https://liuxiaofei.com.cn/blog/hotspot-jit%e7%bc%96%e8%af%91c1/
Hotspot JIT编译(C1)

Content:

上一篇文章介绍了JVM怎么用template interpreter来完成字节码到机器码的映射,但是频繁的取指,译指,然后执行,性能会有所损耗.
所以hotspot对频繁执行的方法进行编译优化能大大缩短代码执行时间.
这儿以编译器C1来说明JIT编译,过程主要包括生成BlockList,生成HIR,发射LIR,寄存器分配,生成机器码,代码安装.
对应的代码位置为jdk8\hotspot\src\share\vm\c1.

1 字节码
以String.indexOf(int,int)方法的编译为例:

```
//LineNo  SourceCode
1545      public int indexOf(int ch, int fromIndex) {
1546          final int max = value.length;
1547          if (fromIndex < 0) {
1548              fromIndex = 0;
1549          } else if (fromIndex >= max) {
1550              // Note: fromIndex might be near -1>>>1.
1551              return -1;
1552          }
1553
1554          if (ch < Character.MIN_SUPPLEMENTARY_CODE_POINT) {
1555              // handle most cases here (ch is a BMP code point or a
1556              // negative value (invalid code point))
1557              final char[] value = this.value;
1558              for (int i = fromIndex; i < max; i++) {
1559                  if (value[i] == ch) {
1560                      return i;
1561                  }
1562              }
1563              return -1;
1564          } else {
1565              return indexOfSupplementary(ch, fromIndex);
1566          }
1567      }

//BCI(Byte Code Index): ByteCode
 0: aload_0
 1: getfield        #423 // Field value:[C
 4: arraylength
 5: istore_3
 6: iload_2
 7: ifge            15
10: iconst_0
11: istore_2
12: goto            22
15: iload_2
16: iload_3
17: if_icmplt       22
20: iconst_m1
21: ireturn
22: iload_1
23: ldc             #4   // int 65536
25: if_icmpge       63
28: aload_0
29: getfield        #423 // Field value:[C
32: astore          4
34: iload_2
35: istore          5
37: iload           5
39: iload_3
40: if_icmpge       61
43: aload           4
45: iload           5
47: caload
48: iload_1
49: if_icmpne       55
52: iload           5
54: ireturn
55: iinc            5, 1
58: goto            37
61: iconst_m1
62: ireturn
63: aload_0
64: iload_1
65: iload_2
66: invokespecial   #465 // Method indexOfSupplementary:(II)I
69: ireturn
```

2 Block List生成

```
Call stack to build block list:

int Compilation::compile_java_method()
    void Compilation::build_hir()
        IR::IR(Compilation* compilation, ciMethod* method, int osr_bci)
            IRScope::IRScope(Compilation* compilation, IRScope* caller, int caller_bci, ciMethod* method, int osr_bci, bool create_graph)
                BlockBegin* IRScope::build_graph(Compilation* compilation, int osr_bci)
                    GraphBuilder::GraphBuilder(Compilation* compilation, IRScope* scope)
                        BlockListBuilder::BlockListBuilder(Compilation* compilation, IRScope* scope, int osr_bci)
                            BlockListBuilder::set_leaders()

//代码文件c1_GraphBuilder.cpp
//通过此方法对所有的字节码进行分析,然后划分并生成Block,从下面代码可以看到所有的跳转命令,比较命令,switch语句会生成新的block.
void BlockListBuilder::set_leaders() {
  bool has_xhandlers = xhandlers()->has_handlers();
  BlockBegin* current = NULL;

  // The information which bci starts a new block simplifies the analysis
  // Without it, backward branches could jump to a bci where no block was created
  // during bytecode iteration. This would require the creation of a new block at the
  // branch target and a modification of the successor lists.
  BitMap bci_block_start = method()->bci_block_start();

  ciBytecodeStream s(method());
  while (s.next() != ciBytecodeStream::EOBC()) {
    int cur_bci = s.cur_bci();

    if (bci_block_start.at(cur_bci)) {
      current = make_block_at(cur_bci, current);
    }
    assert(current != NULL, "must have current block");

    if (has_xhandlers && GraphBuilder::can_trap(method(), s.cur_bc())) {
      handle_exceptions(current, cur_bci);
    }

    switch (s.cur_bc()) {
      // track stores to local variables for selective creation of phi functions
      //在桢栈中标记局部变量
      case Bytecodes::_iinc:     store_one(current, s.get_index()); break;
      case Bytecodes::_istore:   store_one(current, s.get_index()); break;
      case Bytecodes::_lstore:   store_two(current, s.get_index()); break;
      case Bytecodes::_fstore:   store_one(current, s.get_index()); break;
      case Bytecodes::_dstore:   store_two(current, s.get_index()); break;
      case Bytecodes::_astore:   store_one(current, s.get_index()); break;
      case Bytecodes::_istore_0: store_one(current, 0); break;
      case Bytecodes::_istore_1: store_one(current, 1); break;
      case Bytecodes::_istore_2: store_one(current, 2); break;
      case Bytecodes::_istore_3: store_one(current, 3); break;
      case Bytecodes::_lstore_0: store_two(current, 0); break;
      case Bytecodes::_lstore_1: store_two(current, 1); break;
      case Bytecodes::_lstore_2: store_two(current, 2); break;
      case Bytecodes::_lstore_3: store_two(current, 3); break;
      case Bytecodes::_fstore_0: store_one(current, 0); break;
      case Bytecodes::_fstore_1: store_one(current, 1); break;
      case Bytecodes::_fstore_2: store_one(current, 2); break;
      case Bytecodes::_fstore_3: store_one(current, 3); break;
      case Bytecodes::_dstore_0: store_two(current, 0); break;
      case Bytecodes::_dstore_1: store_two(current, 1); break;
      case Bytecodes::_dstore_2: store_two(current, 2); break;
      case Bytecodes::_dstore_3: store_two(current, 3); break;
