Московский авиационный институт
(национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №3 по курсу «Дискретный анализ»

Студент: Э. Л. Носов
Преподаватель: А. А. Кухтичев
Группа: М8О-307Б
Дата:
Оценка:
Подпись:

Москва, 2021

Лабораторная работа №3

Задача: Для реализации словаря из предыдущей лабораторной работы необходимо провести исследование скорости выполнения и потребления оперативной памяти. В случае выявления ошибок или явных недочётов, требуется их исправить.

Средства диагностики: valgrind и gprof.

1 gprof

gprof - средство анализа производительности Unix приложений, способное измерять абсолютное и относительное время работы приложения, а также строить граф вызовов функций. Для использования gprof необходимо скомпилировать программу с ключом -pg, после чего запустить программу без gprof, чтобы создать файл gmon.out, после чего запустить программу с gprof для получения результата.

```
maloletniydebil@LAPTOP-LNCHG0M3:/mnt/d/X-Files/MAI/3 sem/DA/lab2$ cat test | ./a.out > /dev/null
maloletniydebil@LAPTOP-LNCHG0M3:/mnt/d/X-Files/MAI/3 sem/DA/lab2$ gprof ./a.out ./gmon.out -p
Flat profile:

Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  us/call  us/call  name
 22.70      0.51     0.51  1194930     0.43     0.43  TString::operator=(TString const&)
 15.13      0.85     0.34   400000     0.85     1.79  TPatricia::DeleteNode(TString&)
 13.35      1.15     0.30  1800000     0.17     0.17  TString::Updatelength()
 13.35      1.45     0.30   399999     0.75     1.34  TPatricia::InsertNode(TPatricia::TNode&, TPatricia
 11.13      1.70     0.25   200000     1.25     1.69  TPatricia::NodeSearch(TPatricia::TNode&, TString&,
  8.90      1.90     0.20   600000     0.33     0.33  TString::operator==(TString const&) const
  7.57      2.07     0.17  1000000     0.17     0.17  TString::ToLower()
  3.12      2.14     0.07 30107817     0.00     0.00  TString::operator[](int)
  2.67      2.20     0.06 25777810     0.00     0.00  TPatricia::GetBit(int, int, TString&)
  1.11      2.23     0.03 29601721     0.00     0.00  TString::Size()
  0.89      2.25     0.02   400000     0.05     1.82  TPatricia::AddNode(TString&, unsigned long long)
  0.22      2.25     0.01   797466     0.01     0.01  TString::~TString()
  0.00      2.25     0.00  1800001     0.00     0.17  operator>>(std::istream&, TString&)
  0.00      2.25     0.00  1594931     0.00     0.00  TString::clear()
  0.00      2.25     0.00   797465     0.00     0.00  TString::TString()
  0.00      2.25     0.00   797465     0.00     0.00  TPatricia::TNode::TNode()
  0.00      2.25     0.00   797465     0.00     0.01  TPatricia::TNode::~TNode()
  0.00      2.25     0.00   399999     0.00     0.06  TPatricia::FindBitDifference(TPatricia::TNode&, TP
  0.00      2.25     0.00   397465     0.00     0.43  TPatricia::TNode::operator=(TPatricia::TNode const
  0.00      2.25     0.00   200000     0.00     1.69  TPatricia::FindNode(TString&)
  0.00      2.25     0.00        1     0.00     0.00  _GLOBAL__sub_I__ZlsRSoRK7TString
  0.00      2.25     0.00        1     0.00     0.00  __static_initialization_and_destruction_0(int, int
  0.00      2.25     0.00        1     0.00     0.00  TString::TString(int)
  0.00      2.25     0.00        1     0.00     0.00  TPatricia::Clear()
  0.00      2.25     0.00        1     0.00     0.00  TPatricia::TPatricia()
  0.00      2.25     0.00        1     0.00     0.00  TPatricia::~TPatricia()

 %         the percentage of the total running time of the
time       program used by this function.

cumulative a running sum of the number of seconds accounted
 seconds   for by this function and those listed above it.

 self      the number of seconds accounted for by this
seconds    function alone.  This is the major sort for this
```

```
           listing.

calls      the number of times this function was invoked, if
           this function is profiled, else blank.

 self      the average number of milliseconds spent in this
ms/call    function per call, if this function is profiled,
           else blank.

 total     the average number of milliseconds spent in this
ms/call    function and its descendents per call, if this
           function is profiled, else blank.

name       the name of the function.  This is the minor sort
           for this listing. The index shows the location of
           the function in the gprof listing. If the index is
           in parenthesis it shows where it would appear in
           the gprof listing if it were to be printed.


Copyright (C) 2012-2020 Free Software Foundation, Inc.

Copying and distribution of this file, with or without modification,
are permitted in any medium without royalty provided the copyright
notice and this notice are preserved.
```

Как видно из отчета: операции со строками занимают почти в два раза больше времени, чем каждая прочая операция в отдельности.

