我们这样做Java Profiling

费辉
花名 成滔
阿里巴巴集团-核心系统研发-专用计算组

关于我

- 工作
  - 2011年7月，中科院软件所毕业后加入淘宝
- 兴趣
  - JVM相关的未知问题
- 微博
  - @呱哥在淘宝 http://weibo.com/u/2651541140

议程

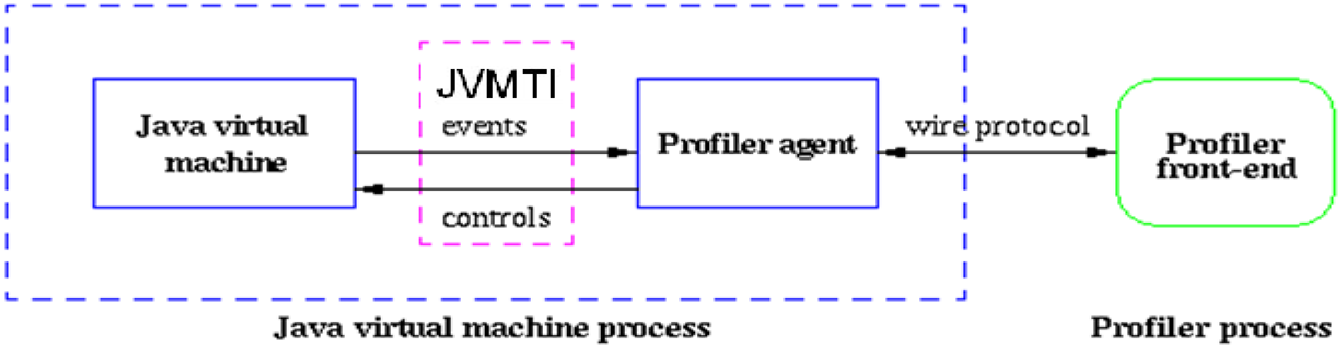
- ✓Java Profiler 工具分析
- ✓使用vtune/perf分析java应用的热点
- ✓vtune/perf分析java应用实例

Java Profiler 工具分析

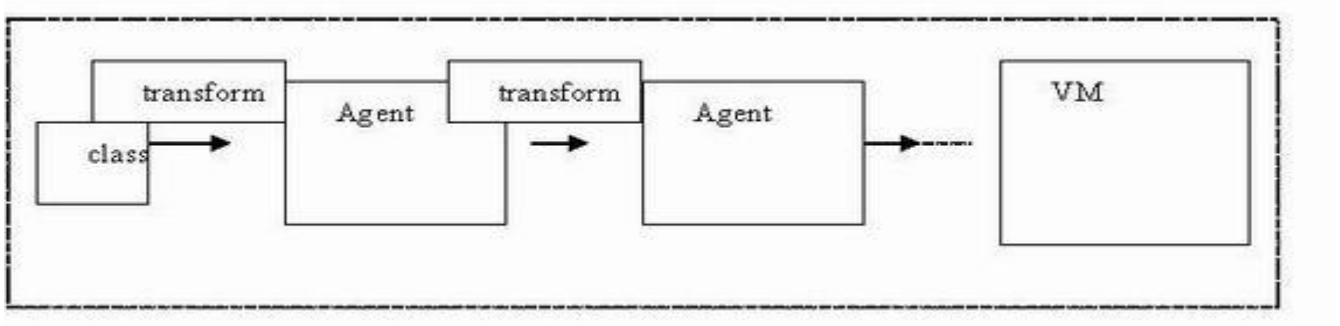
- 商业软件
  - CodePro Profiler
  - YourKit Java Profiler
  - Jprofiler
- 开源软件
  - TPTP（Test and Performance Tools Platform）
  - VisualVM
  - TProfiler

