南开大学
Nankai University

南　　开　　大　　学

计 算 机 学 院

预习报告

丁屹

年级：2020 级

专业：计算机科学与技术

2022 年 9 月 19 日

目录

一、 作业 1

对于以下两种代码，选择 LOOP #2 可以得到更快的执行速度。

```
/* LOOP #1 */
int i;
for (i = 0; i < N; ++i) {
  a[i] *= 2000;
  a[i] /= 10000;
}
```

```
/* LOOP #2 */
int* b = a;
int  i;
for (i = 0; i < N; ++i) {
  *b *= 2000;
  *b /= 10000;
  ++b;
}
```

在 x86_64 CPU，Arch Linux 系统使用 GCC 12.2.0 编译器测试，其中 N 取值 1000000000，取 4 次用时的平均值，计时器使用 clock_gettime(CLOCK_MONOTONIC, &ts)，代码位于 src/loop1.c 与 src/loop2.c。

如果不打开-O 优化，可以得到 LOOP #1 平均用时 5049361472 ns，得到 LOOP #2 平均用时 2849261662 ns。LOOP #2 速度更快。

```
a:
  .zero   4000000000
main:
  push    rbp
  mov     rbp, rsp
  mov     DWORD PTR [rbp-4], 0
  jmp     .L2
.L3:
  mov     eax, DWORD PTR [rbp-4]
  cdqe
  mov     eax, DWORD PTR a[0+rax*4]
  imul    edx, eax, 2000
  mov     eax, DWORD PTR [rbp-4]
  cdqe
  mov     DWORD PTR a[0+rax*4], edx
  mov     eax, DWORD PTR [rbp-4]
  cdqe
```

```
    mov     eax, DWORD PTR a[0+rax*4]
    movsx   rdx, eax
    imul    rdx, rdx, 1759218605
    shr     rdx, 32
    sar     edx, 12
    sar     eax, 31
    sub     edx, eax
    mov     eax, DWORD PTR [rbp-4]
    cdqe
    mov     DWORD PTR a[0+rax*4], edx
    add     DWORD PTR [rbp-4], 1
.L2:
    cmp     DWORD PTR [rbp-4], 999999999
    jle     .L3
    mov     eax, 0
    pop     rbp
    ret
```

```
a:
    .zero   4000000000
main:
    push    rbp
    mov     rbp, rsp
    mov     QWORD PTR [rbp-8], OFFSET FLAT:a
    mov     DWORD PTR [rbp-12], 0
    jmp     .L2
.L3:
    mov     rax, QWORD PTR [rbp-8]
    mov     eax, DWORD PTR [rax]
    imul    edx, eax, 2000
    mov     rax, QWORD PTR [rbp-8]
    mov     DWORD PTR [rax], edx
    mov     rax, QWORD PTR [rbp-8]
    mov     eax, DWORD PTR [rax]
    movsx   rdx, eax
    imul    rdx, rdx, 1759218605
    shr     rdx, 32
    sar     edx, 12
    sar     eax, 31
    sub     edx, eax
    mov     rax, QWORD PTR [rbp-8]
    mov     DWORD PTR [rax], edx
    add     QWORD PTR [rbp-8], 4
    add     DWORD PTR [rbp-12], 1
```

```
.L2:
    cmp     DWORD PTR [rbp-12], 999999999
    jle     .L3
    mov     eax, 0
    pop     rbp
    ret
```

比较二者的汇编代码可以发现主要的区别在于：LOOP #1 在每次访问数组时都会计算 0+rax*4，做下标转换；而 LOOP #2 中每次只对指针 [rbp-8]+4，计算量更小。

如果开启-O2 优化，可以得到 LOOP #1 平均用时 1652242168 ns，得到 LOOP #2 平均用时 1664580829 ns。可以认为 LOOP #1 与 LOOP #2 没有性能差距。此时二者汇编代码没有区别，优化为使用 SIMD 超标量技术加速。

二、 作业 2

分词

- Model (field)
- = (boolean operator)
- ” (string begin)
- Civic (string content)
- ” (string end)
- AND (boolean operator)
- Year (field)
- = (boolean operator)
- ” (string begin)
- 2001 (string content)
- ” (string end)