2 valgrind

valgrind - инструмент для исследования ошибок связанных с памятью, выявления утечек памяти и профилирования. Для работы с valgrind необходимо скомпилировать программу с ключом -g, после чего запустить её с необходимыми ключами.

```
maloletniydebil@LAPTOP-LNCHG0M3:/mnt/d/X-Files/MAI/3 sem/DA/lab2$ python3 tests.py tests 30
maloletniydebil@LAPTOP-LNCHG0M3:/mnt/d/X-Files/MAI/3 sem/DA/lab2$ valgrind ./a.out < tests
==7434== Memcheck, a memory error detector
==7434== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==7434== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==7434== Command: ./a.out
==7434==
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK: 5271039388302272487
OK: 77107583195168778
OK: 14935627783876654825
OK: 8283430140897024806
OK: 11979618145377135509
OK: 10533444657081922184
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
==7434==
==7434== HEAP SUMMARY:
==7434==     in use at exit: 122,880 bytes in 6 blocks
==7434==   total heap usage: 77 allocs, 71 frees, 200,456 bytes allocated
==7434==
==7434== LEAK SUMMARY:
==7434==    definitely lost: 0 bytes in 0 blocks
```

```
==7434==    indirectly lost: 0 bytes in 0 blocks
==7434==      possibly lost: 0 bytes in 0 blocks
==7434==    still reachable: 122,880 bytes in 6 blocks
==7434==         suppressed: 0 bytes in 0 blocks
==7434== Rerun with --leak-check=full to see details of leaked memory
==7434==
==7434== For lists of detected and suppressed errors, rerun with: -s
```