      case Bytecodes::_astore_0: store_one(current, 0); break;
      case Bytecodes::_astore_1: store_one(current, 1); break;
      case Bytecodes::_astore_2: store_one(current, 2); break;
      case Bytecodes::_astore_3: store_one(current, 3); break;

      // track bytecodes that affect the control flow
      case Bytecodes::_athrow:  // fall through
      case Bytecodes::_ret:     // fall through
      case Bytecodes::_ireturn: // fall through
      case Bytecodes::_lreturn: // fall through
      case Bytecodes::_freturn: // fall through
      case Bytecodes::_dreturn: // fall through
      case Bytecodes::_areturn: // fall through
      case Bytecodes::_return:
        current = NULL;
        break;

      case Bytecodes::_ifeq:      // fall through
      case Bytecodes::_ifne:      // fall through
      case Bytecodes::_iflt:      // fall through
      case Bytecodes::_ifge:      // fall through
      case Bytecodes::_ifgt:      // fall through
      case Bytecodes::_ifle:      // fall through
      case Bytecodes::_if_icmpeq: // fall through
      case Bytecodes::_if_icmpne: // fall through
```

```
        case Bytecodes::_if_icmplt: // fall through
        case Bytecodes::_if_icmpge: // fall through
        case Bytecodes::_if_icmpgt: // fall through
        case Bytecodes::_if_icmple: // fall through
        case Bytecodes::_if_acmpeq: // fall through
        case Bytecodes::_if_acmpne: // fall through
        case Bytecodes::_ifnull:    // fall through
        case Bytecodes::_ifnonnull:
          make_block_at(s.next_bci(), current);//条件成功时,下一个字节码开始下一个block
          make_block_at(s.get_dest(), current);//条件不成功时,跳转到的字节码地方开始下一个block
          current = NULL;
          break;

        case Bytecodes::_goto:
          make_block_at(s.get_dest(), current);
          current = NULL;
          break;

        case Bytecodes::_goto_w:
          make_block_at(s.get_far_dest(), current);
          current = NULL;
          break;

        case Bytecodes::_jsr:
          handle_jsr(current, s.get_dest(), s.next_bci());
          current = NULL;
          break;

        case Bytecodes::_jsr_w:
          handle_jsr(current, s.get_far_dest(), s.next_bci());
          current = NULL;
          break;

        case Bytecodes::_tableswitch: {
          // set block for each case
          Bytecode_tableswitch sw(&s);
          int l = sw.length();
          for (int i = 0; i < l; i++) {
            make_block_at(cur_bci + sw.dest_offset_at(i), current);
          }
          make_block_at(cur_bci + sw.default_offset(), current);
          current = NULL;
          break;
        }

        case Bytecodes::_lookupswitch: {
          // set block for each case
          Bytecode_lookupswitch sw(&s);
          int l = sw.number_of_pairs();
          for (int i = 0; i < l; i++) {
            make_block_at(cur_bci + sw.pair_at(i).offset(), current);
          }
          make_block_at(cur_bci + sw.default_offset(), current);
          current = NULL;
          break;
        }
      }
    }
  }
```

最后,Block List生成如下:

```
  begin_compilation
    name " java.lang.String::indexOf"
    method "virtual jint java.lang.String.indexOf(jint, jint)"
    date 1527264031403
  end_compilation
  begin_cfg
    name "BlockListBuilder virtual jint java.lang.String.indexOf(jint, jint)"
    begin_block
      name "B0" //Block名称
      from_bci 0 //Block开始的字节码索引
      to_bci -1  //Block的结束字节码索引
      predecessors //前置Block
      successors "B1" "B2" //此Block会跳转到的Block名称
      xhandlers
      flags "std"
    end_block
    begin_block
      name "B1"
      from_bci 10
      to_bci -1
      predecessors
      successors "B3"
      xhandlers
      flags
    end_block
```

```
    begin_block
      name "B2"
      from_bci 15
      to_bci -1
      predecessors
      successors "B4" "B3"
      xhandlers
      flags
    end_block
    begin_block
      name "B4"
      from_bci 20
      to_bci -1
      predecessors
      successors
      xhandlers
      flags
    end_block
    begin_block
      name "B3"
      from_bci 22
      to_bci -1
      predecessors
      successors "B5" "B6"
      xhandlers
      flags
    end_block
    begin_block
      name "B5"
      from_bci 28
      to_bci -1
      predecessors
      successors "B7"
      xhandlers
      flags
    end_block
    begin_block
      name "B7"
      from_bci 37
      to_bci -1
      predecessors
      successors "B8" "B9"
      xhandlers
      flags "plh"
    end_block
    begin_block
      name "B8"
      from_bci 43
      to_bci -1
      predecessors
      successors "B10" "B11"
      xhandlers
      flags
    end_block
    begin_block
      name "B10"
      from_bci 52
      to_bci -1
      predecessors
      successors
      xhandlers
      flags
    end_block
    begin_block
      name "B11"
      from_bci 55
      to_bci -1
      predecessors
      successors "B7"
      xhandlers
      flags
    end_block
    begin_block
      name "B9"
      from_bci 61
      to_bci -1
      predecessors
      successors
      xhandlers
      flags
    end_block
    begin_block
      name "B6"
      from_bci 63
      to_bci -1
      predecessors
```

```
    successors
    xhandlers
    flags
  end_block
end_cfg
```

生成高级中间表示(HIR)

Block List生成完后,接下来就需要为每个Block生成HIR了.

