CS 350M: COMPUTER SYSTEMS
ASSIGNMENT- 24/03/2016

JAYANTHI SAI MURALIDHAR
Roll No.- 130108014
EEE Dept.
IIT Guwahati

The following is the code that simulates the paging system for FIFO, LRU and NFU page replacement algorithms taking 'Page Frames' as an input parameter:

*Page references are read from a file *pageReferences.txt*

*Plots were obtained using GNUPLOT and for the same, data is stored in the file *data.dat*

```
#include <iostream>
#include <cmath>
#include <fstream>
#include <stdio.h>
#include <stdlib.h>

using namespace std;

#define pageReference_total 1000
ofstream outfile;

// Functions Declaration
void fifo(int);
void lru(int);
void nfu(int);

int main(){
	outfile.open("data.dat");
	int select;
	cout << "Type 1 for FIFO, 2 for LRU, 3 for NFU: " << endl;
	cin >> select;
	// int pageFrames;  Cin  >> pageFrames;

	int pageFrames=100; // Edit the number of pageFrames here
	while(pageFrames>0){
		switch(select){
			case 1:
				fifo(pageFrames);
				break;
			case 2:
				lru(pageFrames);
				break;
			case 3:
				nfu(pageFrames);
				break;
			default:
				cout << "Error. Please press 1 or 2 or 3"<< endl;
				break;
		}
		// cout << "pageFrames: " << pageFrames << endl;
		pageFrames=pageFrames-1;
	}
```

```
	// Plotting the graph
	outfile.close();
	FILE *gnuplot = popen("gnuplot -persist","w");
	fprintf(gnuplot,"plot 'data.dat' with linespoints\n");
	fclose(gnuplot);
	return 0;
}

void fifo(int pageFrames){

	int temp,i;
	int num[pageFrames]={0};
	int hit,hitCount=0;
	int pageReference,pageReference_n=0;

	// File to read Page Refernces
	ifstream infile("pageReferences.txt");
	while (infile >> pageReference){
		pageReference_n=pageReference_n+1;
		hit=0;
		for (temp=0;temp<pageFrames;temp++){
			if(num[temp]==pageReference) {
				hit=1;
				break;
			}
		}
		if(hit==1)
			hitCount=hitCount+1;
		else{
			for(temp=pageFrames-2;temp>=0;temp--)
				num[temp+1]=num[temp];
			num[0]=pageReference;
		}
	}
	outfile << pageFrames << "\t" << pageReference_n-hitCount << "\n";
}



/*Please Turn Over*/
```

```
void lru(int pageFrames){
	int temp,i;
	int num[pageFrames]={0};
	int hit=0,hitCount=0;
	int pageReference,pageReference_n=0;

	ifstream infile("pageReferences.txt");
	while (infile >> pageReference){
		pageReference_n=pageReference_n+1;
		hit=0;
		for (temp=0;temp<pageFrames;temp++){
			if(num[temp]==pageReference) {
				hit=1;
				break;
			}
		}

		//Put the page in the list
		if (hit==1){
			hitCount=hitCount+1;
			for(i=temp-1;i>=0;i--)
				num[i+1]=num[i];
			num[0]=pageReference;
		}
		else {
			for(i=pageFrames-2;i>=0;i--)
				num[i+1]=num[i];
			num[0]=pageReference;
		}
	}
	outfile << pageFrames << "\t" << pageReference_n-hitCount << "\n";
}


/* Please Turn Over*/
```

```
void nfu(int pageFrames){

	int counter[pageReference_total]={0}; // Considering max. no. of pages=1000
	int i,min=0,min_pageFrames,index;
	int num[pageFrames]={0};			// The page Frames
	int hit=0,hitCount=0,flag;
	int pageReference,pageReference_n=0;

	ifstream infile("pageReferences.txt");
	while (infile >> pageReference){
		cout << "Page Refrence: " << pageReference << endl;
		pageReference_n=pageReference_n+1;
		hit=0;
		for (i=0;i<pageFrames;i++){
			if(num[i]==pageReference) {
				hit=1;
				break;
			}
		}
		counter[pageReference-1]=counter[pageReference-1]+1;
		/*for (i=0;i<10;i++)
			cout << "C: " << counter[i] << "\t";
		cout << endl; */
		if(hit==1)
			hitCount=hitCount+1;
		else{
			// Make sure that no num[] is Zero
			flag=0;
			for (i=0;i<pageFrames;i++){
				if(num[i]==0){
					flag=1;
					break;
				}
			}
			// cout << "FLAG: " << flag << endl;
			if(flag==1){
				for(i=pageFrames-2;i>=0;i--)
					num[i+1]=num[i];
				num[0]=pageReference;
			}
```

```
                        // Replacement. Tie-Breaker: FIFO
                        else if (flag==0){
                                min=1000;
                                for(i=0;i<pageFrames;i++){
                                        index=num[i]-1;
                                        if(counter[index]<=min){
                                                min=counter[index];
                                                min_pageFrames=i;
                                        }
                                }
                                for(i=min_pageFrames;i>0;i--)
                                        num[i]=num[i-1];
                                num[0]=pageReference;
                        }
                }
                /*for(i=0;i<5;i++)
                        cout << "Num: " << num[i]<< "\t";
                cout << endl << endl;*/
        }
        outfile << pageFrames << "\t" << pageReference_n-hitCount << "\n";
}
```