Java Profiler 工具分析

- *JVMTI*



- *Instrument*
  - *修改字节码*



Java Profiler 工具分析

- 缺点
  - 侵入式
  - 开销大
  - 影响应用热点
- 适用范围
  - 开发者粗略分析
  - 不适合分析线上应用

使用vtune/perf分析java应用的热点

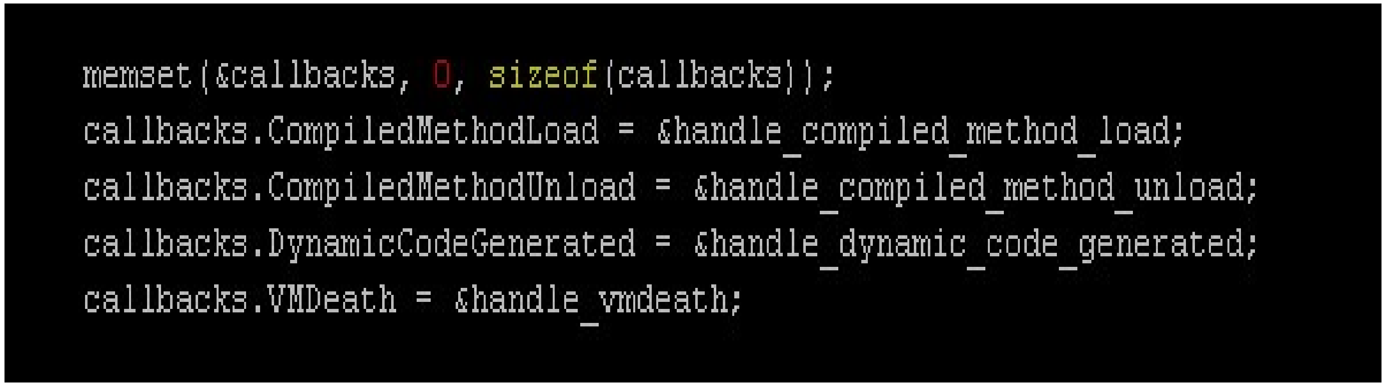
- Java Profiling的系统工具
  - Vtune
  - Perf
  - Oprofile
- Oprofile
  - 对Java Profiling支持不够好
  - profiling结果偏差大
  - http://xiaotaoge.iteye.com/blog/1458654

使用vtune/perf分析java应用的热点

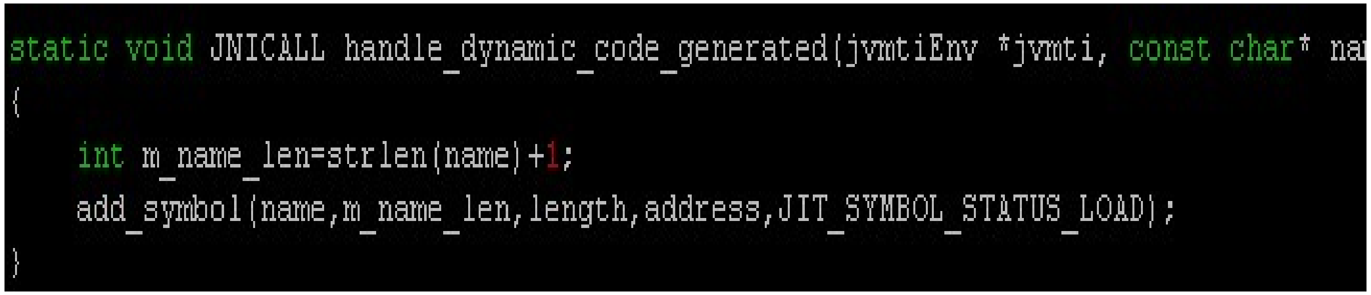
- 原理
  - JVM中的C1/C2编译热点java代码
  - 通过JVMTI暴露方法信息
  - Vtune/Perf通过agent获得编译的方法信息
  - Vtune/Perf内部采样分析
  - 输出结果
  - Java方法可以在结果中看到

使用vtune/perf分析java应用的热点

- Perf Agent



```
memset(&callbacks, 0, sizeof(callbacks));
callbacks.CompiledMethodLoad = &handle_compiled_method_load;
callbacks.CompiledMethodUnload = &handle_compiled_method_unload;
callbacks.DynamicCodeGenerated = &handle_dynamic_code_generated;
callbacks.VMDeath = &handle_vmdeath;
```



```
static void JNICALL handle_dynamic_code_generated(jvmtiEnv *jvmti, const char* na
{
    int m_name_len=strlen(name)+1;
    add_symbol(name,m_name_len,length,address,JIT_SYMBOL_STATUS_LOAD);
}
```