Call stack to create HIR:

```
int Compilation::compile_java_method()
    void Compilation::build_hir()
        IR::IR(Compilation* compilation, ciMethod* method, int osr_bci)
            IRScope::IRScope(Compilation* compilation, IRScope* caller, int caller_bci, ciMethod* method, int osr_bci, bool create_graph)
                BlockBegin* IRScope::build_graph(Compilation* compilation, int osr_bci)
                    GraphBuilder::GraphBuilder(Compilation* compilation, IRScope* scope)
                        void GraphBuilder::iterate_all_blocks(bool start_in_current_block_for_inlining)
                            void GraphBuilder::connect_to_end(BlockBegin* beg)
                                BlockEnd* GraphBuilder::iterate_bytecodes_for_block(int bci)

//代码文件c1_GraphBuilder.cpp
//循环每个字节码,生成对应的Instruction,所以的Instruction声明都在c1_Instruction.hpp中.
BlockEnd* GraphBuilder::iterate_bytecodes_for_block(int bci) {
......
  _skip_block = false;
  assert(state() != NULL, "ValueStack missing!");
  CompileLog* log = compilation()->log();
  ciBytecodeStream s(method());
  s.reset_to_bci(bci);
  int prev_bci = bci;
  scope_data()->set_stream(&s);
  // iterate
  Bytecodes::Code code = Bytecodes::_illegal;
  bool push_exception = false;

  if (block()->is_set(BlockBegin::exception_entry_flag) && block()->next() == NULL) {
    // first thing in the exception entry block should be the exception object.
    push_exception = true;
  }

  while (!bailed_out() && last()->as_BlockEnd() == NULL &&
         (code = stream()->next()) != ciBytecodeStream::EOBC() &&
         (block_at(s.cur_bci()) == NULL || block_at(s.cur_bci()) == block())) {
......

    // handle bytecode
    switch (code) {
      case Bytecodes::_nop            : /* nothing to do */ break;
......
      case Bytecodes::_tableswitch    : table_switch(); break;
      case Bytecodes::_lookupswitch   : lookup_switch(); break;
      case Bytecodes::_ireturn        : method_return(ipop()); break;
      case Bytecodes::_lreturn        : method_return(lpop()); break;
      case Bytecodes::_freturn        : method_return(fpop()); break;
      case Bytecodes::_dreturn        : method_return(dpop()); break;
      case Bytecodes::_areturn        : method_return(apop()); break;
      case Bytecodes::_return         : method_return(NULL  ); break;
      case Bytecodes::_getstatic      : // fall through
      case Bytecodes::_putstatic      : // fall through
      case Bytecodes::_getfield       : // fall through
      case Bytecodes::_putfield       : access_field(code); break;
      case Bytecodes::_invokevirtual  : // fall through
      case Bytecodes::_invokespecial  : // fall through
      case Bytecodes::_invokestatic   : // fall through
      case Bytecodes::_invokedynamic  : // fall through
      case Bytecodes::_invokeinterface: invoke(code); break;
......
      default                         : ShouldNotReachHere(); break;
    }

    if (log != NULL)
      log->clear_context(); // skip marker if nothing was printed

    // save current bci to setup Goto at the end
    prev_bci = s.cur_bci();

  }
......

  // done
  return end;
}
```

以字节码“ 1: getfield #423 // Field value:[C”为例,

上面的代码就会生成对应的LoadField Instruction(LoadField* load = new LoadField(obj, offset, field, false, state_before, needs_patching)）.

LoadField的定义为:

```
//通过宏定义,为LoadField添加了as_LoadField(), LoadField()和visit(InstructionVisitor)方法.
```

```
//这儿的LoadField.visit方法会调用do_LoadField(this)方法,此方法会在两个类中进行实现.
LEAF(LoadField, AccessField)

BASE(AccessField, Instruction)

#define BASE(class_name, super_class_name)       \
  class class_name: public super_class_name {    \
   public:                                       \
    virtual class_name* as_##class_name()        { return this; }             \


#define LEAF(class_name, super_class_name)       \
  BASE(class_name, super_class_name)             \
   public:                                       \
    virtual const char* name() const             { return #class_name; }      \
    virtual void visit(InstructionVisitor* v)    { v->do_##class_name(this); } \

//代码在c1_InstructionPrinter.cpp中,在打印HIR的指令到output.cfg中时会调用.
void InstructionPrinter::do_LoadField(LoadField* x) {
  print_field(x);
  output()->print(" (%c)", type2char(x->field()->type()->basic_type()));
  output()->print(" %s", x->field()->name()->as_utf8());
}

//代码在c1_LIRGenerator.cpp中,在生成LIR代码时调用.