PLOTS

***X-Axis: No. of page Frames**

***Y-Axis: No.of page faults per 1000 memory references**

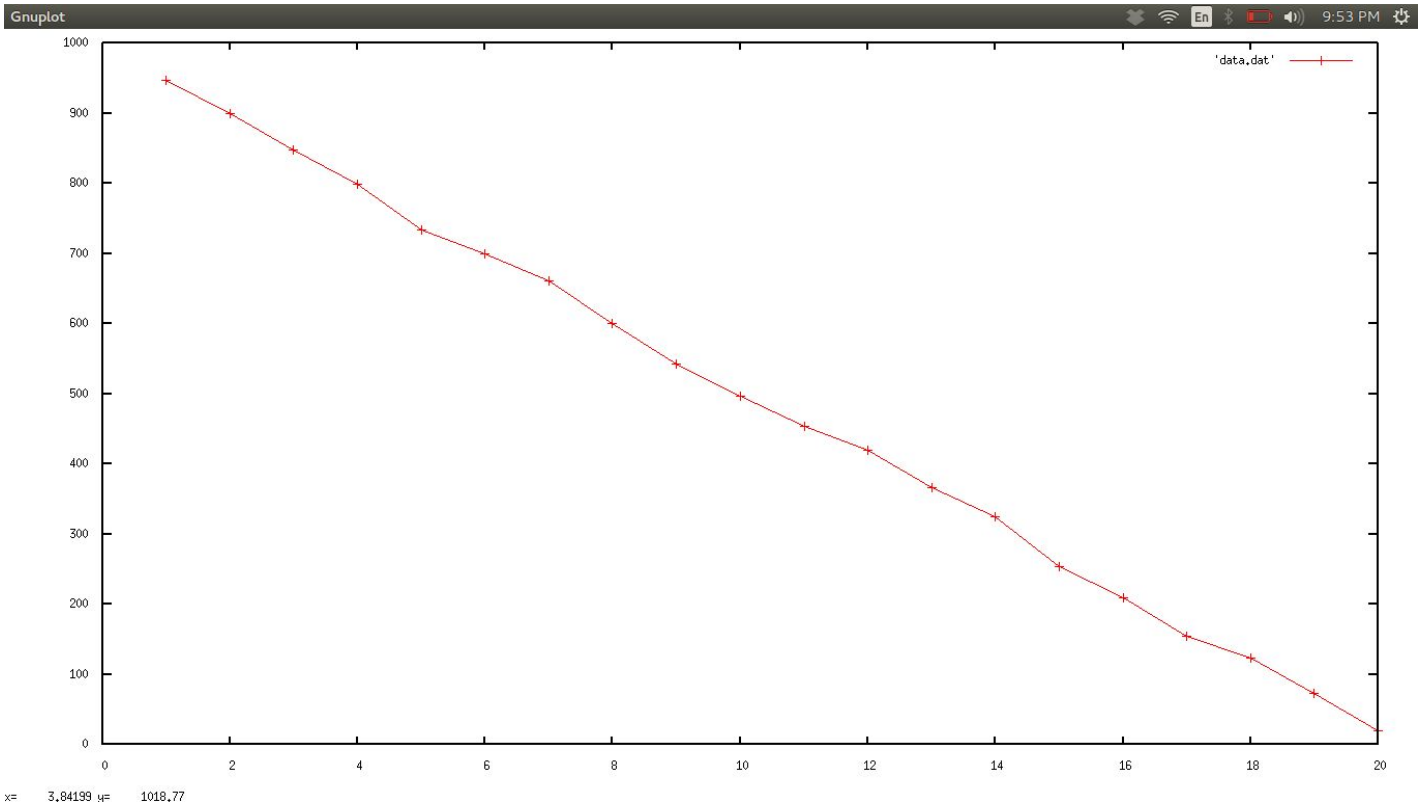

Fig 1: FIFO: 20 Unique pages: 1000 Page References

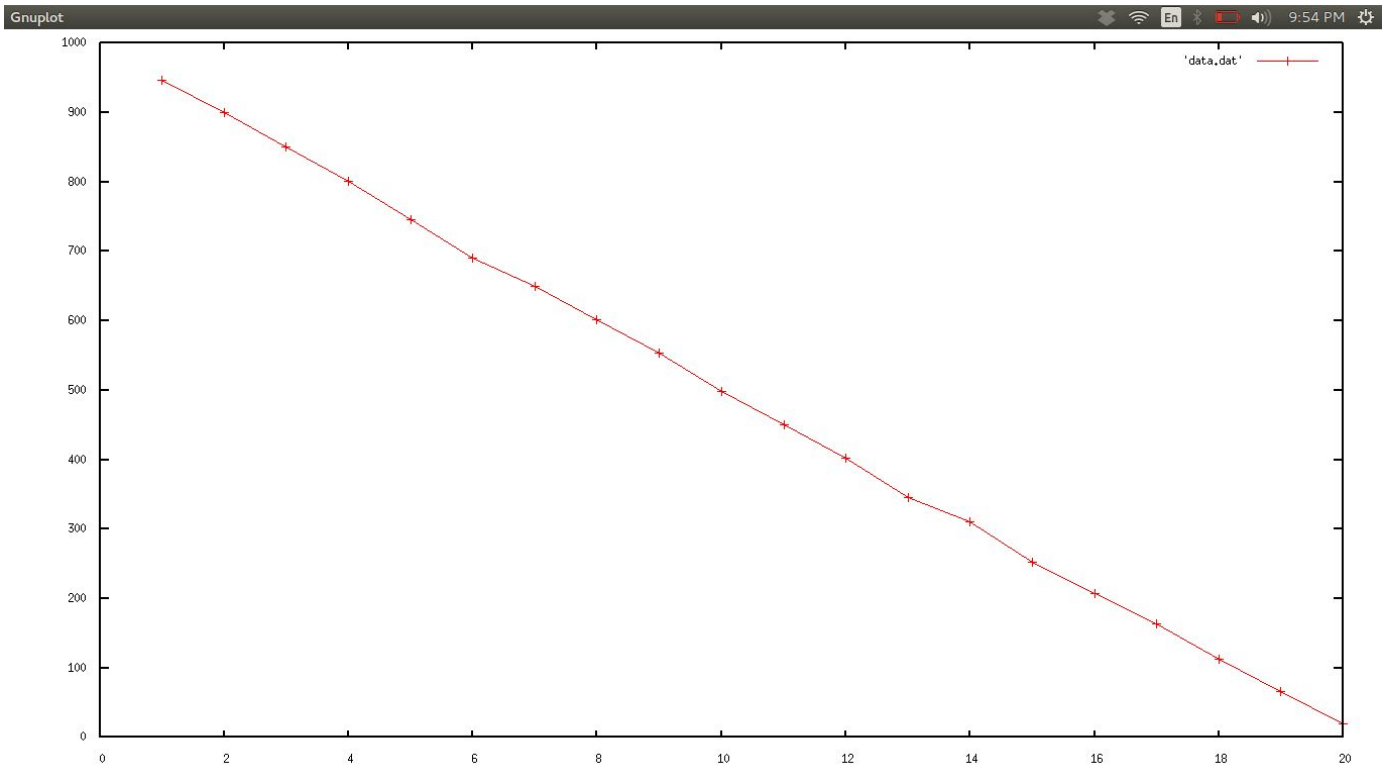