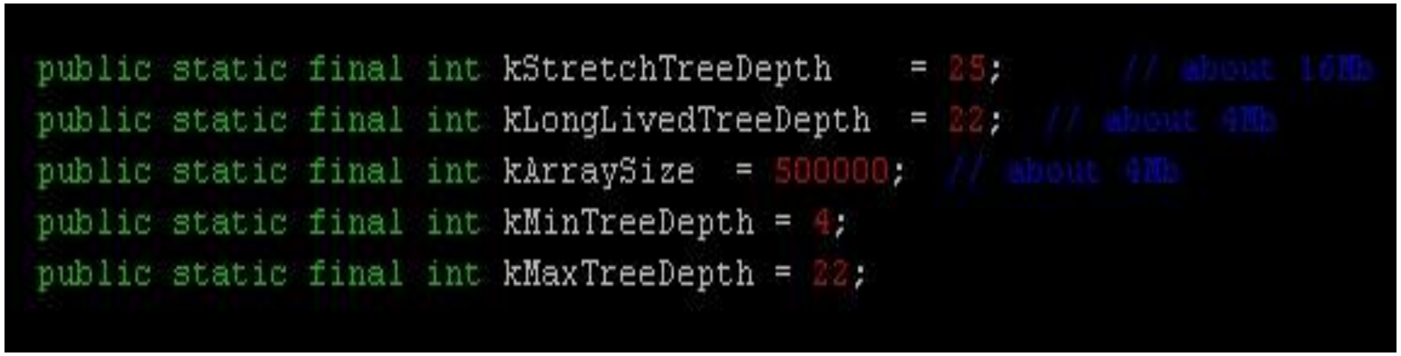
使用vtune/perf分析java应用的热点

- Vtune
  - 商业版
  - 非商业版

- Perf
  - Linux Kernel的一部分
  - Taobao Kernel http://kernel.taobao.org
  - Perf Agent
  - TaobaoJVM http://jvm.taobao.org

vtune/perf分析java应用实例

- 实例
  - GCBench
  - http://www.hpl.hp.com/personal/Hans_Boehm/gc/gc_bench/applet/GCBench.java