void LIRGenerator::do_LoadField(LoadField* x) {
  bool needs_patching = x->needs_patching();
  bool is_volatile = x->field()->is_volatile();
  BasicType field_type = x->field_type();

  CodeEmitInfo* info = NULL;
  if (needs_patching) {
    assert(x->explicit_null_check() == NULL, "can't fold null check into patching field access");
    info = state_for(x, x->state_before());
  } else if (x->needs_null_check()) {
    NullCheck* nc = x->explicit_null_check();
    if (nc == NULL) {
      info = state_for(x);
    } else {
      info = state_for(nc);
    }
  }

  LIRItem object(x->obj(), this);

  object.load_item();

#ifndef PRODUCT
  if (PrintNotLoaded && needs_patching) {
    tty->print_cr("   ###class not loaded at load_%s bci %d",
                  x->is_static() ?  "static" : "field", x->printable_bci());
  }
#endif

  bool stress_deopt = StressLoopInvariantCodeMotion && info && info->deoptimize_on_exception();
  if (x->needs_null_check() &&
      (needs_patching ||
       MacroAssembler::needs_explicit_null_check(x->offset()) ||
       stress_deopt)) {
    LIR_Opr obj = object.result();
    if (stress_deopt) {
      obj = new_register(T_OBJECT);
      __ move(LIR_OprFact::oopConst(NULL), obj);
    }
    // emit an explicit null check because the offset is too large
    __ null_check(obj, new CodeEmitInfo(info));
  }

  LIR_Opr reg = rlock_result(x, field_type);
  LIR_Address* address;
  if (needs_patching) {
    // we need to patch the offset in the instruction so don't allow
    // generate_address to try to be smart about emitting the -1.
    // Otherwise the patching code won't know how to find the
    // instruction to patch.
    address = new LIR_Address(object.result(), PATCHED_ADDR, field_type);
  } else {
    address = generate_address(object.result(), x->offset(), field_type);
  }

  if (is_volatile && !needs_patching) {
    volatile_field_load(address, reg, info);
  } else {
    LIR_PatchCode patch_code = needs_patching ? lir_patch_normal : lir_patch_none;
    __ load(address, reg, info, patch_code);
  }
```

```
517
518     if (is_volatile && os::is_MP()) {
519       __ membar_acquire();
520     }
521   }
522   最后,生成的高级中间表示(HIR)为:
523
524   begin_cfg
525     name "After Generation of HIR"
526     begin_block
527       name "B12"
528       from_bci 0
529       to_bci 0
530       predecessors
531       successors "B13"
532       xhandlers
533       flags
534       begin_states
535         begin_locals
536           size 6
537           method "virtual jint java.lang.String.indexOf(jint, jint)"
538    0  a12
539    1  i13
540    2  i14
541         end_locals
542       end_states
543       begin_HIR
544   .0 0  47 std entry B13 <|@
545       end_HIR
546     end_block
547     begin_block
548       name "B13"
549       from_bci 0
550       to_bci 0
551       predecessors "B12"
552       successors "B0"
553       xhandlers
554       flags "std"
555       begin_states
556         begin_locals
557           size 6
558           method "virtual jint java.lang.String.indexOf(jint, jint)"
559    0  a12
560    1  i13
561    2  i14
562         end_locals
563       end_states
564       begin_HIR
565   .0 0  46 goto B0 <|@
566       end_HIR
567     end_block
568     begin_block
569       name "B0"
570       from_bci 0
571       to_bci 7
572       predecessors "B13"
573       successors "B2" "B1"
574       xhandlers
575       flags "std"
576       begin_states
577         begin_locals
578           size 6
579           method "virtual jint java.lang.String.indexOf(jint, jint)"
580   //格式为index value SSA(静态单一赋值,即每个变量只会赋值一次)
581    0  a12
582    1  i13
583    2  i14
584         end_locals
585       end_states
586       begin_HIR
587   //格式为pin.bci usecnt SSA type+id do_LoadField(obj.field type name)
588   .1 0 a15 a12._12 ([) value <|@  // 1: getfield对应的HIR
589   .4 0 i16 a15.length <|@
590   //bci usecnt type+id do_Constant(Int0)
591   7 0 i17 0 <|@
592   //pin.bci usecnt id do_If(if x cond y then block0 else block1)
593   .7 0  18 if i14 >= i17 then B2 else B1 <|@
594       end_HIR
595     end_block
596     begin_block
597       name "B1"
598       from_bci 10
599       to_bci 12
600       predecessors "B0"
601       successors "B3"
602       xhandlers
```

```
      flags
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
    0  a12
    1  i13
    3  i16
        end_locals
      end_states
      begin_HIR
   10 0 i19 0 <|@
   .12 0  20 goto B3 <|@
      end_HIR
    end_block
    begin_block
      name "B3"
      from_bci 22
      to_bci 25
      predecessors "B1" "B2"
      successors "B6" "B5"
      xhandlers
      flags
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
    0  a12
    1  i13
    2  i22 [ i19 i14]
    3  i16
        end_locals
      end_states
      begin_HIR
   23 0 i25 65536 <|@
   .25 0  26 if i13 >= i25 then B6 else B5 <|@
      end_HIR
    end_block
    begin_block
      name "B5"
      from_bci 28
      to_bci 37
      predecessors "B3"
      successors "B7"
      xhandlers
      flags
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
    0  a12
    1  i13
    2  i22
    3  i16
        end_locals
      end_states
      begin_HIR
   .29 0 a27 a12._12 ([) value <|@
   .37 0  28 goto B7 <|@
      end_HIR
    end_block
    begin_block
      name "B7"
      from_bci 37
      to_bci 40
      predecessors "B5" "B11"
      successors "B9" "B8"
      xhandlers
