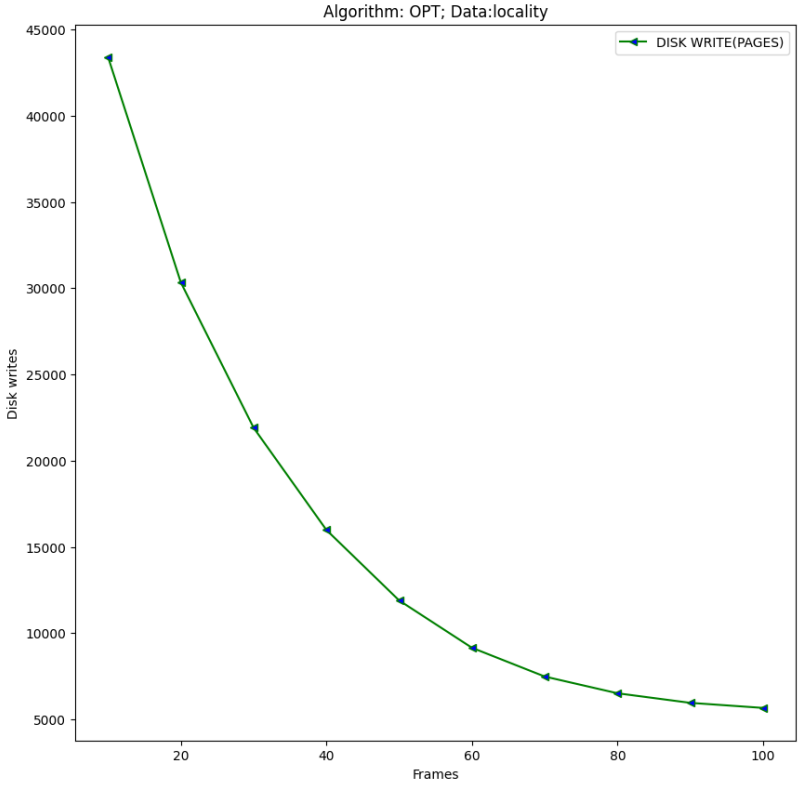
Page faults:



Interrupts:

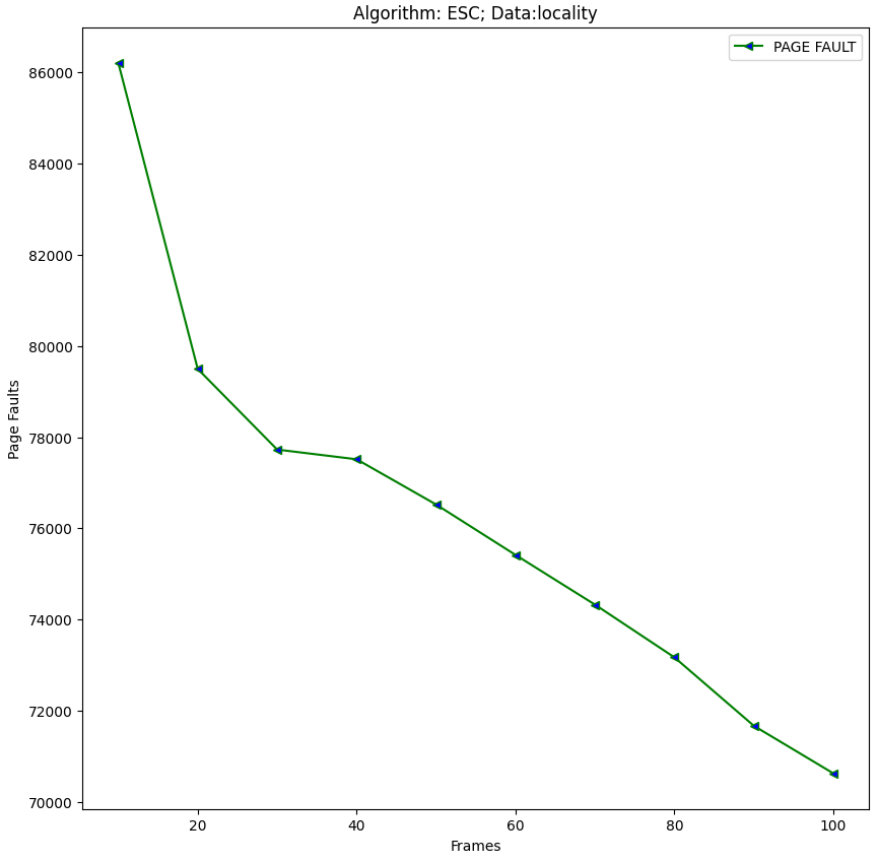


Disk writes:

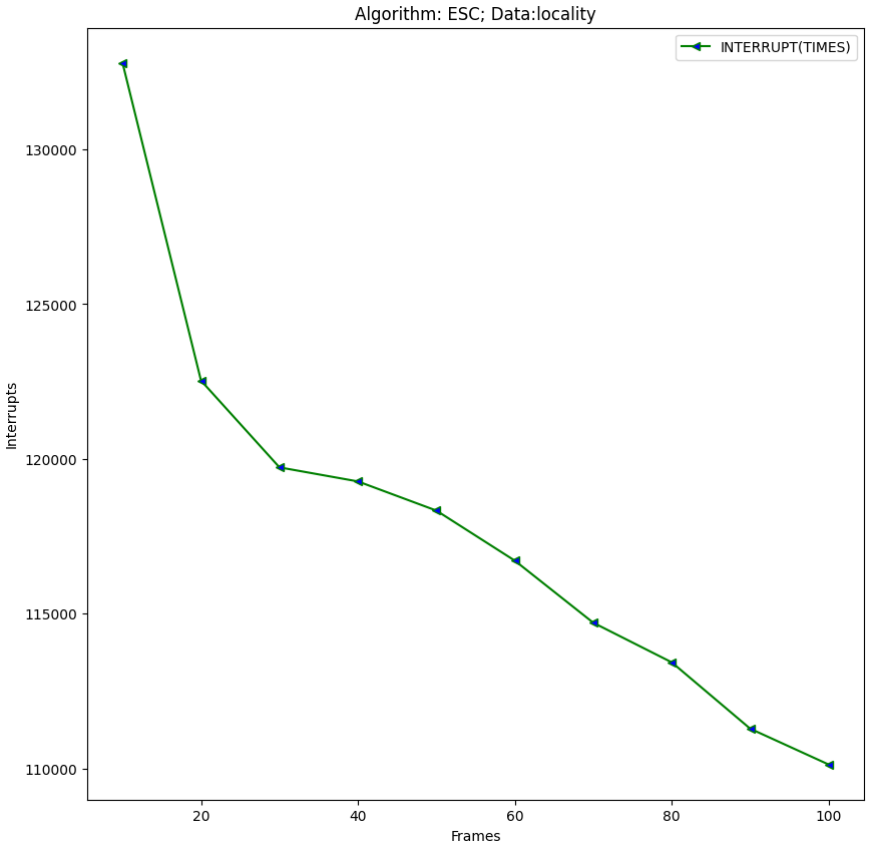


4.2.3 Enhanced second-chance algorithm

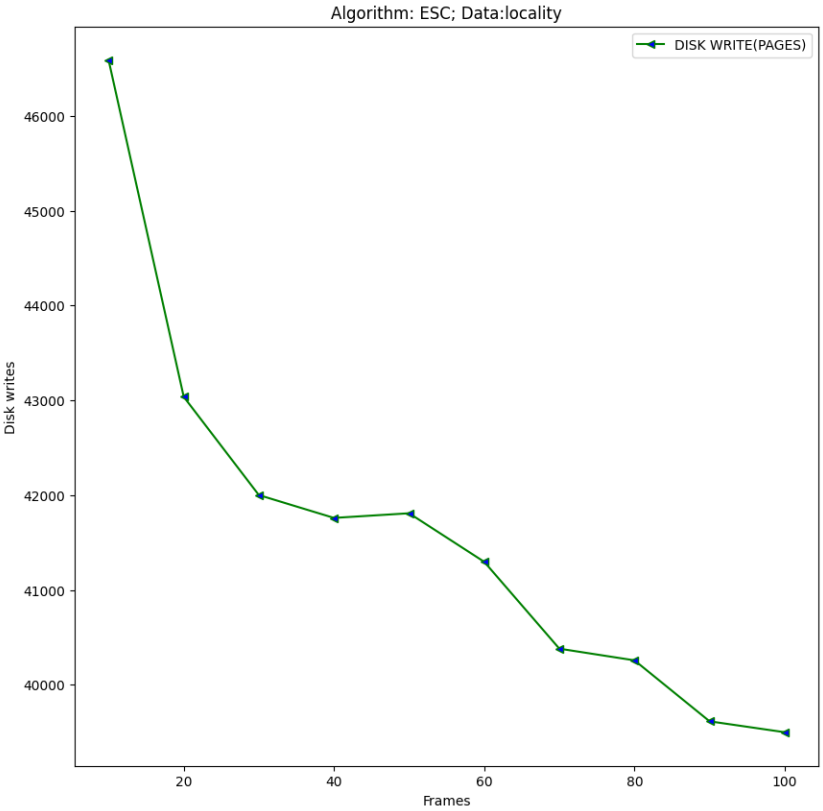
Page faults:



Interrupts:

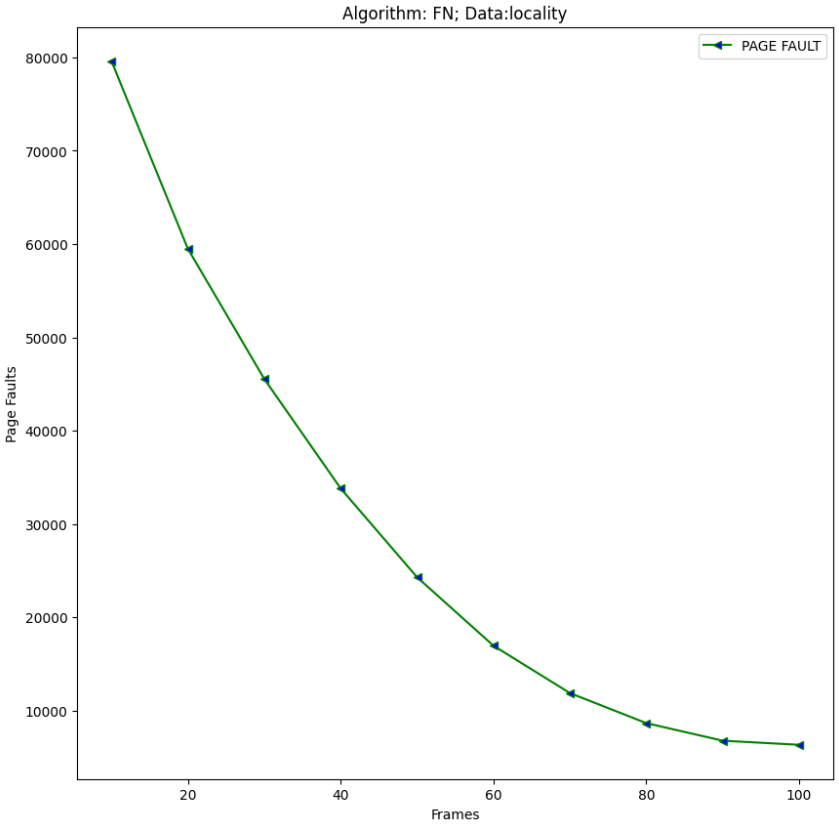


Disk writes:

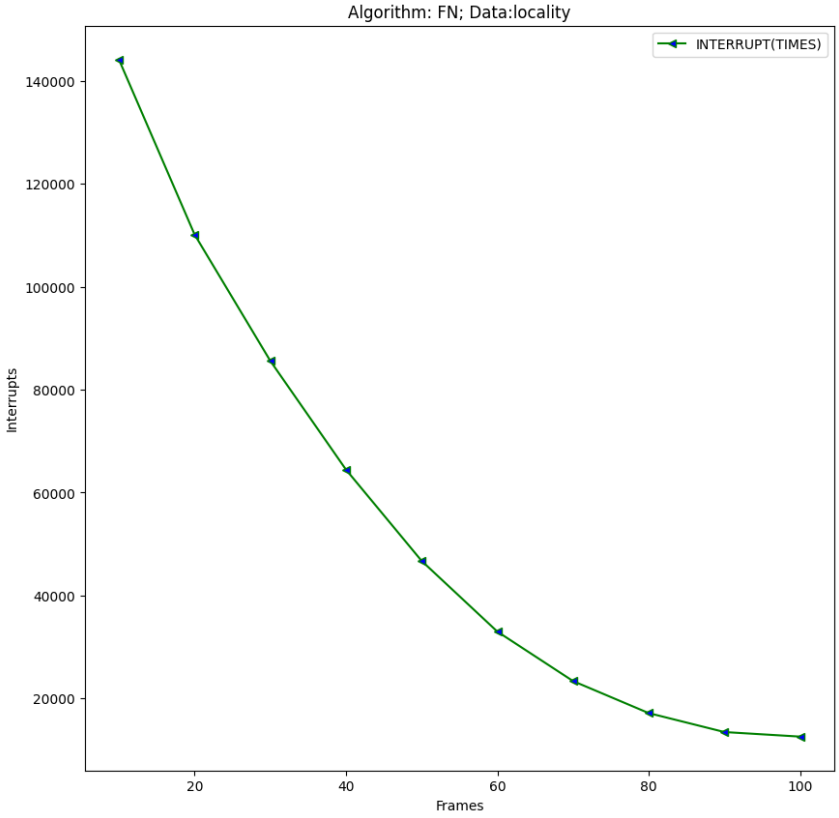


4.2.4 Farthest neighbor algorithm

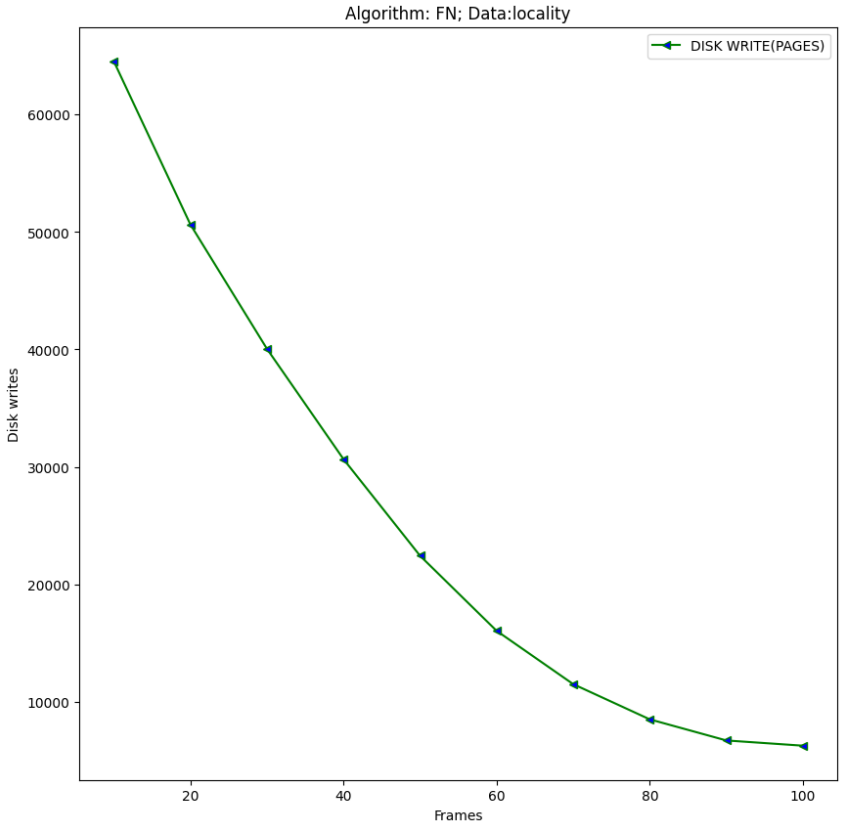
Page faults:



Interrupts:



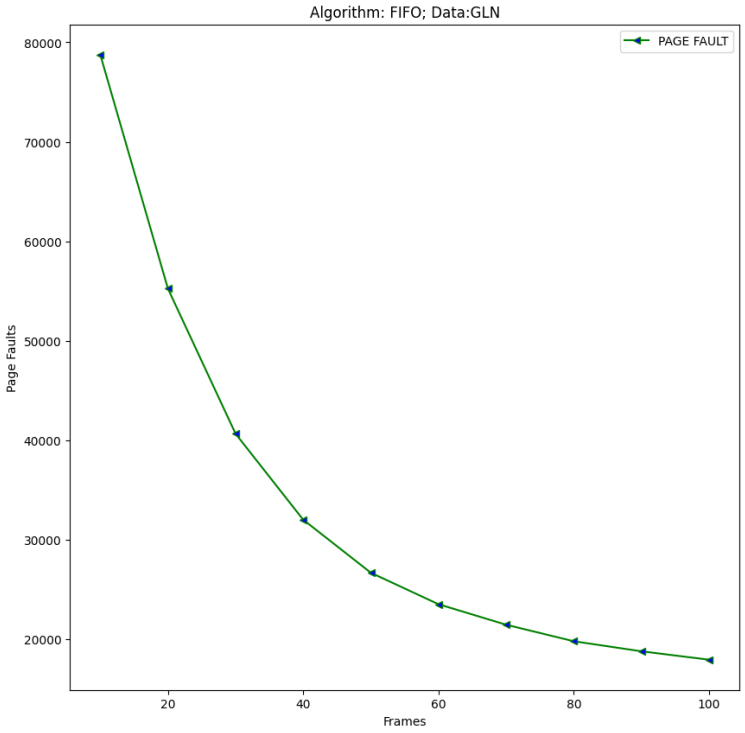
Disk writes:



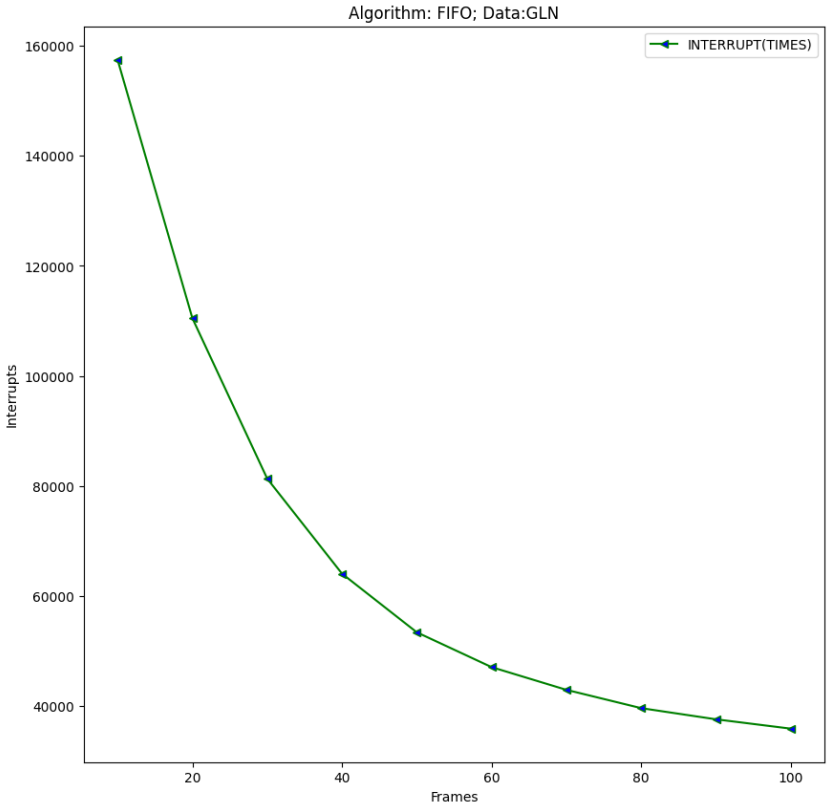
(4.3) Dataset: Gaussian-distributed locality with noise