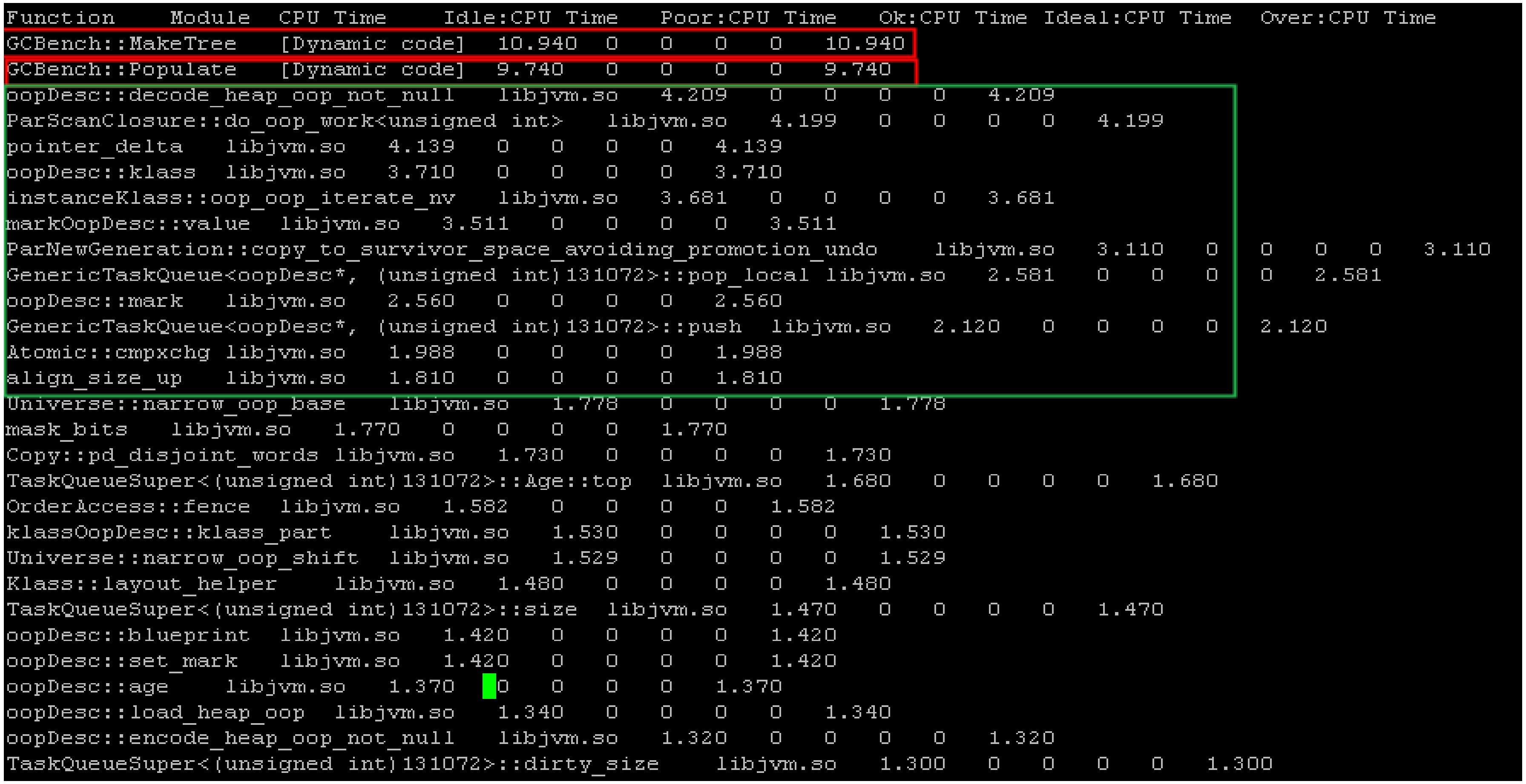
```
public static final int kStretchTreeDepth    = 25;      // about 16Mb
public static final int kLongLivedTreeDepth  = 22;  // about 4Mb
public static final int kArraySize  = 500000;  // about 4Mb
public static final int kMinTreeDepth = 4;
public static final int kMaxTreeDepth = 22;
```

vtune/perf分析java应用实例

- Vtune
  - **环境变量 export AMPLXE_EXPERIMENTAL=1**
  - 运行应用 amplxe-runss –r **test_hot** -interval=**<integer>** -- appname
  - 生成统计数据 amplxe-cl -report **hotspots** -r **test_hot** -report-out **test_out**
  - 图形界面分析数据
  - 参考博文 http://xiaotaoge.iteye.com/blog/1458661