Fig 2: LRU: 20 Unique pages: 1000 Page References

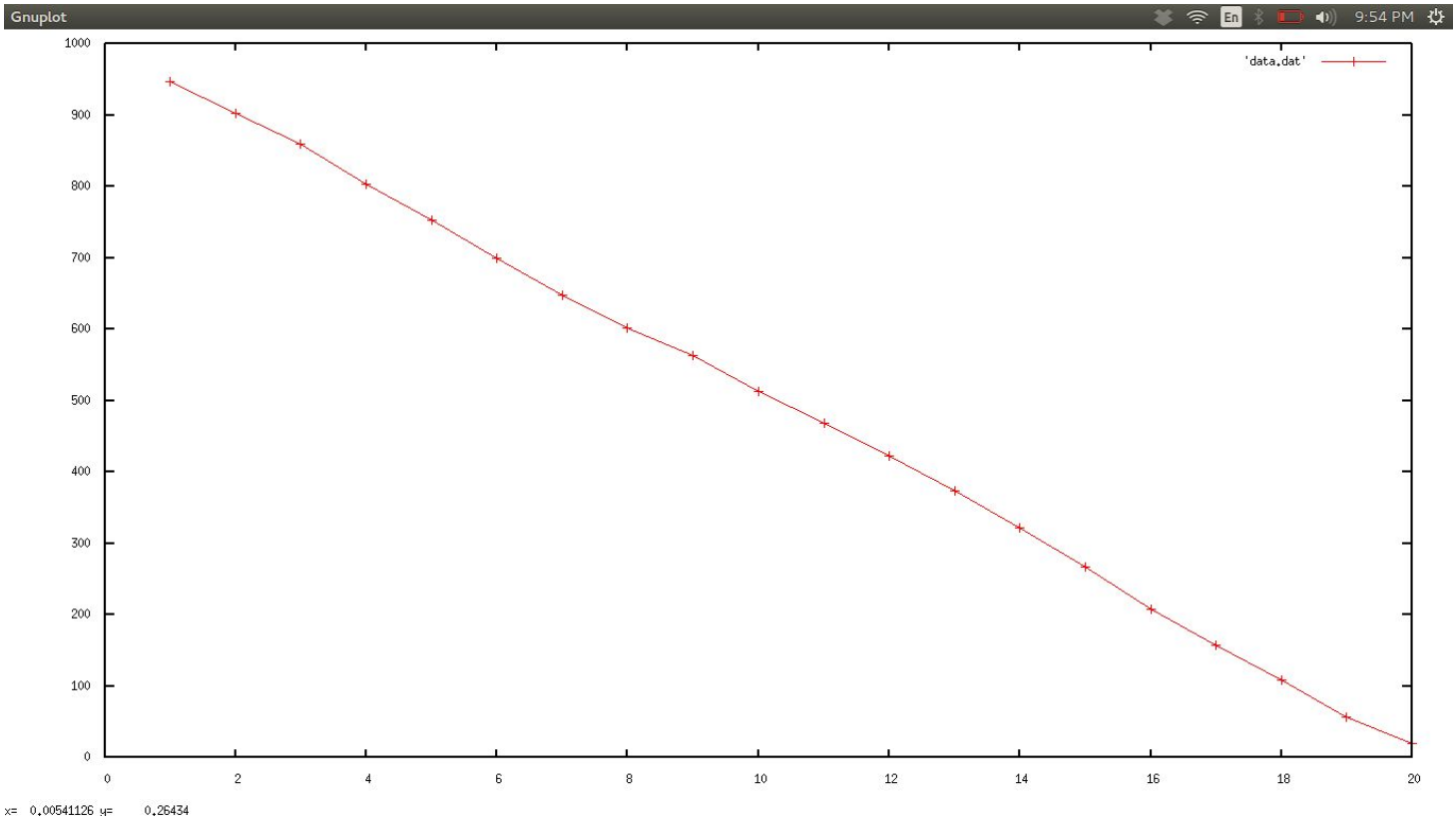