4.3.1 FIFO algorithm

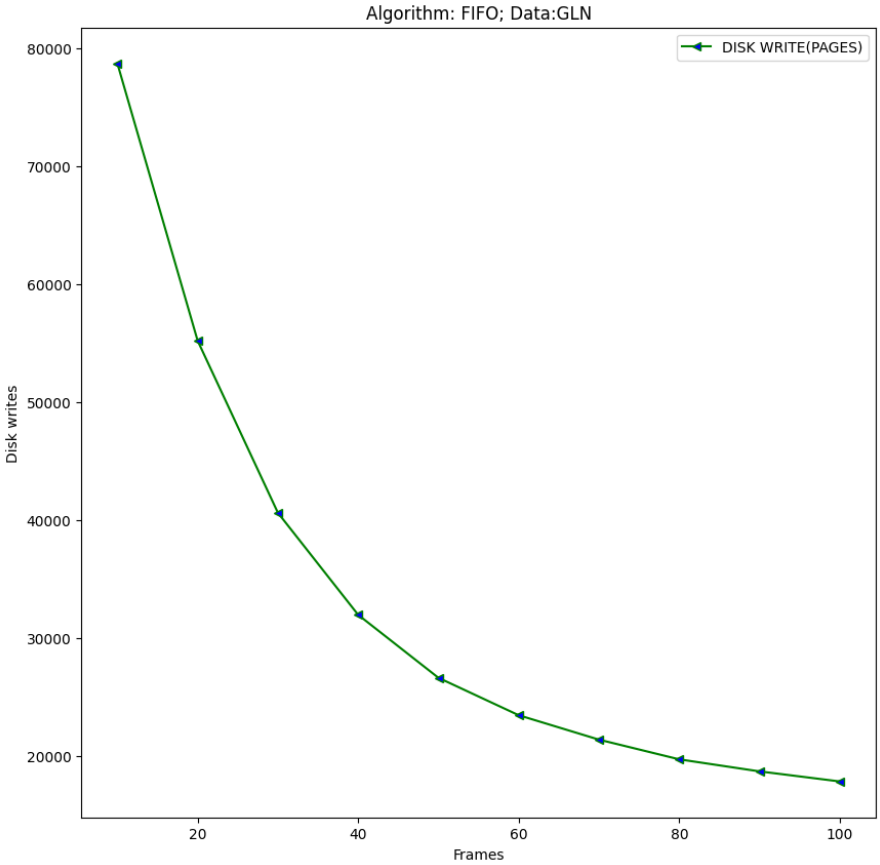
Page faults:



Interrupts:

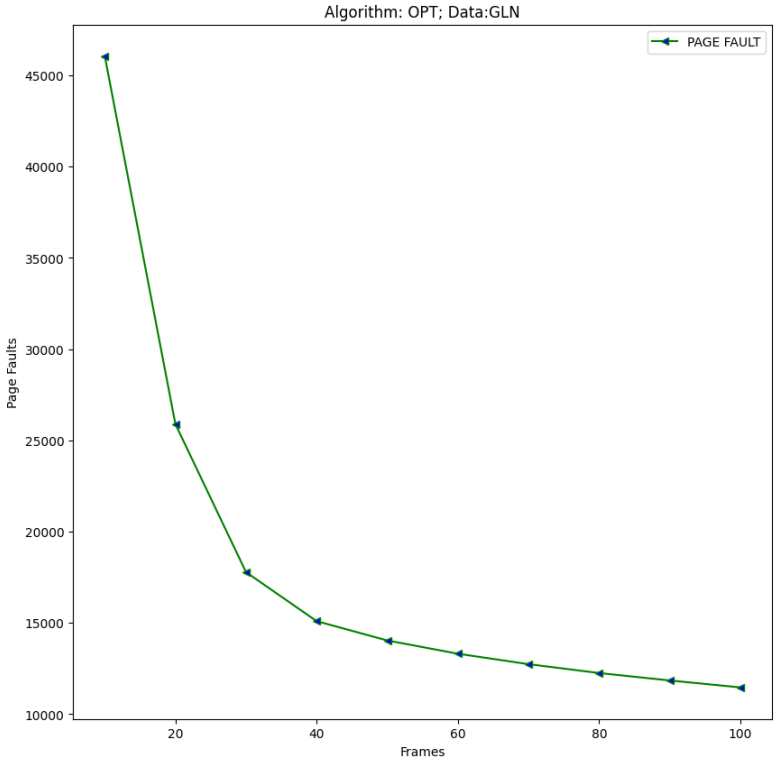


Disk writes:

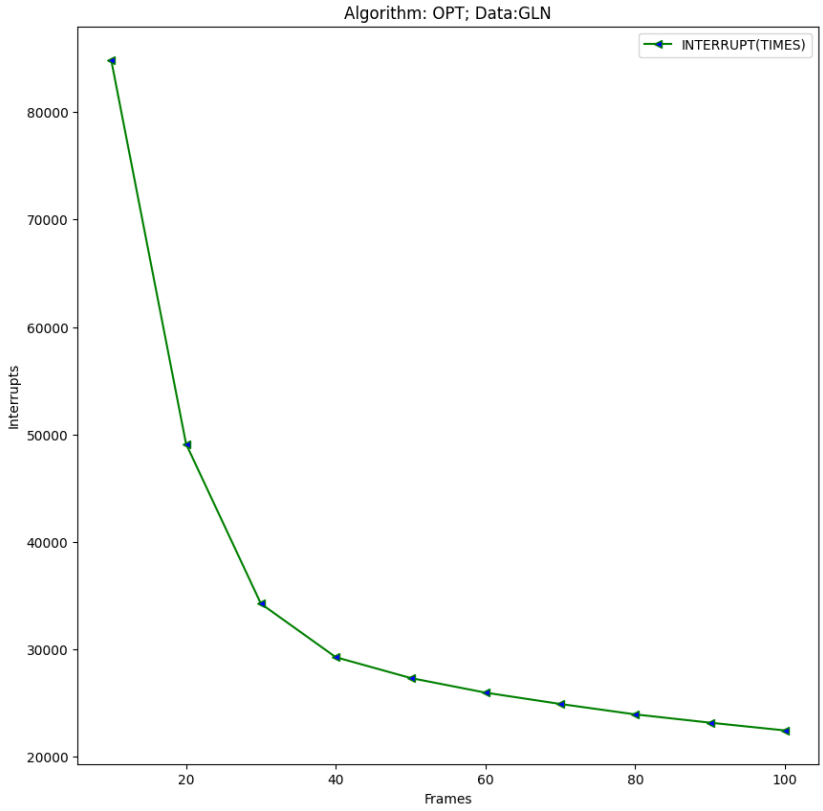


4.3.2 Optimal algorithm

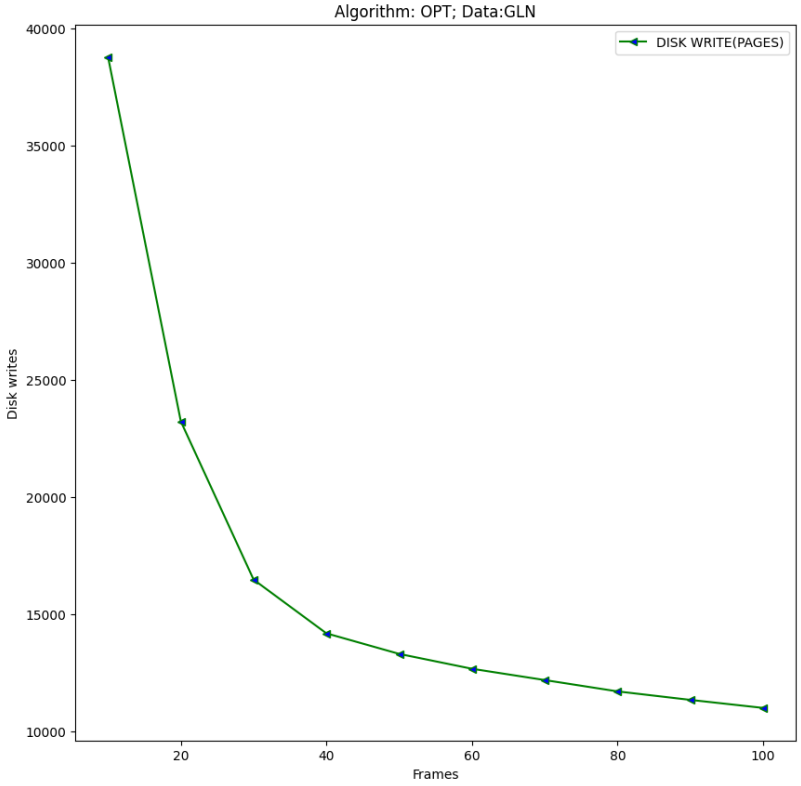
Page faults:



Interrupts:

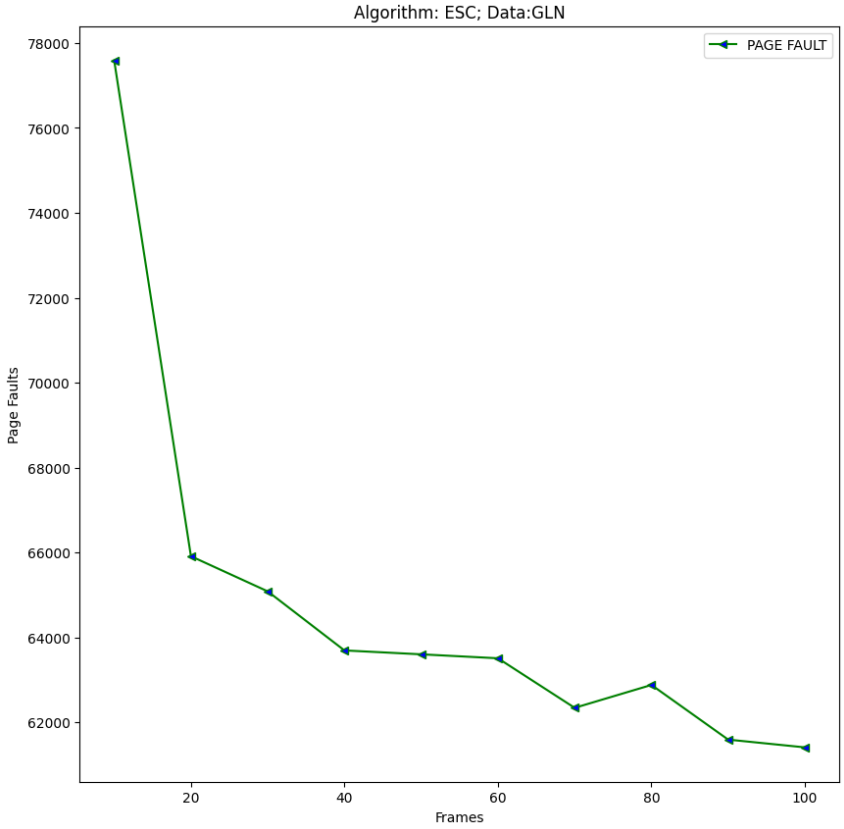


Disk writes:

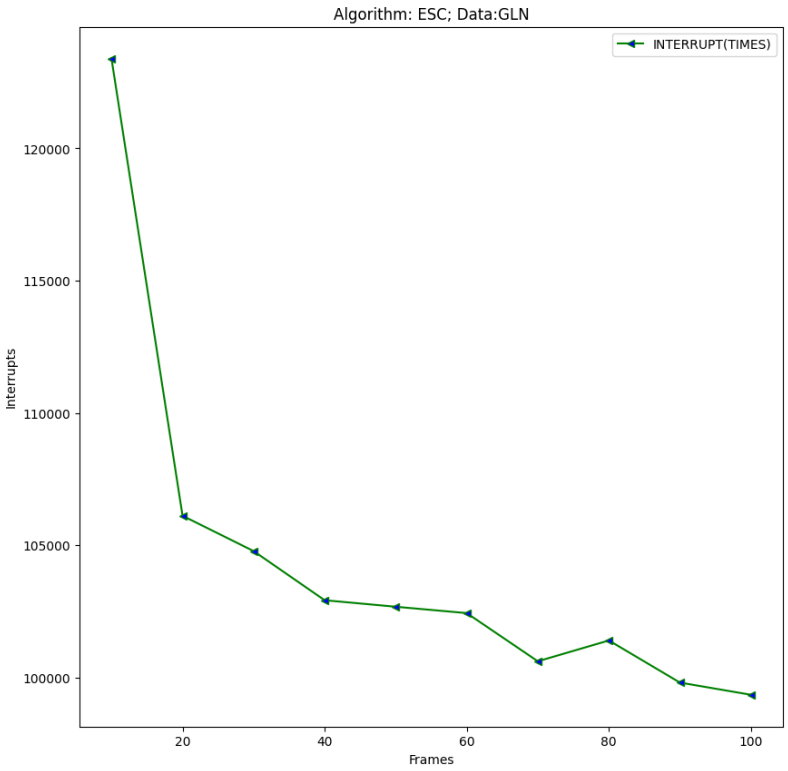


4.3.3 Enhance second-chance algorithm

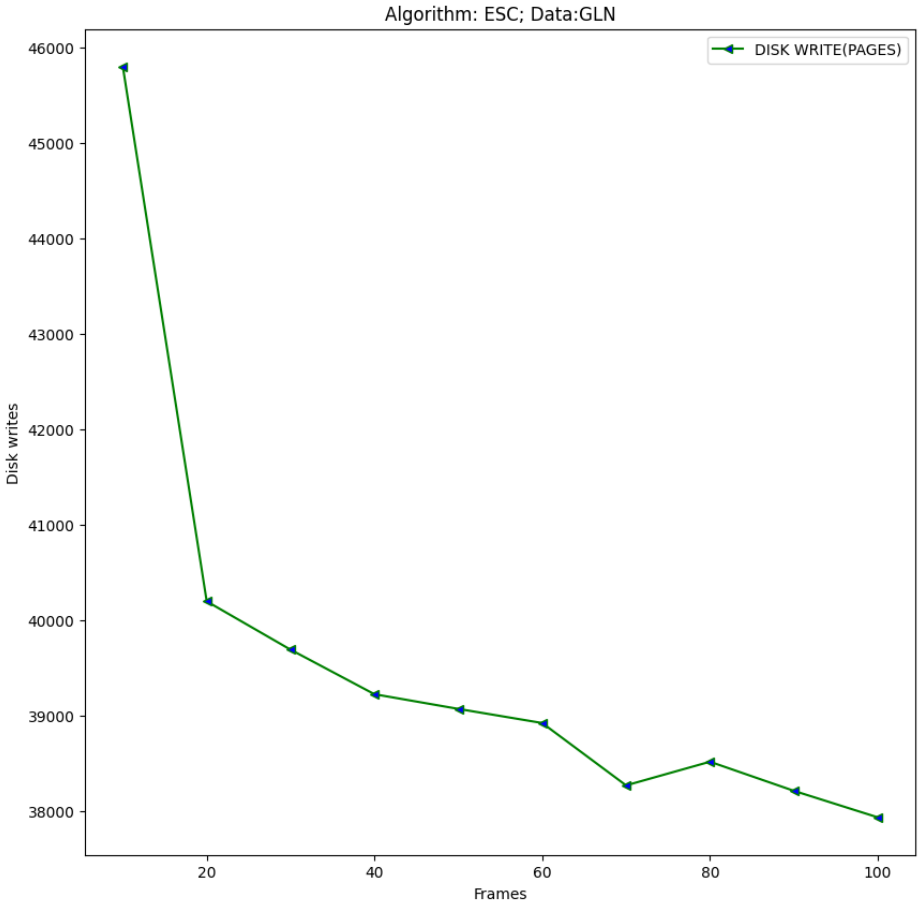
Page faults:



Interrupts:

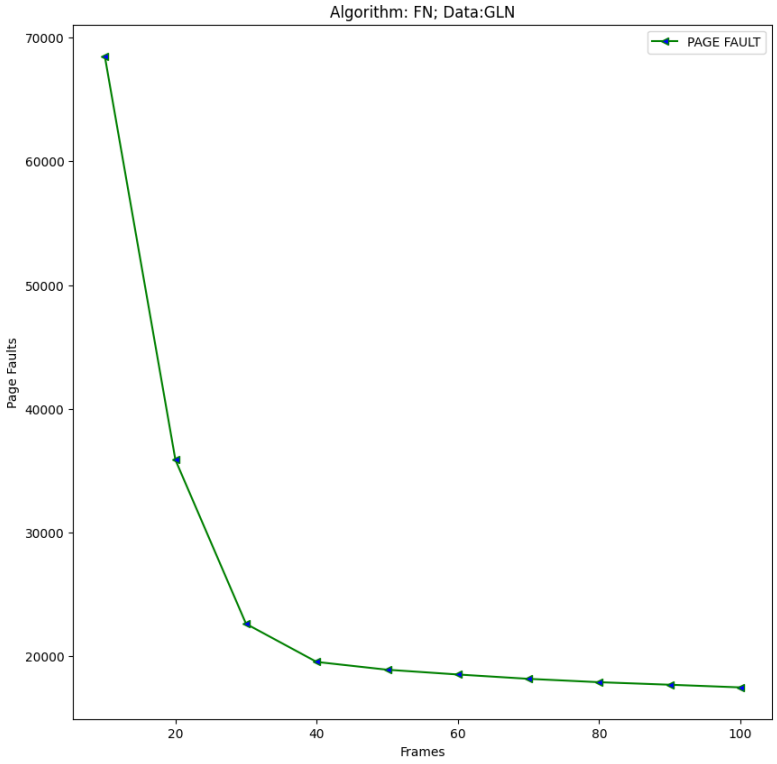


Disk writes:

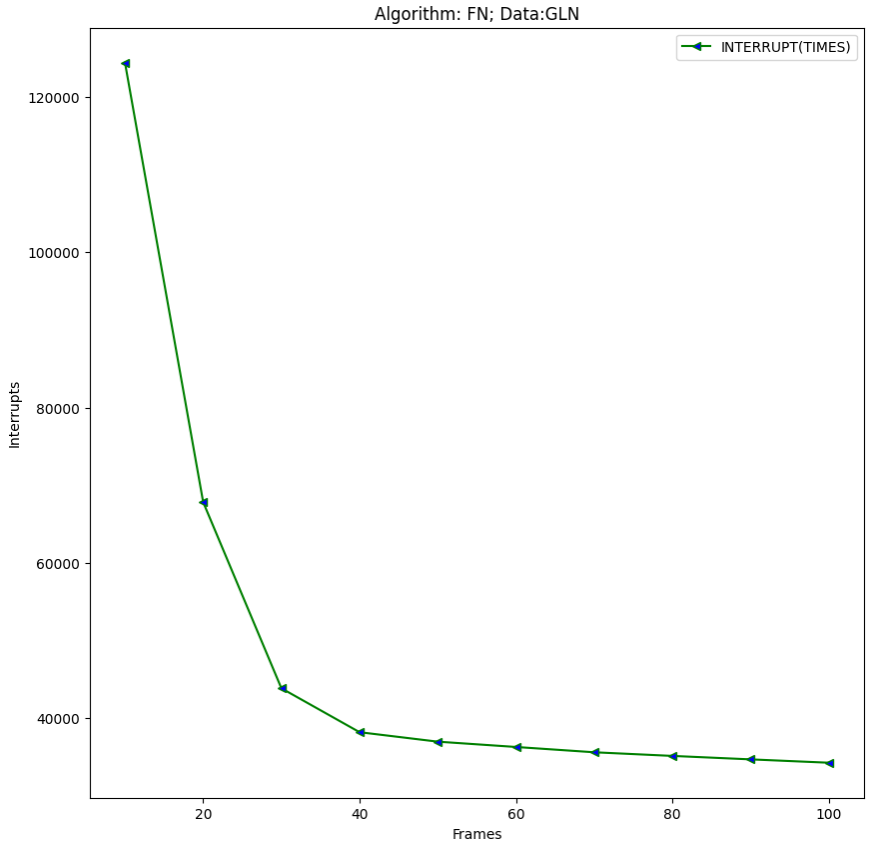


4.3.4 Furthest neighbor algorithm

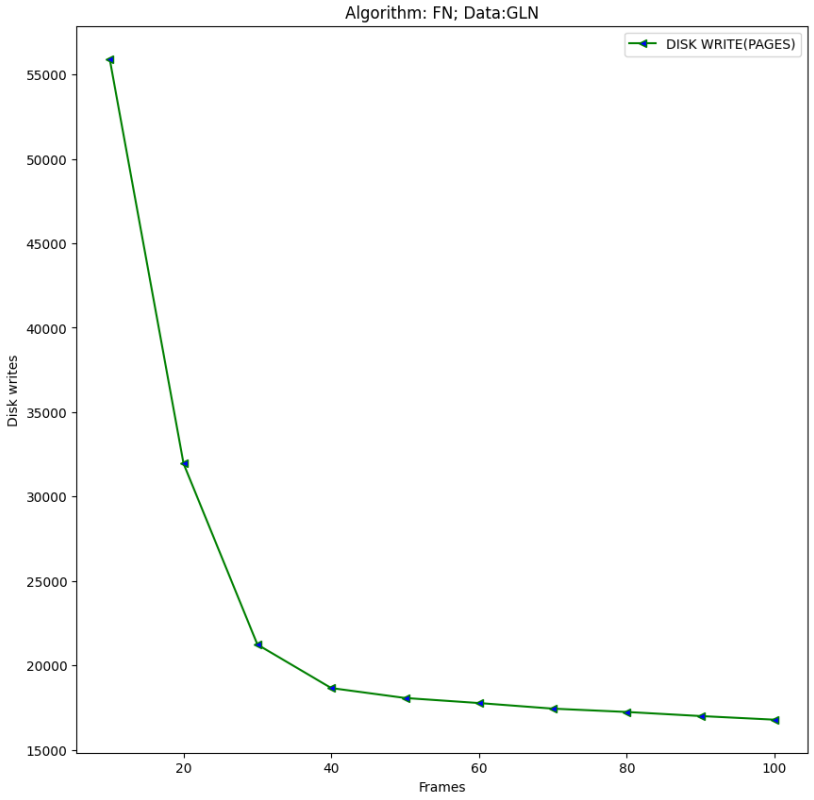
Page faults:



Interrupts:



Disk writes:



(5) Comparison result & remarks

(5-1) Numerical analysis:

Numerical analysis is provided for performance evaluation. Yellow denotes the champion in each competition item, and green denotes the second place.

Data: random									
frames	10			50			100		
Cost type	Page faults	Interrupts	Disk writes	Page faults	Interrupts	Disk writes	Page faults	Interrupts	Disk writes
FIFO	119024	237930	118906	115019	229988	114969	109939	219778	109839
Optimal	107693	175090	67397	89098	155369	66271	76206	137738	61532
ESC	119000	178941	59941	114926	173468	58542	109850	167059	57209
FN	119007	211541	92534	115045	206201	91156	110023	198819	88796

Chart 5.1. We can see that ESC algorithm is more outstanding in random dataset. Notice that ESC even outperforms optimal in disk write item. This may due to the design of hardware behavior.

Data: Locality									
frames	10			50			100		
Cost type	Page faults	Interrupts	Disk writes	Page faults	Interrupts	Disk writes	Page faults	Interrupts	Disk writes
FIFO	79683	159326	79643	24269	48488	24219	6428	12756	6328
Optimal	52401	95809	43408	12110	24003	11893	5762	11424	5662
ESC	86264	132766	46502	76762	118299	41537	70111	109055	38944
FN	79594	144144	64550	24303	46764	22461	6349	12596	6247

Chart 5.2. Under locality dataset, the optimal algorithm is dominating the performance competition as expected. For ESC, we can still see its advantage in disk write. However, ESC benefits little from the increasing frame numbers, making it weak under large memory size. For FIFO and FN, we can observe that they have similar page fault rates, while FN's prior performance in disk writes stands out in the clutch.

Data: G-D Locality									
frames	10			50			100		
Cost type	Page faults	Interrupts	Disk writes	Page faults	Interrupts	Disk writes	Page faults	Interrupts	Disk writes
FIFO	78756	157492	78736	26707	53364	26657	17971	35842	17871
Optimal	46055	84872	38817	14039	27352	13313	11467	22482	11015
ESC	77380	123200	45820	63711	102882	39171	61420	99389	37969
FN	68490	124479	55989	18924	37021	18097	17497	34314	16817

Chart 5.3. We can see that FN outscored FIFO even more in GDL dataset. However, this gap is decreased as the frame size grows. FN that calculates simply distances for only one point may be not enough in a large-scaled dataset.

(5.2) Remarks on optimal & ESC:

Optimal: Optimal shows the best result to obtain with page replacement algorithm. For example, we could find algorithms that performance fits the similar curve as optimal algorithm, as an approximation to optimal.

ESC: Enhanced second-chance makes use of hardware-set bits. However, in a simulation scheme, it is **difficult** to simulate the dynamic and stochastic **behavior of hardware**. Applying real-world dataset might give more insight about such algorithm's potential.

(5.3) Main Competitor: FIFO

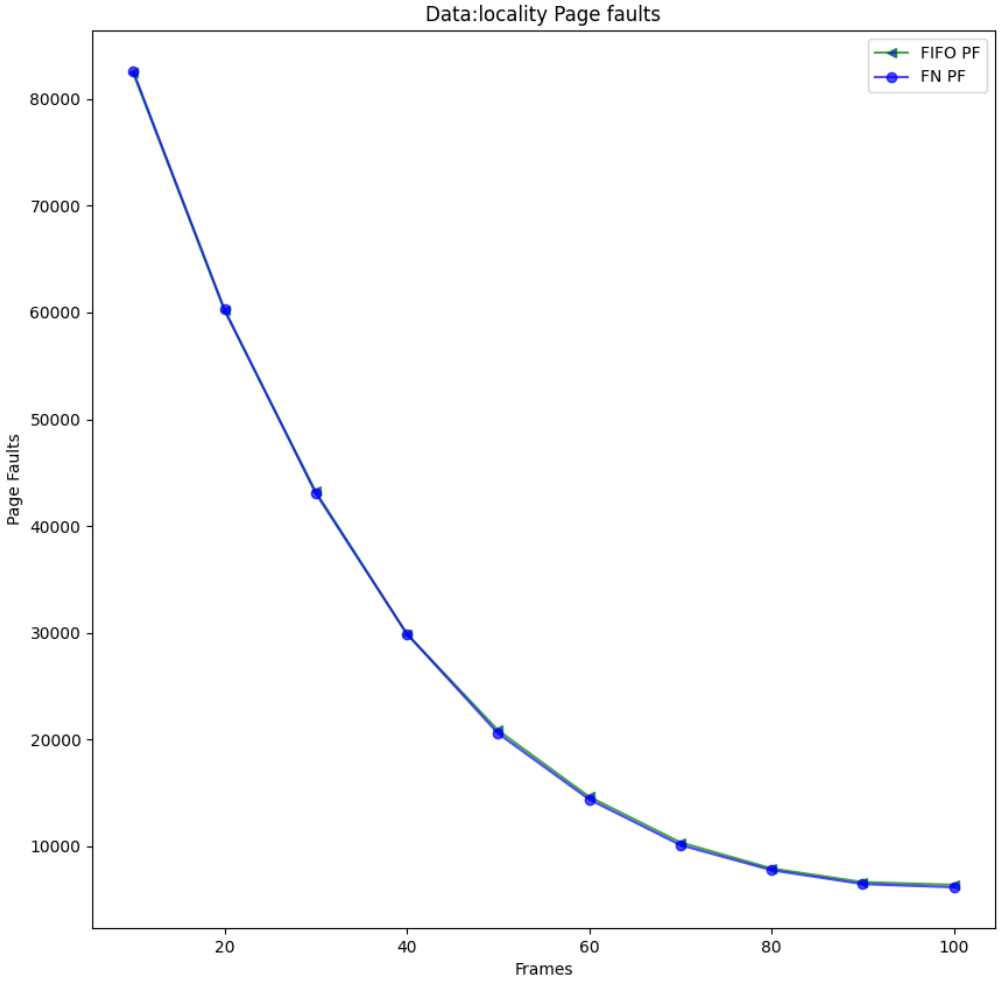


Fig 5.1. Page fault of FIFO vs FN dataset: locality

We can see that FN and FIFO almost overlap in locality dataset. Making FIFO a most challenging opponent to FN. However, previous results have shown FIFO lacks ability to predict hardware behavior.