vtune/perf分析java应用实例

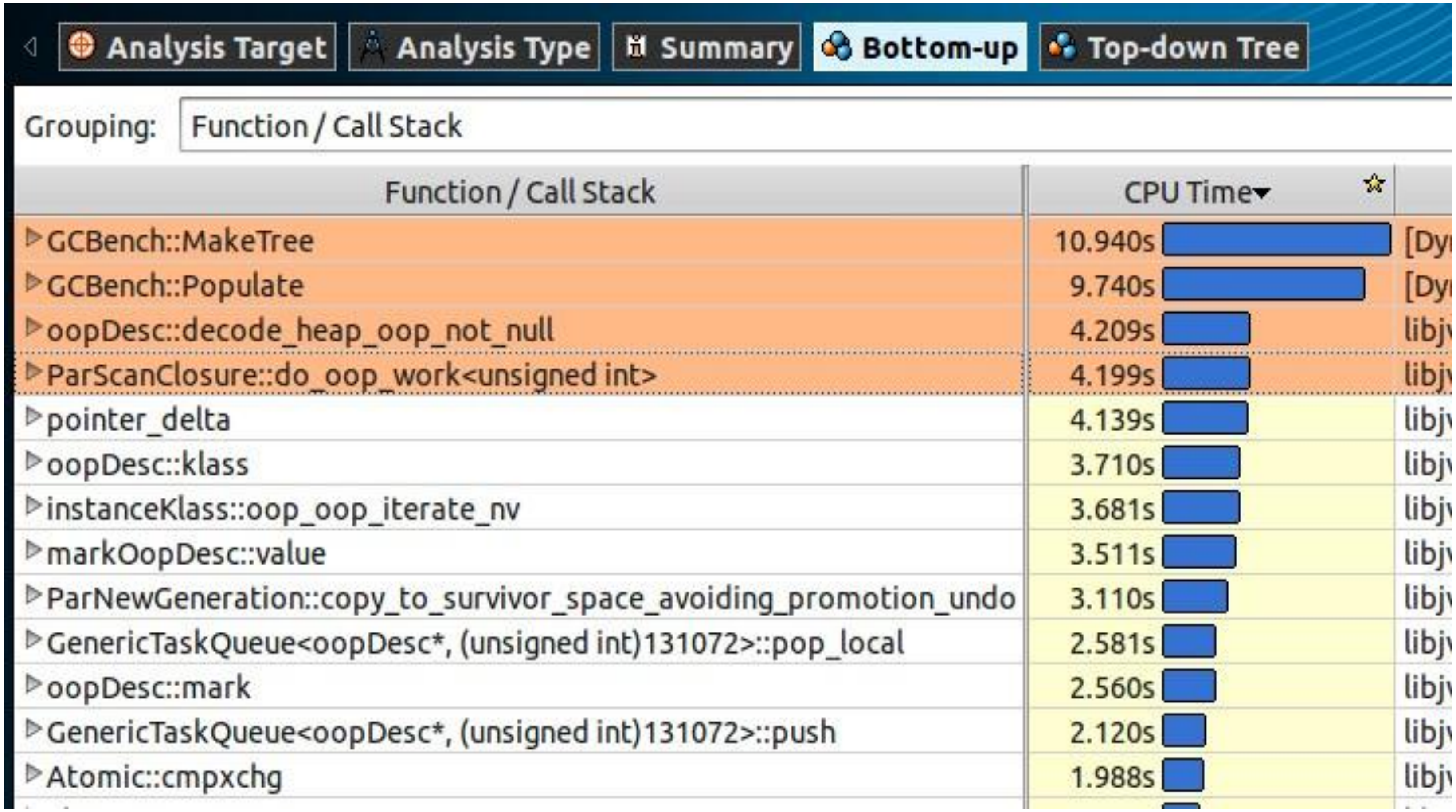
- Vtune text结果



```
Function     Module   CPU Time     Idle:CPU Time    Poor:CPU Time    Ok:CPU Time Ideal:CPU Time  Over:CPU Time
GCBench::MakeTree     [Dynamic code]  10.940  0   0   0   0   10.940
GCBench::Populate     [Dynamic code]  9.740   0   0   0   0   9.740
oopDesc::decode_heap_oop_not_null     libjvm.so   4.209   0   0   0   0   4.209
ParScanClosure::do_oop_work<unsigned int>    libjvm.so   4.199   0   0   0   0   4.199
pointer_delta    libjvm.so   4.139   0   0   0   0   4.139
oopDesc::klass  libjvm.so   3.710   0   0   0   0   3.710
instanceKlass::oop_oop_iterate_nv    libjvm.so   3.681   0   0   0   0   3.681
markOopDesc::value  libjvm.so   3.511   0   0   0   0   3.511
ParNewGeneration::copy_to_survivor_space_avoiding_promotion_undo     libjvm.so   3.110   0   0   0   0   3.110
GenericTaskQueue<oopDesc*, (unsigned int)131072>::pop_local libjvm.so   2.581   0   0   0   0   2.581
oopDesc::mark   libjvm.so   2.560   0   0   0   0   2.560
GenericTaskQueue<oopDesc*, (unsigned int)131072>::push  libjvm.so   2.120   0   0   0   0   2.120
Atomic::cmpxchg libjvm.so   1.988   0   0   0   0   1.988
align_size_up   libjvm.so   1.810   0   0   0   0   1.810
Universe::narrow_oop_base   libjvm.so   1.778   0   0   0   0   1.778
mask_bits   libjvm.so   1.770   0   0   0   0   1.770
Copy::pd_disjoint_words libjvm.so   1.730   0   0   0   0   1.730
TaskQueueSuper<(unsigned int)131072>::Age::top  libjvm.so   1.680   0   0   0   0   1.680
OrderAccess::fence  libjvm.so   1.582   0   0   0   0   1.582
klassOopDesc::klass_part    libjvm.so   1.530   0   0   0   0   1.530
Universe::narrow_oop_shift  libjvm.so   1.529   0   0   0   0   1.529
Klass::layout_helper    libjvm.so   1.480   0   0   0   0   1.480
TaskQueueSuper<(unsigned int)131072>::size  libjvm.so   1.470   0   0   0   0   1.470
oopDesc::blueprint  libjvm.so   1.420   0   0   0   0   1.420
oopDesc::set_mark   libjvm.so   1.420   0   0   0   0   1.420
oopDesc::age    libjvm.so   1.370   0   0   0   0   1.370
oopDesc::load_heap_oop  libjvm.so   1.340   0   0   0   0   1.340
oopDesc::encode_heap_oop_not_null     libjvm.so   1.320   0   0   0   0   1.320
TaskQueueSuper<(unsigned int)131072>::dirty_size     libjvm.so   1.300   0   0   0   0   1.300
```