Fig 3: NFU: 20 Unique pages: 1000 Page References

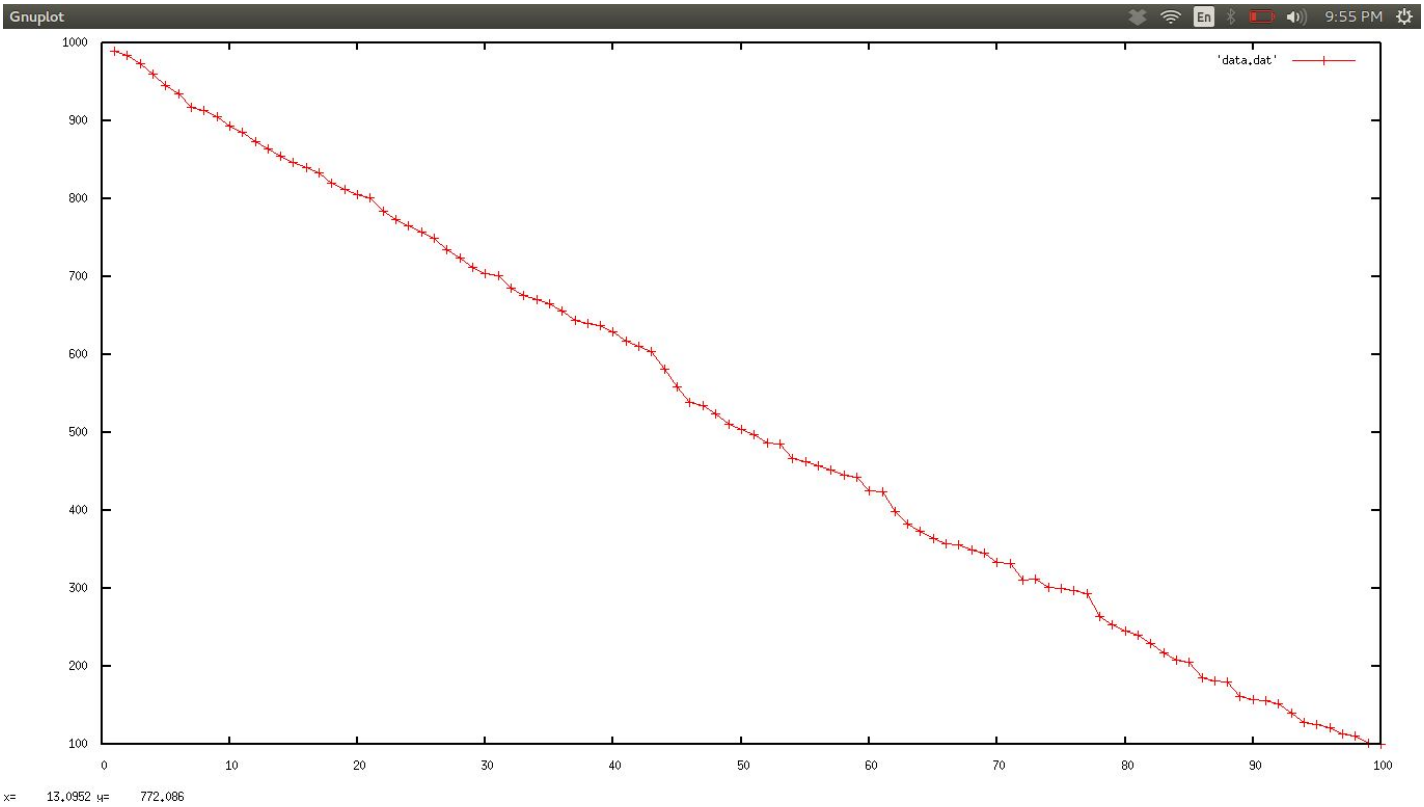

Fig 4: FIFO: 100 