各个Hash算法的执行时间

在程序实际执行的过程中，不同的Hash算法计算时间不同。从各个Hash算法的代码中，可以笼统地看出它们各自的时间复杂度。然而在实际应用中，同一级别的时间复杂度仍然有可能导致实际执行时间有大的差别。因此，我们组在预实验中，首先计算了各个Hash的计算时间。在实验过程中，遇到的难题以及解决方法如下。

时间计算代码1

```
    time_t start_t, end_t;
    double diff_t;
    //start timer
    time(&start_t);
    // 执行算法
    time(&end_t);
    diff_t = difftime(end_t, start_t);
    printf("执行时间 = %f\n", diff_t);
```

主程序

```
int main() {
    time_t start_t, end_t;
    double diff_t;
    //start timer
    time(&start_t);
        FILE* f_string = NULL;
        f_string = fopen("string.txt", "r");

        if (!f_string)
```

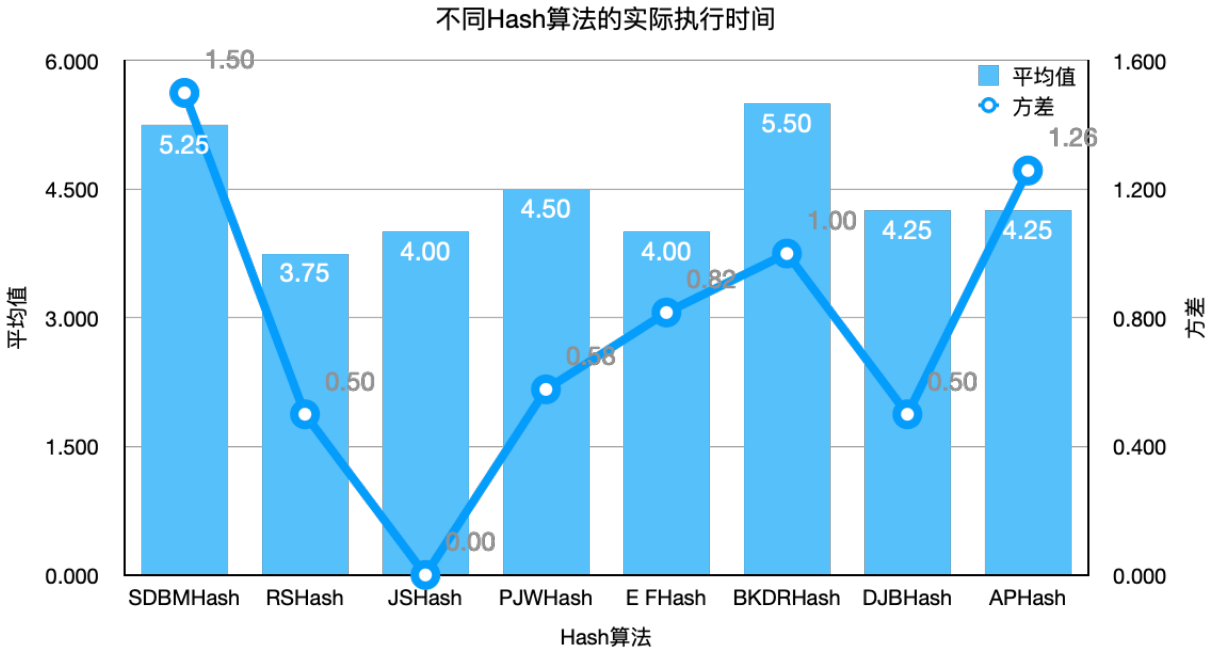
```
            {
            printf("Failed.\n");
            return 0;
            }

        for (size_t i = 0; i < 200000; i++)
            {
                char str[99];
                fscanf(f_string, "%s", str);
                printf("%12s 的hash值是: %u\n", str, APHash(str));
            }
        fclose(f_string);
    // conclude timer
    time(&end_t);
    diff_t = difftime(end_t, start_t);
    printf("执行时间 = %f\n", diff_t);
}
```

结果

以下是每个算法执行四次，分别花费的时间

- SDBMHash 6 4 7 4
- RSHash 4 4 3 4
- JSHash 4 4 4 4
- PJWHash 4 4 5 5
- ELFHash 5 4 4 3
- BKDRHash 7 5 5 5
- DJBHash 4 4 4 5
- APHash 4 6 3 4

其中，各个算法的执行平均时间和方差如下图，得出结论---



分析（优点与缺点）

时间计算代码2

```
#include <time.h>
#include <stdio.h>

int main()
{
   clock_t start_t, end_t;
   double total_t;
   int i;

   start_t = clock();
   printf("程序启动，start_t = %ld\n", start_t);
   printf("开始一个大循环，start_t = %ld\n", start_t);
   for(i=0; i< 10000000; i++)
   {
    //执行算法
```

```
    }
    end_t = clock();
    printf("大循环结束，end_t = %ld\n", end_t);

    total_t = (double)(end_t - start_t) / CLOCKS_PER_SEC;
    printf("CPU 占用的总时间：%f\n", total_t  );
    printf("程序退出...\n");

    return(0);
}
```