vtune/perf分析java应用实例

- Vtune 可视化结果

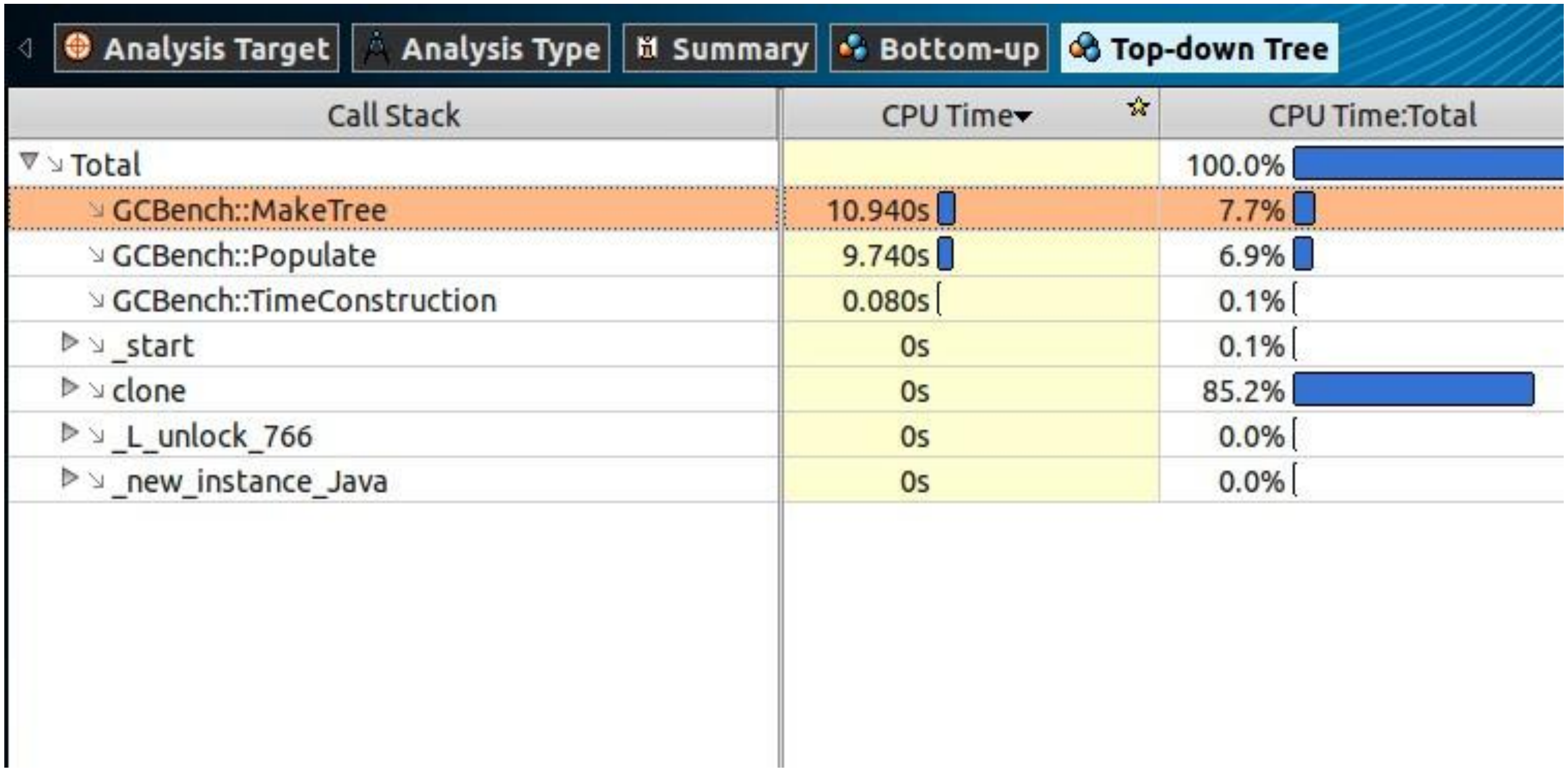


Grouping: Function / Call Stack

Function / Call Stack	CPU Time	
GCBench::MakeTree	10.940s	[Dy
GCBench::Populate	9.740s	[Dy
oopDesc::decode_heap_oop_not_null	4.209s	libj
ParScanClosure::do_oop_work<unsigned int>	4.199s	libj
pointer_delta	4.139s	libj
oopDesc::klass	3.710s	libj
instanceKlass::oop_oop_iterate_nv	3.681s	libj
markOopDesc::value	3.511s	libj
ParNewGeneration::copy_to_survivor_space_avoiding_promotion_undo	3.110s	libj
