

# 实验报告——lab3 better angels

PB20151793 宋玮

## 1. 分析原程序：

```
.ORIG x3000
    ADD R1, R1, #1
    ADD R2, R2, #1
    ADD R3, R3, #2
    ADD R7, R7, R0
    ADD R0, R0, #-2
    BRnz STOP
AGAIN2 ADD R7, R3, R1
    ADD R7, R7, R1
    LD R5, MOD
    AND R7, R7, R5
    ADD R1, R2, #0
    ADD R2, R3, #0
    ADD R3, R7, #0
    ADD R0, R0, #-1
    BRp AGAIN2
STOP   HALT
MOD    .FILL #1023
Fa     .FILL #930
Fb     .FILL #1
Fc     .FILL #1
Fd     .FILL #306
.END
```

从最后四行

**Fa .FILL #930**

**Fb .FILL #1**

**Fc .FILL #1**

**Fd .FILL #306**

#1 可以是  $f(0)$ , 也可以是  $f(1)$ , 经过对班级名单的查询, 得出  $b=0, c=1$ 。

于是  $a=20, b=0, c=1, d=94$ 。

可知这位同学的学号为 PB20000194。

## 2. 优化过程

### (1)原程序：

```
.ORIG x3000
    ADD R1, R1, #1
    ADD R2, R2, #1
    ADD R3, R3, #2
    ADD R7, R7, R0
    ADD R0, R0, #-2
    BRnz STOP
AGAIN2 ADD R7, R3, R1
    ADD R7, R7, R1
    LD R5, MOD
    AND R7, R7, R5
    ADD R1, R2, #0
    ADD R2, R3, #0
    ADD R3, R7, #0
    ADD R0, R0, #-1
    BRp AGAIN2
STOP   HALT
MOD    .FILL #1023
Fa     .FILL #930
Fb     .FILL #1
Fc     .FILL #1
Fd     .FILL #306
.END
```

$n=1$  和  $n=2$  的情况下, 程序只需执行 6 条指令。(不包括 halt)

除此之外, 程序执行的指令条数都与  $n$  存在线性关系。而该线性系数即为循环体 AGAIN2 的指令条数。

在原程序中, 循环体指令条数为 9 条。因此可以得出指令执行总条数  $g$  与  $n$  的关系式。如下：

$$g(n)=9(n-2)+6$$

测试数据：

$f(24)=706$                       指令数： $9 \times (24-2) + 6 = 204$ .

$f(144)=642$       同理得：指令数：1284

$f(456)=66$       同理得：指令数：4092

$f(1088)=2$       同理得：指令数：9780

$f(1092)=290$       同理得：指令数：9816

$f(2096)=898$       同理得：指令数：18852

$f(4200)=322$       同理得：指令数：37788

$f(8192)=514$       同理得：指令数：73716

$f(12000)=258$       同理得：指令数：107988

$f(14000)=898$       同理得：指令数：125988

平均指令条数为：38950.8

## (2) 优化方案

### ① 减少循环体指令条数。

通过观察原程序，发现有两条指令可以搬移到循环体外，不需要每次循环执行。

即 LD R5, MOD 和 AND R7, R7, R5

这两条指令的作用是对结果 mode 1024，显然可以放到循环体外，只需各执行一次。

修改完的程序如下：

```
.ORIG x3000
    ADD R1, R1, #1
    ADD R2, R2, #1
    ADD R3, R3, #2
    LD R5, MOD
    ADD R7, R7, R0
    ADD R0, R0, #-2
    BRnz STOP
AGAIN2 ADD R7, R3, R1
    ADD R7, R7, R1
    ADD R1, R2, #0
    ADD R2, R3, #0
    ADD R3, R7, #0
    ADD R0, R0, #-1
    BRp AGAIN2
    AND R7, R7, R5
STOP   HALT
MOD    .FILL #1023
Fa     .FILL #930
Fb     .FILL #1
Fc     .FILL #1
Fd     .FILL #306
.END
```