      flags "plh"
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
    1  i13
    3  i16
    4  a27
    5  i29 [ i22 i35]
        end_locals
      end_states
      begin_HIR
   .40 0  30 if i29 >= i16 then B9 else B8 <|@
      end_HIR
    end_block
    begin_block
      name "B8"
      from_bci 43
```

```
      to_bci 49
      predecessors "B7"
      successors "B11" "B10"
      xhandlers
      flags
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   1  i13
   3  i16
   4  a27
   5  i29
        end_locals
      end_states
      begin_HIR
  .47 0 i31 a27[i29] (C) [rc] <|@
  .49 0  32 if i31 != i13 then B11 else B10 <|@
      end_HIR
    end_block
    begin_block
      name "B10"
      from_bci 52
      to_bci 54
      predecessors "B8"
      successors
      xhandlers
      flags
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   5  i29
        end_locals
      end_states
      begin_HIR
  .54 0 i33 ireturn i29 <|@
      end_HIR
    end_block
    begin_block
      name "B11"
      from_bci 55
      to_bci 58
      predecessors "B8"
      successors "B7"
      xhandlers
      flags
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   1  i13
   3  i16
   4  a27
   5  i29
        end_locals
      end_states
      begin_HIR
  55 0 i34 1 <|@
  55 0 i35 i29 + i34 <|@
  .58 0  36 goto B7 (safepoint) <|@
      end_HIR
    end_block
    begin_block
      name "B9"
      from_bci 61
      to_bci 62
      predecessors "B7"
      successors
      xhandlers
      flags
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
        end_locals
      end_states
      begin_HIR
  61 0 i37 -1 <|@
  .62 0 i38 ireturn i37 <|@
      end_HIR
    end_block
    begin_block
      name "B6"
      from_bci 63
      to_bci 69
```

```
      predecessors "B3"
      successors
      xhandlers
      flags
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   0  a12
   1  i13
   2  i22
        end_locals
      end_states
      begin_HIR
  .66 0 a39 null_check(a12) <|@
  .66 0 v41 profile a12 java/lang/String.indexOf) <|@
  .66 0 i42 a12.invokespecial(i13, i22)
                    java/lang/String.indexOfSupplementary(II)I <|@
  .69 0 i43 ireturn i42 <|@
      end_HIR
    end_block
    begin_block
      name "B2"
      from_bci 15
      to_bci 17
      predecessors "B0"
      successors "B3" "B4"
      xhandlers
      flags
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   0  a12
   1  i13
   2  i14
   3  i16
        end_locals
      end_states
      begin_HIR
  .17 0  21 if i14 < i16 then B3 else B4 <|@
      end_HIR
    end_block
    begin_block
      name "B4"
      from_bci 20
      to_bci 21
      predecessors "B2"
      successors
      xhandlers
      flags
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
        end_locals
      end_states
      begin_HIR
  20 0 i23 -1 <|@
  .21 0 i24 ireturn i23 <|@
      end_HIR
    end_block
  end_cfg
  begin_cfg
    name "Before RangeCheckElimination"
    begin_block
      name "B12"
      from_bci 0
      to_bci 0
      predecessors
      successors "B13"
      xhandlers
      flags
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   0  a12
   1  i13
   2  i14
        end_locals
      end_states
      begin_HIR
  .0 0  47 std entry B13 <|@
      end_HIR
    end_block
```

```
    begin_block
      name "B13"
      from_bci 0
      to_bci 0
      predecessors "B12"
      successors "B0"
      xhandlers
      flags "std"
      dominator "B12"
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   0  a12
   1  i13
   2  i14
        end_locals
      end_states
      begin_HIR
  .0 0  46 goto B0 <|@
      end_HIR
    end_block
    begin_block
      name "B0"
      from_bci 0
      to_bci 7
      predecessors "B13"
      successors "B2" "B1"
      xhandlers
      flags "std"
      dominator "B13"
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   0  a12
   1  i13
   2  i14
        end_locals
      end_states
      begin_HIR
  .1 0 a15 a12._12 ([) value <|@
  .4 0 i16 a15.length <|@
  7 0 i17 0 <|@
  .7 0  18 if i14 >= i17 then B2 else B1 <|@
      end_HIR
    end_block
    begin_block
      name "B1"
      from_bci 10
      to_bci 12
      predecessors "B0"
      successors "B3"
      xhandlers
      flags
      dominator "B0"
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   0  a12
   1  i13
   3  i16
        end_locals
      end_states
      begin_HIR
  .12 0  20 goto B3 <|@
      end_HIR
    end_block
    begin_block
      name "B3"
      from_bci 22
      to_bci 25
      predecessors "B1" "B14"
      successors "B6" "B5"
      xhandlers
      flags
      dominator "B0"
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   0  a12
   1  i13
   2  i22 [ i17 i14]
   3  i16
```

```
        end_locals
      end_states
      begin_HIR
   23 0 i25 65536 <|@
   .25 0  26 if i13 >= i25 then B6 else B5 <|@
      end_HIR
    end_block
    begin_block
      name "B5"
      from_bci 28
      to_bci 37
      predecessors "B3"
      successors "B7"
      xhandlers
      flags
      dominator "B3"
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
    0  a12
    1  i13
    2  i22
    3  i16
        end_locals
      end_states
      begin_HIR
   .37 0  28 goto B7 <|@
      end_HIR
    end_block
    begin_block
      name "B7"
      from_bci 37
      to_bci 40
      predecessors "B5" "B11"
      successors "B9" "B8"
      xhandlers