Remarks on FIFO: FIFO performs well in the simulation scheme, but it is considered as a trivial solution to real-world applications. This may due to the **lack consideration of hardware** behavior. The “first-in” page has a large chance being modified, causing additional disk write cost.

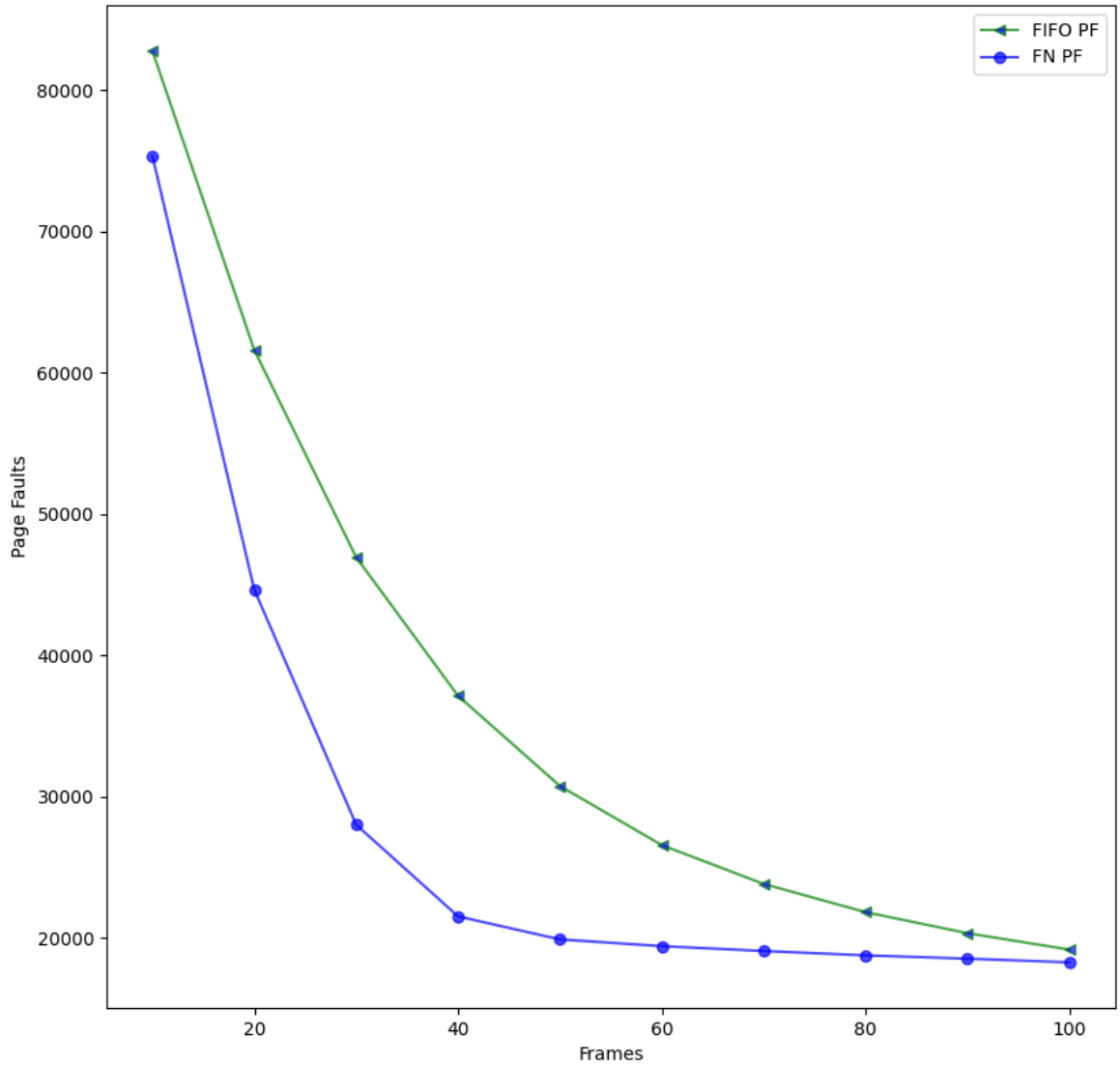


Fig 5.2. Page fault of FIFO vs FN dataset: Gaussian-distributed locality with noise

Remarks on FN: Though surpassing FIFO in low number of frames. FN show weakness when the frames get larger. To make use of the distance between pages, we should implement algorithms that consider higher dimension in complexity (more than absolute distance) and scaling (more than 1 page).

Clustering algorithms could be one of the efficient decision-making algorithms. We could aims to cluster localities in memory and labeled them into “hot/cold“ clusters. Therefore, choose the victim frame form the coldest cluster. One of the possible solutions could be **Kernel density estimation** (KDE) [3]. KDE could fit 1-d data and therefore cluster the data by local minima. The advantages of using KDE include computational efficiency and dynamic/unlimited cluster numbers.

We should also take care of input reference string data. With notice of real-world page reference dataset's features. We could provide a better clustering / estimation algorithm.

Conclusion: I have implemented farthest neighbor algorithm and compare it with current page replacement algorithms. Though FN show advantages in many simulated cases. It is still worth analyze on more complicated and genuine dataset, and optimize FN into a more suitable page replacement algorithm.

Reference

[1] Abraham Silberschatz, Peter B. Galvin, and Greg Gagne, "Operating System Concepts", 10th Edition, Wiley, 2019.
[2] Sun, Jingwen & Du, Weixing & Shi, Niancai. (2018). A Survey of kNN Algorithm. Information Engineering and Applied Computing. 1. 10.18063/ieac.v1i1.770.
[3] Wei-Jun Wang, Yong-Xi Tan, Jian-Hui Jiang, Jian-Zhong Lu, Guo-Li Shen, Ru-Qin Yu,
Clustering based on kernel density estimation: nearest local maximum searching algorithm,
Chemometrics and Intelligent Laboratory Systems,
Volume 72, Issue 1,
2004,
Pages 1-8,
ISSN 0169-7439,
https://doi.org/10.1016/j.chemolab.2004.02.006.
(https://www.sciencedirect.com/science/article/pii/S016974390400053X)