此时循环体的指令条数缩减至 7 条。

$n=1$  和  $n=2$  的情况下，程序只需执行 7 条指令。（不包括 halt）

除此之外，指令执行总条数  $g$  与  $n$  的关系式。如下：

$$g(n) = 7(n-2) + 8$$

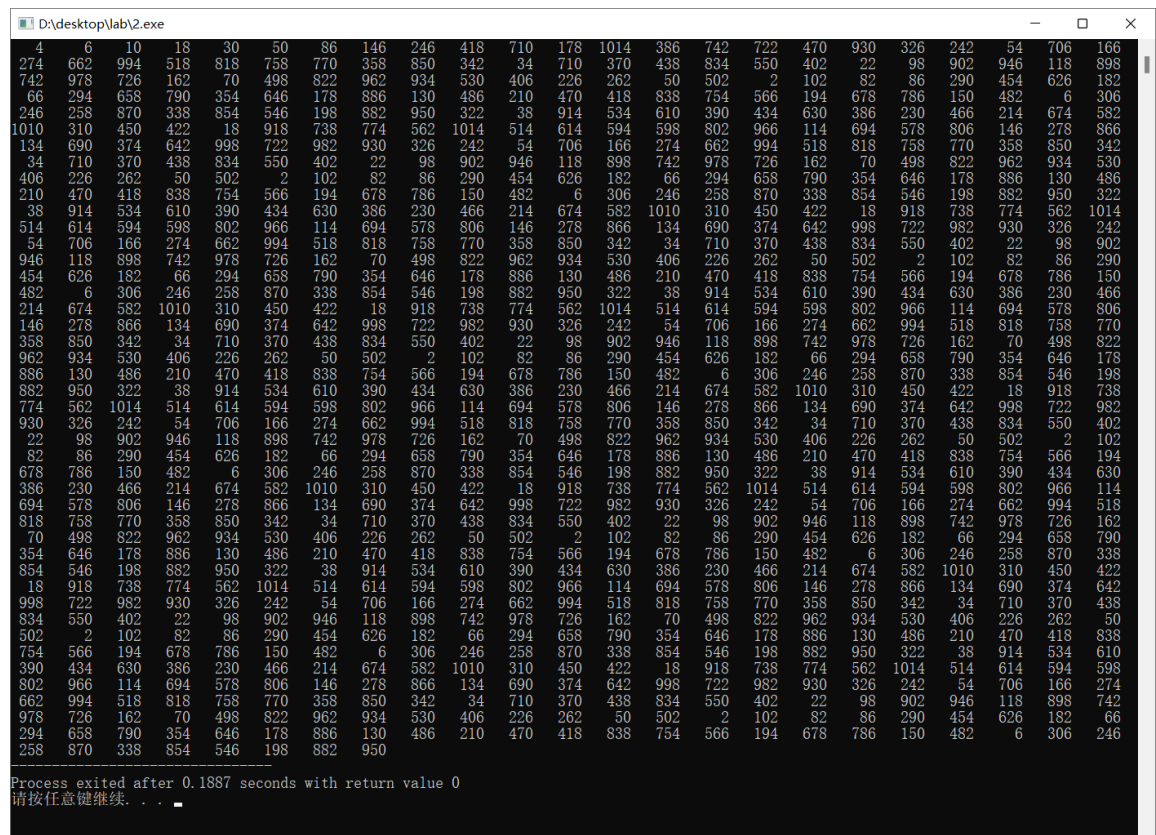
根据对原程序的分析方法，同理可得在 24, 144, 456, 1088, 1092, 2096, 4200, 8192, 12000, 14000 这十个测试数据下的平均指令执行条数为 30298.4

优化比率为： $30298.4 / 38950.8 \approx 77.8\%$

## ②寻找周期规律

由于在计算  $f(n)$  时，我们对结果进行了  $\text{mode } 1024$  的操作。在测试数据时，我也发现了一些  $n$  值不同，但结果相同的数据。因此，我猜想， $f(n)$  的结果应该是一个有周期的数列。

于是，我修改 lab2 中编写的 C 程序，打印了  $3 < n < 1000$  的结果。



随后，我发现，确实存在周期。

从  $n=20$  开始， $f(n)$  为一个周期为 128 的周期数列。

930 326 242 54 706 166 274 662 994 518 818 758 770 358 850 342 34 710  
370 438 834 550 402 22 98 902 946 118 898 742 978 726 162 70 498 822  
962 934 530 406 226 262 50 502 2 102 82 86 290 454 626 182 66 294 658  
790 354 646 178 886 130 486 210 470 418 838 754 566 194 678 786 150  
482 6 306 246 258 870 338 854 546 198 882 950 322 38 914 534 610 390  
434 630 386 230 466 214 674 582 1010 310 450 422 18 918 738 774 562  
1014 514 614 594 598 802 966 114 694 578 806 146 278 866 134 690 374  
642 998 722 982

因此，我们可以利用周期性。对于  $n \geq 20$  的数，先对其进行减 20， $\text{mode } 128$ ，再加 20 的操作。对于  $n < 20$  的数，先不做操作。

然后，可以利用优化方案①中的程序求解。

程序如下：

```

.ORG x3000
    ADD R1, R1, #1
    ADD R2, R2, #1
    ADD R3, R3, #2
    LD R5, VALUE
    ADD R6, R0, R5
    BRn NEXT
    ADD R0, R0, R5
    LD R6, MOD1
    AND R0, R0, R6
    LD R6, VALUE2
    ADD R0, R0, R6
NEXT    LD R5, MOD
    ADD R7, R7, R0
    ADD R0, R0, #-2
    BRnz STOP
AGAIN2  ADD R7, R3, R1
    ADD R7, R7, R1
    ADD R1, R2, #0
    ADD R2, R3, #0
    ADD R3, R7, #0
    ADD R0, R0, #-1
    BRp AGAIN2
    AND R7, R7, R5
STOP    HALT
VALUE   .FILL #-20
VALUE2  .FILL #20
MOD1    .FILL #127
MOD     .FILL #1023
Fa      .FILL #930
Fb      .FILL #1
Fc      .FILL #1
Fd      .FILL #306
.END

```

(指令执行条数分析方法同上)

测试数据:

f(24)=706      由于 24>20, 需要进行减 20, mode 128, 再加 20 的操作. 之后与优化方案①  
中的执行流程一样。因此执行指令条数为: 162+8=170

指令数: 170

f(144)=642      同理得: 指令数: 1010

f(456)=66      同理得: 指令数: 506

f(1088)=2      同理得: 指令数: 450

f(1092)=290      同理得: 指令数: 478

f(2096)=898      同理得: 指令数: 338

f(4200)=322      同理得: 指令数: 730

f(8192)=514      同理得: 指令数: 898

f(12000)=258      同理得: 指令数: 674

f(14000)=898      同理得: 指令数: 338

平均指令条数为: 559.2

优化比率为:  $559.2/38950.8 \approx 1.4\%$

总结: 通过最终方案, 得到优化比率为  $559.2/38950.8 \approx 1.4\%$