      flags "bb" "plh" "llh"
      dominator "B5"
      loop_index 0
      loop_depth 1
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
    1  i13
    3  i16
    4  a15
    5  i29 [ i22 i35]
        end_locals
      end_states
      begin_HIR
   .40 0  30 if i29 >= i16 then B9 else B8 <|@
      end_HIR
    end_block
    begin_block
      name "B8"
      from_bci 43
      to_bci 49
      predecessors "B7"
      successors "B11" "B10"
      xhandlers
      flags
      dominator "B7"
      loop_index 0
      loop_depth 1
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
    1  i13
    3  i16
    4  a15
    5  i29
        end_locals
      end_states
      begin_HIR
   .47 0 i31 a15[i29] (C) [rc] <|@
   .49 0  32 if i31 != i13 then B11 else B10 <|@
      end_HIR
    end_block
    begin_block
      name "B10"
      from_bci 52
      to_bci 54
      predecessors "B8"
```

```
    successors
    xhandlers
    flags
    dominator "B8"
    begin_states
      begin_locals
        size 6
        method "virtual jint java.lang.String.indexOf(jint, jint)"
 5  i29
      end_locals
    end_states
    begin_HIR
.54 0 i33 ireturn i29 <|@
    end_HIR
  end_block
  begin_block
    name "B11"
    from_bci 55
    to_bci 58
    predecessors "B8"
    successors "B7"
    xhandlers
    flags "lle"
    dominator "B8"
    loop_index 0
    loop_depth 1
    begin_states
      begin_locals
        size 6
        method "virtual jint java.lang.String.indexOf(jint, jint)"
 1  i13
 3  i16
 4  a15
 5  i29
      end_locals
    end_states
    begin_HIR
55 0 i34 1 <|@
55 0 i35 i29 + i34 <|@
.58 0  36 goto B7 (safepoint) <|@
    end_HIR
  end_block
  begin_block
    name "B9"
    from_bci 61
    to_bci 62
    predecessors "B7"
    successors
    xhandlers
    flags
    dominator "B7"
    begin_states
      begin_locals
        size 6
        method "virtual jint java.lang.String.indexOf(jint, jint)"
      end_locals
    end_states
    begin_HIR
61 0 i37 -1 <|@
.62 0 i38 ireturn i37 <|@
    end_HIR
  end_block
  begin_block
    name "B6"
    from_bci 63
    to_bci 69
    predecessors "B3"
    successors
    xhandlers
    flags
    dominator "B3"
    begin_states
      begin_locals
        size 6
        method "virtual jint java.lang.String.indexOf(jint, jint)"
 0  a12
 1  i13
 2  i22
      end_locals
    end_states
    begin_HIR
.66 0 a39 null_check(a12) <|@
.66 0 v41 profile a12 java/lang/String.indexOf) <|@
.66 0 i42 a12.invokespecial(i13, i22)
                    java/lang/String.indexOfSupplementary(II)I <|@
.69 0 i43 ireturn i42 <|@
```

```
      end_HIR
    end_block
    begin_block
      name "B2"
      from_bci 15
      to_bci 17
      predecessors "B0"
      successors "B14" "B4"
      xhandlers
      flags
      dominator "B0"
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   0  a12
   1  i13
   2  i14
   3  i16
        end_locals
      end_states
      begin_HIR
  .17 0  21 if i14 < i16 then B14 else B4 <|@
      end_HIR
    end_block
    begin_block
      name "B4"
      from_bci 20
      to_bci 21
      predecessors "B2"
      successors
      xhandlers
      flags
      dominator "B2"
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
        end_locals
      end_states
      begin_HIR
  20 0 i23 -1 <|@
  .21 0 i24 ireturn i23 <|@
      end_HIR
    end_block
    begin_block
      name "B14"
      from_bci 22
      to_bci 22
      predecessors "B2"
      successors "B3"
      xhandlers
      flags "ces"
      dominator "B2"
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   0  a12
   1  i13
   2  i14
   3  i16
        end_locals
      end_states
      begin_HIR
  .22 0  49 goto B3 <|@
      end_HIR
    end_block
  end_cfg
  生成LIR
  现在,需要根据HIR生成对应的LIR.
  Call stack to create LIR:

  int Compilation::compile_java_method()
      void Compilation::emit_lir()
          void IR::iterate_linear_scan_order(BlockClosure* closure) //closure为LIRGenerator gen(this, method())
              void LIRGenerator::block_do(BlockBegin* block)

  //通过如下的方法,对每个block里的每个instruction调用visit.
  //前面生成生成HIR时,说到过的方法void LIRGenerator::do_LoadField(LoadField* x) 就在此调用并生成LIR.
  void LIRGenerator::block_do(BlockBegin* block) {
    CHECK_BAILOUT();

    block_do_prolog(block);
    set_block(block);

```

```
    for (Instruction* instr = block; instr != NULL; instr = instr->next()) {
      if (instr->is_pinned()) do_root(instr);
    }

    set_block(NULL);
    block_do_epilog(block);
  }

  // This is where the tree-walk starts; instr must be root;
  void LIRGenerator::do_root(Value instr) {
    CHECK_BAILOUT();

    InstructionMark im(compilation(), instr);

    assert(instr->is_pinned(), "use only with roots");
    assert(instr->subst() == instr, "shouldn't have missed substitution");

    instr->visit(this);

    assert(!instr->has_uses() || instr->operand()->is_valid() ||
           instr->as_Constant() != NULL || bailed_out(), "invalid item set");
  }

  //代码在c1_LIR.cpp中
  //do_LoadField方法调用下面方法生成LIR_Op1: lir_move.
  void LIR_List::load(LIR_Address* addr, LIR_Opr src, CodeEmitInfo* info, LIR_PatchCode patch_code) {
    append(new LIR_Op1(
              lir_move,
              LIR_OprFact::address(addr),
              src,
              addr->type(),
              patch_code,
              info));
  }

  //代码在c1_LIR.cpp中
  //此方法会在后面的线性扫描方法中调用,它对每个LIR_Op会做一些标记.
  //比如,lir_move它有输入和输出,所以有do_input和do_output.