В выводе valgrind указано, что не все выделенные блоки памяти были освобождены. Добавим ключи –leak-check=full и –show-leak-kinds=all.

```
maloletniydebil@LAPTOP-LNCHG0M3:/mnt/d/X-Files/MAI/3 sem/DA/lab2$ valgrind --leak-check=full --show-leak
==7651== Memcheck, a memory error detector
==7651== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==7651== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==7651== Command: ./a.out
==7651==
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK: 5271039388302272487
OK: 77107583195168778
OK: 14935627783876654825
OK: 8283430140897024806
OK: 11979618145377135509
OK: 10533444657081922184
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
==7651==
==7651== HEAP SUMMARY:
==7651==     in use at exit: 122,880 bytes in 6 blocks
```

```
==7651==   total heap usage: 77 allocs, 71 frees, 200,456 bytes allocated
==7651==
==7651== 8,192 bytes in 1 blocks are still reachable in loss record 1 of 6
==7651==    at 0x483C583: operator new[](unsigned long) (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload
==7651==    by 0x495EF63: std::basic_filebuf<char, std::char_traits<char> >::_M_allocate_internal_buffer
==7651==    by 0x495CD49: ??? (in /usr/lib/x86_64-linux-gnu/libstdc++.so.6.0.28)
==7651==    by 0x490EB47: std::ios_base::sync_with_stdio(bool) (in /usr/lib/x86_64-linux-gnu/libstdc++.s
==7651==    by 0x10A704: main (main.cpp:6)
==7651==
==7651== 8,192 bytes in 1 blocks are still reachable in loss record 2 of 6
==7651==    at 0x483C583: operator new[](unsigned long) (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload
==7651==    by 0x495EF63: std::basic_filebuf<char, std::char_traits<char> >::_M_allocate_internal_buffer
==7651==    by 0x495CD49: ??? (in /usr/lib/x86_64-linux-gnu/libstdc++.so.6.0.28)
==7651==    by 0x490EB68: std::ios_base::sync_with_stdio(bool) (in /usr/lib/x86_64-linux-gnu/libstdc++.s
==7651==    by 0x10A704: main (main.cpp:6)
==7651==
==7651== 8,192 bytes in 1 blocks are still reachable in loss record 3 of 6
==7651==    at 0x483C583: operator new[](unsigned long) (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload
==7651==    by 0x495EF63: std::basic_filebuf<char, std::char_traits<char> >::_M_allocate_internal_buffer
==7651==    by 0x495CD49: ??? (in /usr/lib/x86_64-linux-gnu/libstdc++.so.6.0.28)
==7651==    by 0x490EB88: std::ios_base::sync_with_stdio(bool) (in /usr/lib/x86_64-linux-gnu/libstdc++.s
==7651==    by 0x10A704: main (main.cpp:6)
==7651==
==7651== 32,768 bytes in 1 blocks are still reachable in loss record 4 of 6
==7651==    at 0x483C583: operator new[](unsigned long) (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload
==7651==    by 0x4960D76: std::basic_filebuf<wchar_t, std::char_traits<wchar_t> >::_M_allocate_internal_
==7651==    by 0x495CF39: ??? (in /usr/lib/x86_64-linux-gnu/libstdc++.so.6.0.28)
==7651==    by 0x490EBFD: std::ios_base::sync_with_stdio(bool) (in /usr/lib/x86_64-linux-gnu/libstdc++.s
==7651==    by 0x10A704: main (main.cpp:6)
==7651==
==7651== 32,768 bytes in 1 blocks are still reachable in loss record 5 of 6
==7651==    at 0x483C583: operator new[](unsigned long) (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload
==7651==    by 0x4960D76: std::basic_filebuf<wchar_t, std::char_traits<wchar_t> >::_M_allocate_internal_
==7651==    by 0x495CF39: ??? (in /usr/lib/x86_64-linux-gnu/libstdc++.so.6.0.28)
==7651==    by 0x490EC17: std::ios_base::sync_with_stdio(bool) (in /usr/lib/x86_64-linux-gnu/libstdc++.s
==7651==    by 0x10A704: main (main.cpp:6)
==7651==
==7651== 32,768 bytes in 1 blocks are still reachable in loss record 6 of 6
==7651==    at 0x483C583: operator new[](unsigned long) (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload
==7651==    by 0x4960D76: std::basic_filebuf<wchar_t, std::char_traits<wchar_t> >::_M_allocate_internal_
==7651==    by 0x495CF39: ??? (in /usr/lib/x86_64-linux-gnu/libstdc++.so.6.0.28)
==7651==    by 0x490EC30: std::ios_base::sync_with_stdio(bool) (in /usr/lib/x86_64-linux-gnu/libstdc++.s
==7651==    by 0x10A704: main (main.cpp:6)
==7651==
==7651== LEAK SUMMARY:
==7651==    definitely lost: 0 bytes in 0 blocks
==7651==    indirectly lost: 0 bytes in 0 blocks
==7651==      possibly lost: 0 bytes in 0 blocks
==7651==    still reachable: 122,880 bytes in 6 blocks
```

```
==7651==         suppressed: 0 bytes in 0 blocks
==7651==
==7651== For lists of detected and suppressed errors, rerun with: -s
==7651== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

Теперь видно что причина утечки в отключении синхронизации ввода-вывода.

```
maloletniydebil@LAPTOP-LNCHG0M3:/mnt/d/X-Files/MAI/3 sem/DA/lab2$ valgrind --leak-check=full --show-leak
==7845== Memcheck, a memory error detector
==7845== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==7845== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==7845== Command: ./a.out
==7845==
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK: 17837766491277476217
OK: 14117666739419636288
OK: 15560456999062722185
OK: 14087741311408232739
OK: 17391834696707841405
OK: 17127414465866374802
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
OK
==7845==
==7845== HEAP SUMMARY:
==7845==     in use at exit: 0 bytes in 0 blocks
==7845==   total heap usage: 73 allocs, 73 frees, 83,245 bytes allocated
==7845==
==7845== All heap blocks were freed -- no leaks are possible
==7845==
```

```
==7845== For lists of detected and suppressed errors, rerun with: -s
==7845== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

После включения синхронизации утечек не осталось.

3 Дневник выполнения работы.

21.11.21 21:00 : приступил к выполнению работы,

21.11.21 22:00 : закончил ознакомление с утилитой gprof, увидел, какое количество времени уходит на использованные функции, количество вызовов и процентное соотношение занятого времени,

21.11.21 22:30 : закончил ознакомление с valgrind, выявил утечку в результате отключения синхронизации ввода-вывода, устранил ее путем включения синхронизации,

21.11.21 23:40 : закончил заполнение отчета.

4 Выводы

Выполнив третью лабораторную работу по курсу «Дискретный анализ», я научился использовать таки средства профилирования, как gprof и valgrind.