Unique pages: 1000 Page References

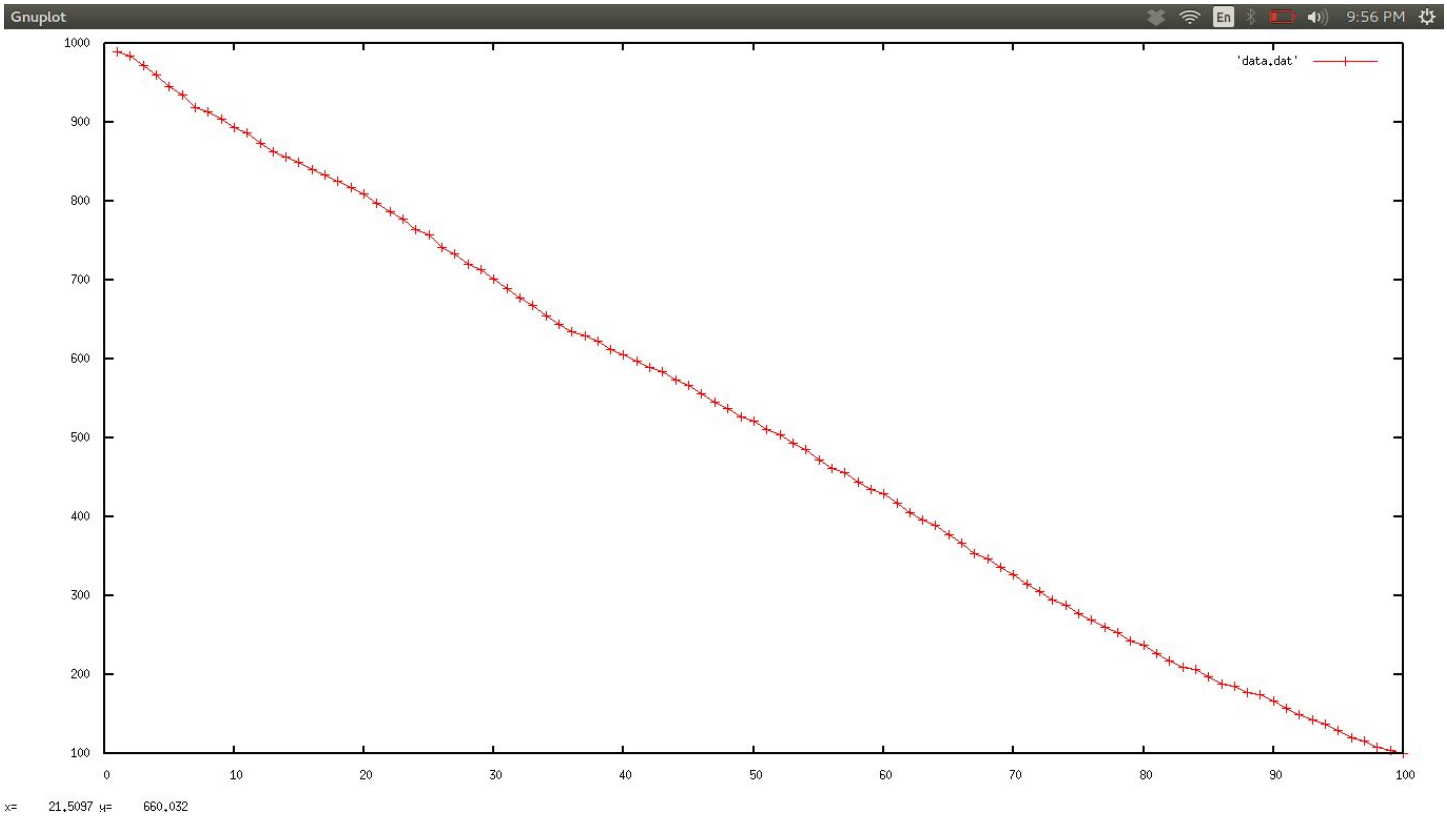

Fig 5: LRU: 100 Unique pages: 1000 