  //在线性扫描中,如果有input,那么就会标记对应寄存器的使用区间(Interval）,有output就会定义一个寄存器开始使用的位置.
  void LIR_OpVisitState::visit(LIR_Op* op) {
    // copy information from the LIR_Op
    reset();
    set_op(op);

    switch (op->code()) {

  ......
      case lir_move:           // input and result always valid, may have info
      case lir_pack64:         // input and result always valid
      case lir_unpack64:       // input and result always valid
      case lir_prefetchr:      // input always valid, result and info always invalid
      case lir_prefetchw:      // input always valid, result and info always invalid
      {
        assert(op->as_Op1() != NULL, "must be");
        LIR_Op1* op1 = (LIR_Op1*)op;

        if (op1->_info)                  do_info(op1->_info);
        if (op1->_opr->is_valid())       do_input(op1->_opr);
        if (op1->_result->is_valid())    do_output(op1->_result);

        break;
      }
  ......
  }
  最后,生成的LIR为:

  begin_cfg
    name "Before Register Allocation"
    begin_block
      name "B12"
      from_bci 0
      to_bci 0
      predecessors
      successors "B13"
      xhandlers
      flags
      first_lir_id 0
      last_lir_id 28
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   0  a12 "[R177|L]"
   1  i13 "[R178|I]"
   2  i14 "[R179|I]"
        end_locals
```

```
    end_states
    begin_HIR
.0 0  47 std entry B13 <|@
    end_HIR
    begin_LIR
   0 label [label:0xfc0043b8] <|@
   2 std_entry  <|@
   4 move [rsi|L] [R177|L]  <|@
   6 move [rdx|I] [R178|I]  <|@
   8 move [rcx|I] [R179|I]  <|@
  10 move [metadata:0x3301c100|M] [R180|M]  <|@
  12 move [Base:[R180|M] Disp: 108|I] [R181|I]  <|@
  14 add [R181|I] [int:8|I] [R181|I] <|@
  16 move [R181|I] [Base:[R180|M] Disp: 108|I]  <|@
  18 move [metadata:0x32dd2238|M] [R182|M]  <|@
  20 logic_and [R181|I] [int:8184|I] [R181|I] <|@
  22 cmp [R181|I] [int:0|I]  <|@
  24 branch [EQ] [CounterOverflowStub: 0xf8025908] <|@
  26 label [label:0xf8025930] <|@
  28 branch [AL] [B13]  <|@
    end_LIR
  end_block
  begin_block
    name "B13"
    from_bci 0
    to_bci 0
    predecessors "B12"
    successors "B0"
    xhandlers
    flags "std"
    dominator "B12"
    first_lir_id 30
    last_lir_id 32
    begin_states
      begin_locals
        size 6
        method "virtual jint java.lang.String.indexOf(jint, jint)"
 0  a12 "[R177|L]"
 1  i13 "[R178|I]"
 2  i14 "[R179|I]"
      end_locals
    end_states
    begin_HIR
.0 0  46 goto B0 <|@
    end_HIR
    begin_LIR
  30 label [label:0xfc0046c8] <|@
  32 branch [AL] [B0]  <|@
    end_LIR
  end_block
  begin_block
    name "B0"
    from_bci 0
    to_bci 7
    predecessors "B13"
    successors "B2" "B1"
    xhandlers
    flags "std"
    dominator "B13"
    first_lir_id 34
    last_lir_id 54
    begin_states
      begin_locals
        size 6
        method "virtual jint java.lang.String.indexOf(jint, jint)"
#index value SSA
 0  a12 "[R177|L]"
 1  i13 "[R178|I]"
 2  i14 "[R179|I]"
      end_locals
    end_states
    begin_HIR
#pin.bci usecnt SSA type+id do_LoadField(obj.field type name)
.1 6  "[R183|L]" a15 a12._12 ([) value <|@
.4 11  "[R184|I]" i16 a15.length <|@
#bci usecnt type+id do_Constant(Int0)
7 2 i17 0 <|@
#pin.bci usecnt id do_If(if x cond y then block0 else block1)
.7 0  18 if i14 >= i17 then B2 else B1 <|@
    end_HIR
    begin_LIR
  34 label [label:0xf801ee78] <|@
  36 move [Base:[R177|L] Disp: 12|L] [R183|L]  <|@ //对应的HIR为 .1 6  "[R183|L]" a15 a12._12 ([) value <|@
  38 move [Base:[R183|L] Disp: 12|I] [R184|I]  [bci:4] <|@  //对应的HIR为 .4 11  "[R184|I]" i16 a15.length <|@
  40 cmp [R179|I] [int:0|I]  <|@
  42 move [metadata:0x3301c100|M] [R185|M]  <|@
```

```
    44 cmove [GE] [lng:152|J] [lng:168|J] [R186|J] <|@
    46 move [Base:[R185|M] Index:[R186|J] Disp: 0|J] [R187|J]  <|@
    48 leal [Base:[R187|J] Disp: 1|I] [R187|J]  <|@
    50 move [R187|J] [Base:[R185|M] Index:[R186|J] Disp: 0|J]  <|@
    52 branch [GE] [B2]  <|@
    54 branch [AL] [B1]  <|@
      end_LIR
    end_block
    begin_block
      name "B1"
      from_bci 10
      to_bci 12
      predecessors "B0"
      successors "B3"
      xhandlers
      flags
      dominator "B0"
      first_lir_id 56
      last_lir_id 64
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   0  a12 "[R177|L]"
   1  i13 "[R178|I]"
   3  i16 "[R184|I]"
        end_locals
      end_states
      begin_HIR
  .12 0  20 goto B3 <|@
      end_HIR
      begin_LIR
    56 label [label:0xf801f198] <|@
    58 move [metadata:0x3301c100|M] [R188|M]  <|@
    60 add [Base:[R188|M] Disp: 184|J] [int:1|I] [Base:[R188|M] Disp: 184|J] <|@