GenericTaskQueue<oopDesc*, (unsigned int)131072>::pop_local	2.581s	libj
oopDesc::mark	2.560s	libj
GenericTaskQueue<oopDesc*, (unsigned int)131072>::push	2.120s	libj
Atomic::cmpxchg	1.988s	libj

vtune/perf分析java应用实例

- Vtune 可视化结果



Analysis Target | Analysis Type | Summary | Bottom-up | Top-down Tree

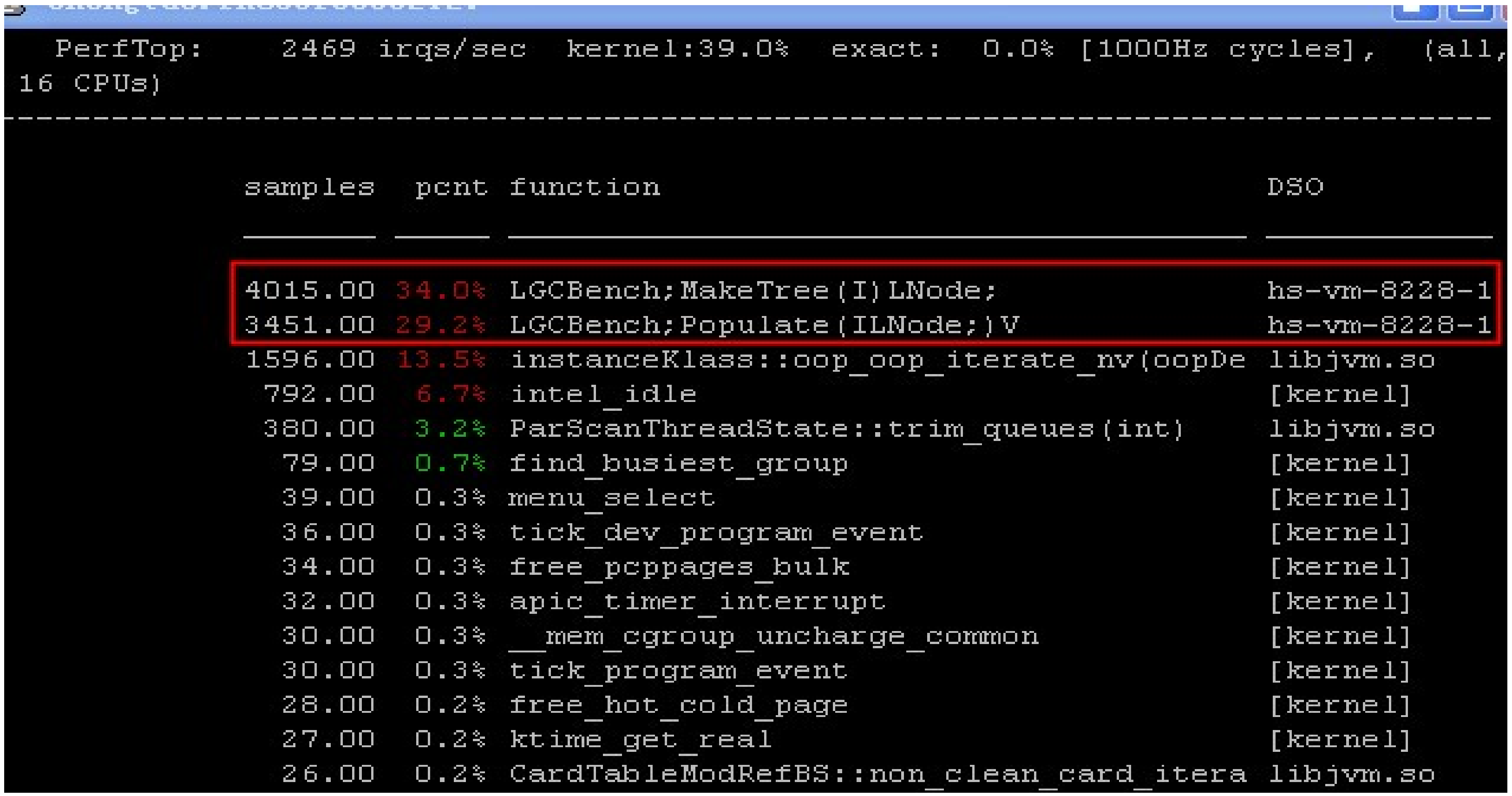
Call Stack	CPU Time	CPU Time:Total
Total		100.0%
GCBench::MakeTree	10.940s	7.7%
GCBench::Populate	9.740s	6.9%
GCBench::TimeConstruction	0.080s	0.1%
_start	0s	0.1%
clone	0s	85.2%
_L_unlock_766	0s	0.0%
_new_instance_Java	0s	0.0%

