CO 322
HASH TABLES

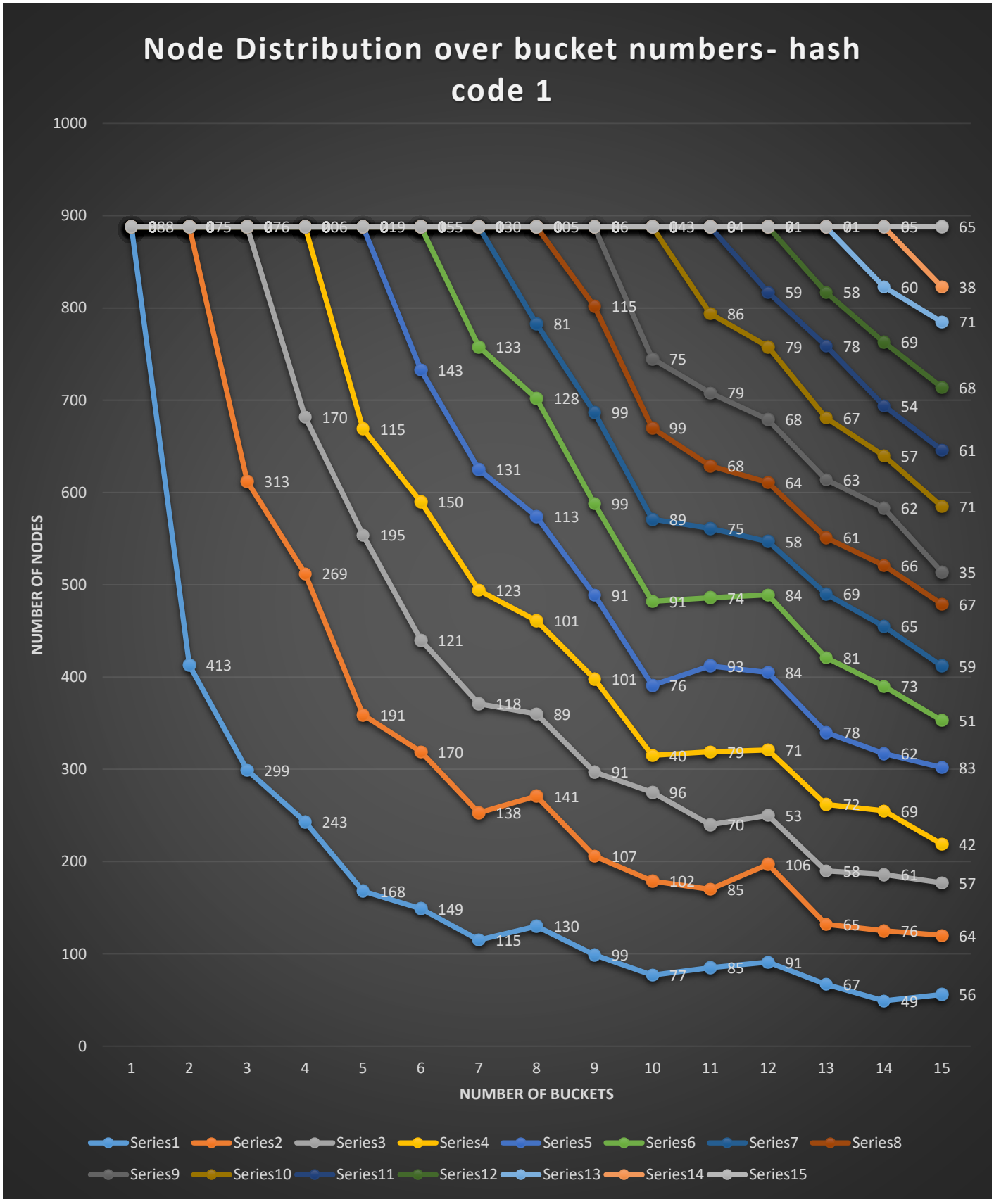
NAME : WIMALASIRI KPGP

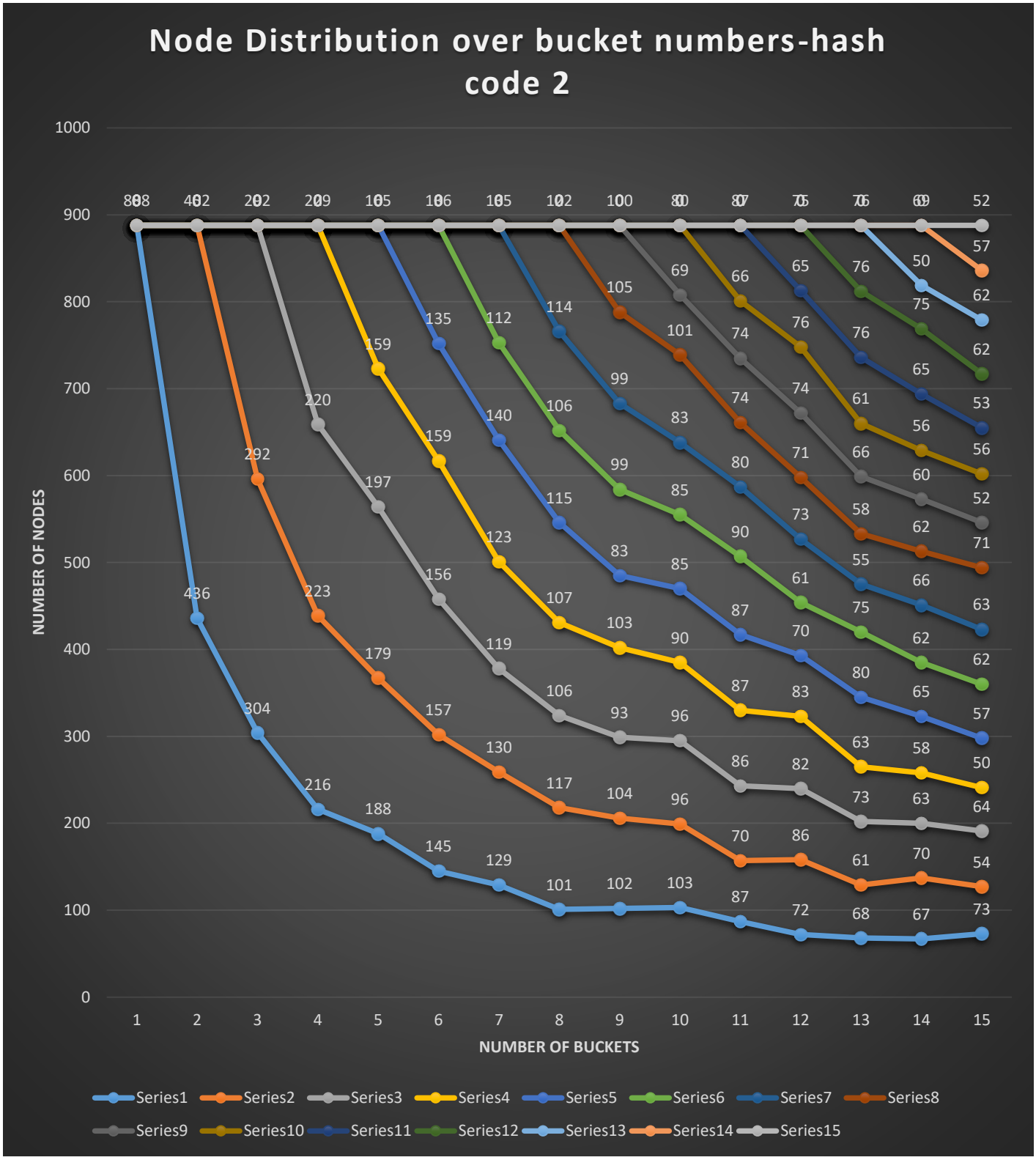
REG NO : E/14/403

SEMESTER : 5th

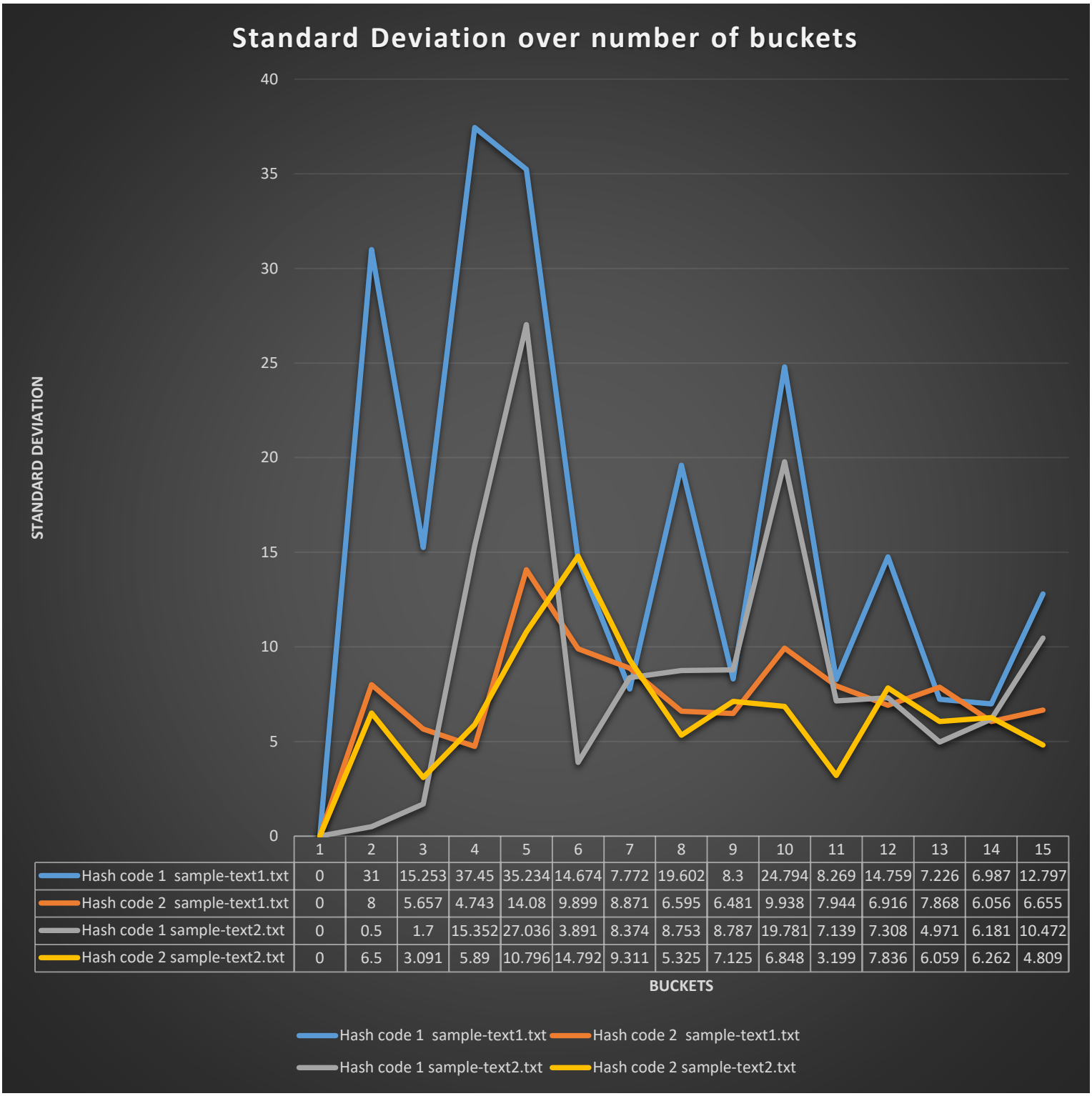
DATE : 03/02/2018

Considering the sample-text1.txt



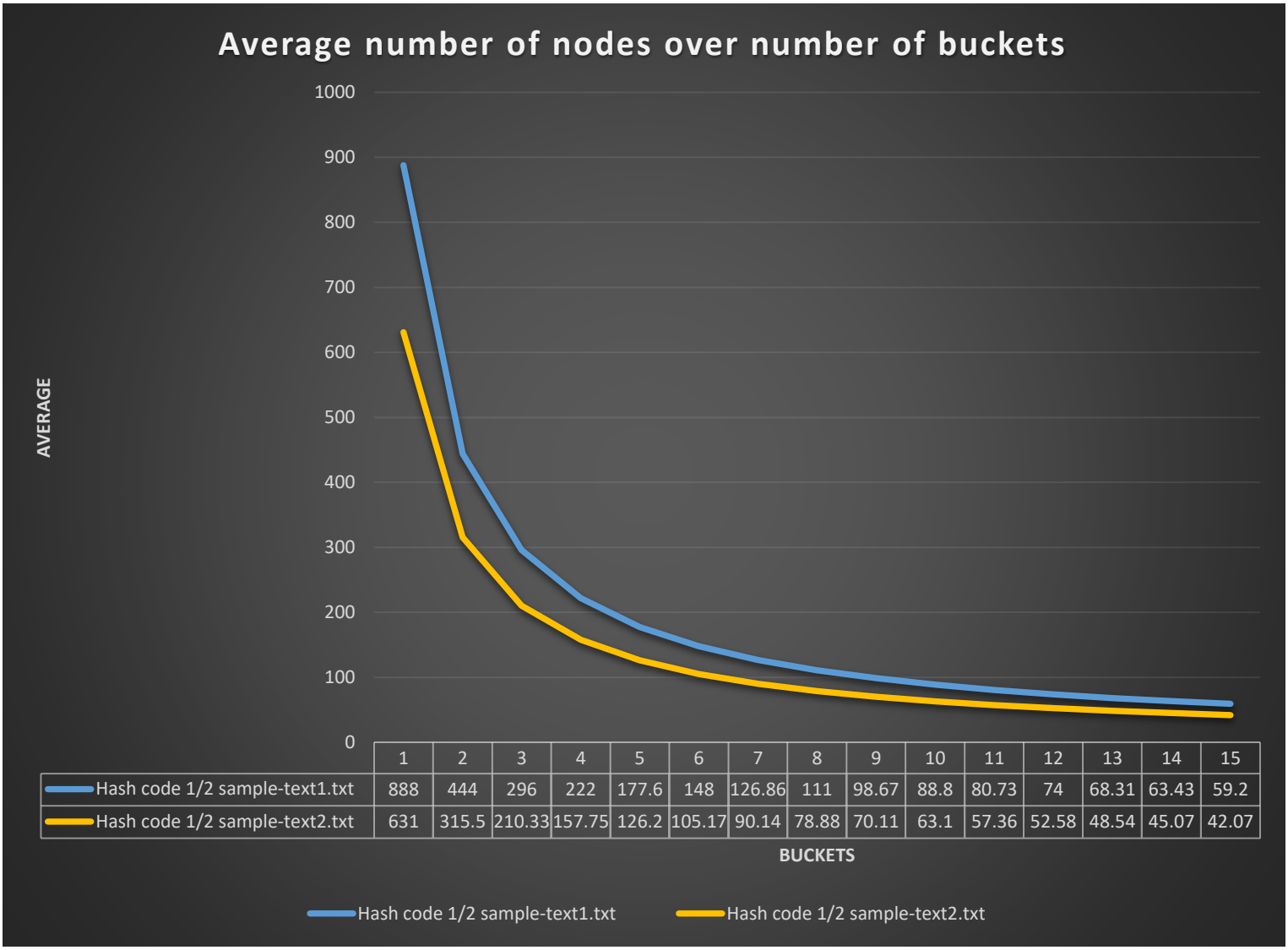


- According to the tables given above hash code 2 is more stable with the variation of bucket numbers (has the better distribution) for sample-text1.txt



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hash code 1 sample-text1.txt	0	31	15.253	37.45	35.234	14.674	7.772	19.602	8.3	24.794	8.269	14.759	7.226	6.987	12.797