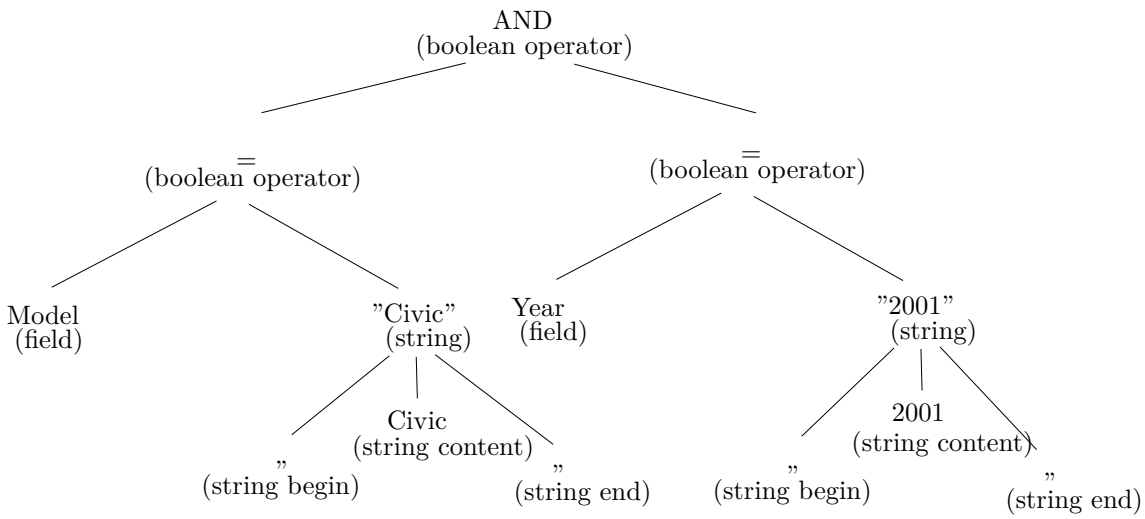

图 1: 语法分析树

三、 作业 3

使用 splint 可以得到如下输出

```
Splint 3.1.2a --- May 25 2020

src/static-check.c: (in function f)
src/static-check.c:13:10: Stack-allocated storage &loc reachable from return
                            value: &loc
  A stack reference is pointed to by an external reference when the function
  returns. The stack-allocated storage is destroyed after the call, leaving a
  dangling reference. (Use -stackref to inhibit warning)
src/static-check.c:13:10: Immediate address &loc returned as implicitly only:
                            &loc
  An immediate address (result of & operator) is transferred inconsistently.
  (Use -immediatetrans to inhibit warning)
src/static-check.c:13:15: Stack-allocated storage *x reachable from parameter x
   src/static-check.c:12:3: Storage *x becomes stack-allocated storage
src/static-check.c:13:15: Function returns with global glob referencing
                            released storage
  A global variable does not satisfy its annotations when control is
  transferred. (Use -globstate to inhibit warning)
   src/static-check.c:13:10: Storage glob released
src/static-check.c: (in function h)
src/static-check.c:18:7: Comparison of unsigned value involving zero: i >= 0
  An unsigned value is used in a comparison with zero in a way that is either a
  bug or confusing. (Use -unsignedcompare to inhibit warning)
src/static-check.c:18:7: Variable i used before definition
  An rvalue is used that may not be initialized to a value on some execution
```

```
  path. (Use -usedef to inhibit warning)
src/static-check.c:7:6: Variable exported but not used outside static-check:
                            glob
  A declaration is exported, but not used outside this module. Declaration can
  use static qualifier. (Use -exportlocal to inhibit warning)

Finished checking --- 7 code warnings
```

- 在函数 f 内，返回了指向局部变量 loc 的指针，可能会导致释放后使用
- 在函数 f 内，传入的参数 x 指向的指针 *x 被写入了局部变量 loc 的地址，函数返回后访问 **x 会导致释放后使用
- 在函数 h 内，未初始化变量 i 就访问其值
- 在函数 h 内，变量 i 类型是无符号整数，只能走 $i \geq 0$ 分支，另一分支无意义
- 在函数 firstChar1 内，没有对传入指针做检查，可能会导致解引用 NULL（此处 splint 不认为有问题）

四、 作业 4

1. int a; 是变量声明
2. int a; 中 a 是标识符列表
3. 若 a 是标识符列表，则 a, b 也是标识符列表