vtune/perf分析java应用实例

- Perf
  - Java -agentpath:/xxx/libjvmti_perf.so.0.0 -XX:+UseOprofile appname
  - sudo perf top
  - 参考博文 http://xiaotaoge.iteye.com/blog/1648995

vtune/perf分析java应用实例

- Perf



```
  PerfTop:    2469 irqs/sec  kernel:39.0%  exact:  0.0% [1000Hz cycles],  (all,
 16 CPUs)
-------------------------------------------------------------------------------

             samples  pcnt function                                DSO
             _______ _____ _______________________________________ ____________

             4015.00 34.0% LGCBench;MakeTree(I)LNode;               hs-vm-8228-1
             3451.00 29.2% LGCBench;Populate(ILNode;)V              hs-vm-8228-1
             1596.00 13.5% instanceKlass::oop_oop_iterate_nv(oopDe libjvm.so
              792.00  6.7% intel_idle                              [kernel]
              380.00  3.2% ParScanThreadState::trim_queues(int)    libjvm.so
               79.00  0.7% find_busiest_group                      [kernel]
               39.00  0.3% menu_select                             [kernel]
               36.00  0.3% tick_dev_program_event                  [kernel]
               34.00  0.3% free_pcppages_bulk                      [kernel]
               32.00  0.3% apic_timer_interrupt                    [kernel]
               30.00  0.3% __mem_cgroup_uncharge_common            [kernel]
               30.00  0.3% tick_program_event                      [kernel]
               28.00  0.2% free_hot_cold_page                      [kernel]
               27.00  0.2% ktime_get_real                          [kernel]
               26.00  0.2% CardTableModRefBS::non_clean_card_itera libjvm.so
```

vtune/perf分析java应用实例

- Hadoop rpc
  - org.apache.hadoop.io.UTF8.writeChars(java.io.DataOutput,java.lang.String,int,int)
  - 找到utf8转码热点，作intrinsic，提高rpc qps
- HSF
  - java.lang. getStackTrace();
  - 促进应用修改代码，考虑重新设计

参考资料

- http://www.ibm.com/developerworks/cn/java/j-lo-profiling/
- https://github.com/taobao/TProfiler
- http://openjdk.java.net/

Any Question？

谢谢！