Page References

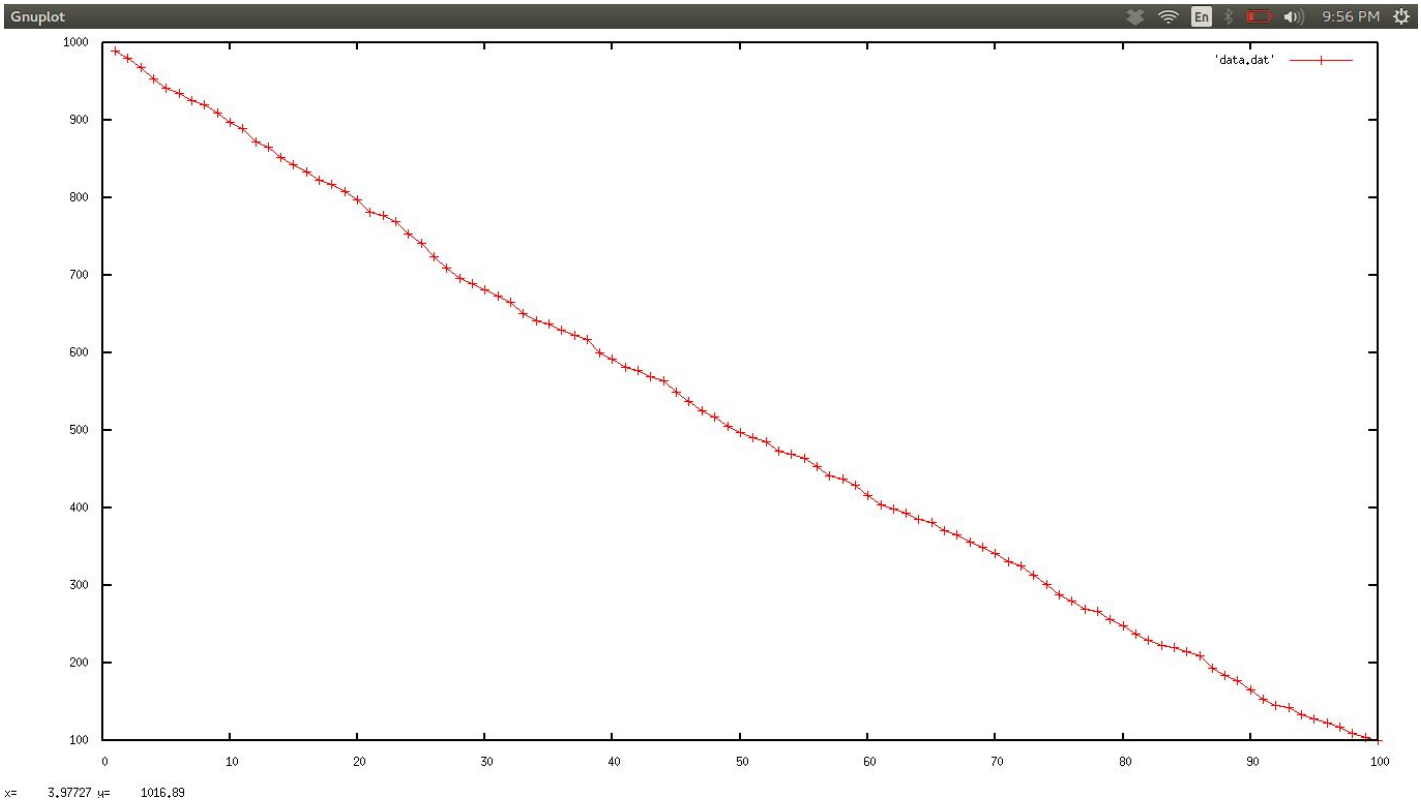

Fig 6: NFU: 100 Unique pages: 1000 Page References