主程序

```
int main() {
    //start timer
    clock_t start_t, end_t;
    double total_t;
    int i;
    start_t = clock();
    printf("程序启动，start_t = %ld\n", start_t);
    printf("开始一个大循环，start_t = %ld\n", start_t);

    for(i=0; i< 10000000; i++)
    {
        FILE* f_string = NULL;
        f_string = fopen("string.txt", "r");

        if (!f_string)
            {
            printf("du wen jian shi bai.\n");//failed
            return 0;
            }

        for (size_t i = 0; i < 200000; i++)
            {
                char str[99];
                fscanf(f_string, "%s", str);
                printf("%12s 的hash值是：%u\n", str, RSHash(str));
            }
```

```
        fclose(f_string);

    // conclude timer
    end_t = clock();
    printf("大循环结束，end_t = %ld\n", end_t);
    total_t = (double)(end_t - start_t) / CLOCKS_PER_SEC;
    printf("CPU 占用的总时间：%f\n", total_t  );
    printf("程序退出...\n");
    return(0);
    }
}
```

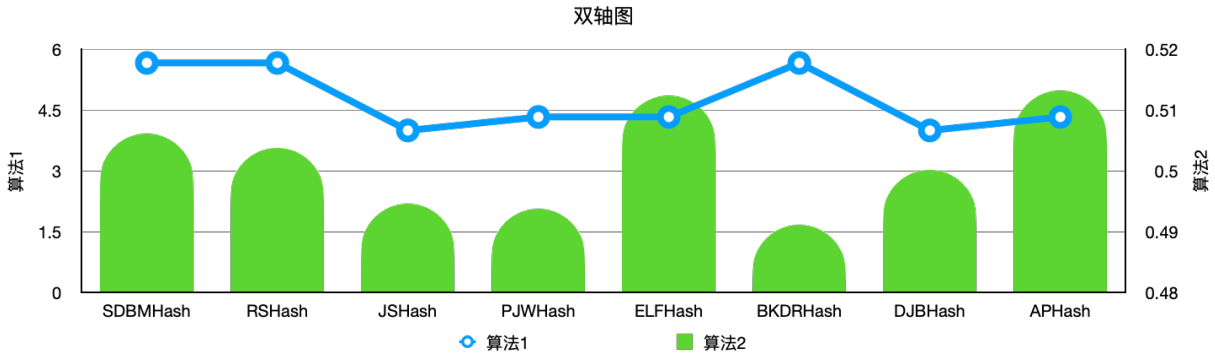
结果

以下是每个算法执行四次，CPU 占用的总时间

- SDBMHash 0.498270 0.528836 0.491364
- RSHash 0.505803 0.500673 0.505015
- JSHash 0.490847 0.496962 0.495953
- PJWHash 0.484274 0.497424 0.499829
- ELFHash 0.493847 0.520552 0.522738
- BKDRHash 0.484216 0.493591 0.495606
- DJBHash 0.501530 0.493006 0.505955
- APHash 0.536680 0.503890 0.499214 0.499886

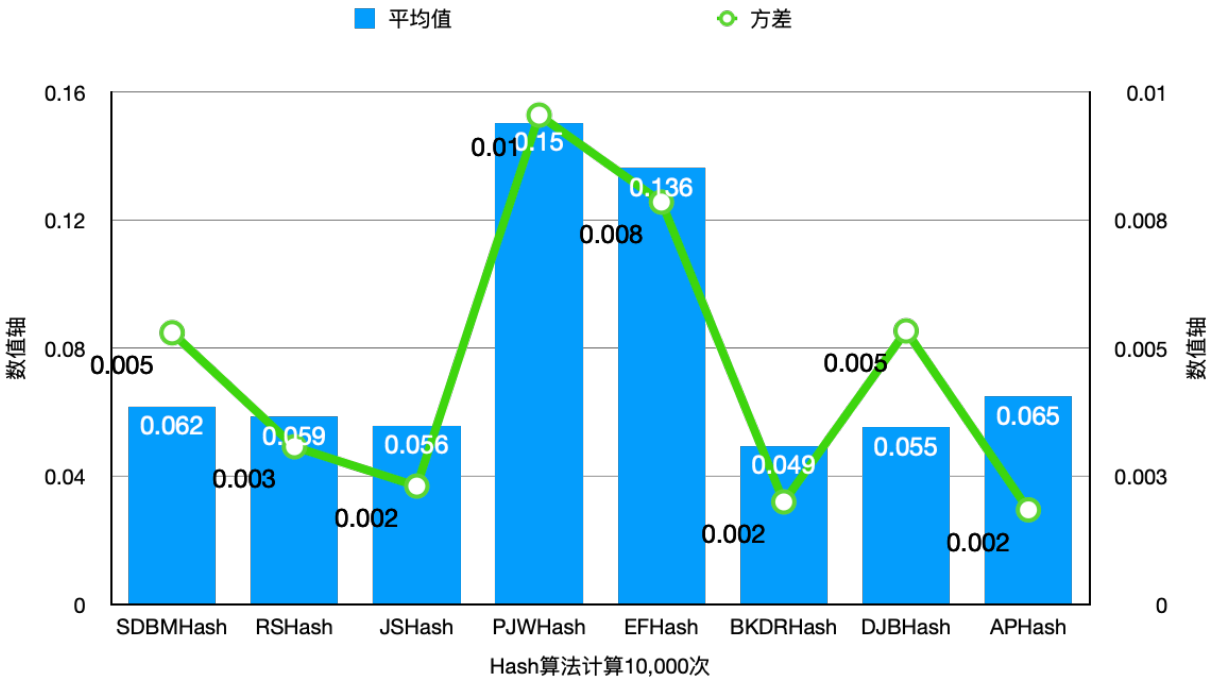
分析

两种方法之间的误差



时间计算代码 3

以上算法的执行时间包含`printf("%12s 的hash值是： %u\n", str, RSHash(str))`，因此结果中执行时间的大部分都花费在printf函数上。在下一节中，将去除导致执行时间不准确代码，仅仅包含Hash计算的时间。



得出结论 BKDRHash的计算时间最短。