Hash code 2 sample-text1.txt	0	8	5.657	4.743	14.08	9.899	8.871	6.595	6.481	9.938	7.944	6.916	7.868	6.056	6.655
Hash code 1 sample-text2.txt	0	0.5	1.7	15.352	27.036	3.891	8.374	8.753	8.787	19.781	7.139	7.308	4.971	6.181	10.472
Hash code 2 sample-text2.txt	0	6.5	3.091	5.89	10.796	14.792	9.311	5.325	7.125	6.848	3.199	7.836	6.059	6.262	4.809

- According to the table above Hash code 2 shows a stable variation in both sample-text1 and sample-text2.
- When consider about sample-text1 or sample-text2 hash code 2 shows relatively less Standard Deviation, which express that hash code 2 distribute nodes among the buckets better.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hash code 1/2 sample-text1.txt	888	444	296	222	177.6	148	126.86	111	98.67	88.8	80.73	74	68.31	63.43	59.2
Hash code 1/2 sample-text2.txt	631	315.5	210.33	157.75	126.2	105.17	90.14	78.88	70.11	63.1	57.36	52.58	48.54	45.07	42.07

- Any proper idea about the comparison of node distribution over number of buckets can't be obtained with the average variation above.
- Both curves have the same shape

CONCLUSION

According to the analysis above hash code 2 distribute nodes evenly among the buckets than hash code 1.

APPENDIX

Hash code number 1

```
char [] charKey = key.toCharArray();
        for(int i=0;i<key.length();i++){
            code = code + (charKey[i])*(int)(pow(10,i));
        }
        if(code<0){
            code *=-1;
        }
        return code%buckets;
```

Hash code number 2

```
char [] charKey = key.toCharArray();
        for(int i=0;i<key.length();i++){
            code = code + charKey[i];
        }
        if(code<0){
            code *=-1;
        }
        return code%buckets;
```