    62 move [int:0|I] [R189|I]  <|@
    64 branch [AL] [B3]  <|@
      end_LIR
    end_block
    begin_block
      name "B2"
      from_bci 15
      to_bci 17
      predecessors "B0"
      successors "B14" "B4"
      xhandlers
      flags
      dominator "B0"
      first_lir_id 66
      last_lir_id 82
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   0  a12 "[R177|L]"
   1  i13 "[R178|I]"
   2  i14 "[R179|I]"
   3  i16 "[R184|I]"
        end_locals
      end_states
      begin_HIR
  .17 0  21 if i14 < i16 then B14 else B4 <|@
      end_HIR
      begin_LIR
    66 label [label:0xf801f4b8] <|@
    68 cmp [R179|I] [R184|I]  <|@
    70 move [metadata:0x3301c100|M] [R190|M]  <|@
    72 cmove [LT] [lng:208|J] [lng:224|J] [R191|J] <|@
    74 move [Base:[R190|M] Index:[R191|J] Disp: 0|J] [R192|J]  <|@
    76 leal [Base:[R192|J] Disp: 1|I] [R192|J]  <|@
    78 move [R192|J] [Base:[R190|M] Index:[R191|J] Disp: 0|J]  <|@
    80 branch [LT] [B14]  <|@
    82 branch [AL] [B4]  <|@
      end_LIR
    end_block
    begin_block
      name "B14"
      from_bci 22
      to_bci 22
      predecessors "B2"
      successors "B3"
      xhandlers
      flags "ces"
      dominator "B2"
      first_lir_id 84
      last_lir_id 88
```

```
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
    0  a12 "[R177|L]"
    1  i13 "[R178|I]"
    2  i14 "[R179|I]"
    3  i16 "[R184|I]"
        end_locals
      end_states
      begin_HIR
   .22 0  49 goto B3 <|@
      end_HIR
      begin_LIR
     84 label [label:0xfc0062c8] <|@
     86 move [R179|I] [R189|I]  <|@
     88 branch [AL] [B3]  <|@
      end_LIR
    end_block
    begin_block
      name "B3"
      from_bci 22
      to_bci 25
      predecessors "B1" "B14"
      successors "B6" "B5"
      xhandlers
      flags
      dominator "B0"
      first_lir_id 90
      last_lir_id 106
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
    0  a12 "[R177|L]"
    1  i13 "[R178|I]"
    2  i22 [ i17 i14]  "[R189|I]"
    3  i16 "[R184|I]"
        end_locals
      end_states
      begin_HIR
   23 1 i25 65536 <|@
   .25 0  26 if i13 >= i25 then B6 else B5 <|@
      end_HIR
      begin_LIR
     90 label [label:0xf801f7d8] <|@
     92 cmp [R178|I] [int:65536|I]  <|@
     94 move [metadata:0x3301c100|M] [R193|M]  <|@
     96 cmove [GE] [lng:240|J] [lng:256|J] [R194|J] <|@
     98 move [Base:[R193|M] Index:[R194|J] Disp: 0|J] [R195|J]  <|@
    100 leal [Base:[R195|J] Disp: 1|I] [R195|J]  <|@
    102 move [R195|J] [Base:[R193|M] Index:[R194|J] Disp: 0|J]  <|@
    104 branch [GE] [B6]  <|@
    106 branch [AL] [B5]  <|@
      end_LIR
     end_block
     begin_block
      name "B5"
      from_bci 28
      to_bci 37
      predecessors "B3"
      successors "B7"
      xhandlers
      flags
      dominator "B3"
      first_lir_id 108
      last_lir_id 112
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
    0  a12 "[R177|L]"
    1  i13 "[R178|I]"
    2  i22 "[R189|I]"
    3  i16 "[R184|I]"
        end_locals
      end_states
      begin_HIR
   .37 0  28 goto B7 <|@
      end_HIR
      begin_LIR
    108 label [label:0xf801fe18] <|@
    110 move [R189|I] [R196|I]  <|@
    112 branch [AL] [B7]  <|@
      end_LIR
     end_block
```

```
    begin_block
      name "B7"
      from_bci 37
      to_bci 40
      predecessors "B5" "B11"
      successors "B9" "B8"
      xhandlers
      flags "bb" "plh" "llh"
      dominator "B5"
      loop_index 0
      loop_depth 1
      first_lir_id 114
      last_lir_id 130
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   1  i13 "[R178|I]"
   3  i16 "[R184|I]"
   4  a15 "[R183|L]"
   5  i29 [ i22 i35]  "[R196|I]"
        end_locals
      end_states
      begin_HIR
  .40 0  30 if i29 >= i16 then B9 else B8 <|@
      end_HIR
      begin_LIR
    114 label [label:0xf8020458] <|@
    116 cmp [R196|I] [R184|I]  <|@
    118 move [metadata:0x3301c100|M] [R197|M]  <|@
    120 cmove [GE] [lng:272|J] [lng:288|J] [R198|J] <|@
    122 move [Base:[R197|M] Index:[R198|J] Disp: 0|J] [R199|J]  <|@
    124 leal [Base:[R199|J] Disp: 1|I] [R199|J]  <|@
    126 move [R199|J] [Base:[R197|M] Index:[R198|J] Disp: 0|J]  <|@
    128 branch [GE] [B9]  <|@
    130 branch [AL] [B8]  <|@
      end_LIR
    end_block
    begin_block
      name "B8"
      from_bci 43
      to_bci 49
      predecessors "B7"
      successors "B11" "B10"
      xhandlers
      flags
      dominator "B7"
      loop_index 0
      loop_depth 1
      first_lir_id 132
      last_lir_id 156
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
    1  i13 "[R178|I]"
    3  i16 "[R184|I]"
    4  a15 "[R183|L]"
    5  i29 "[R196|I]"
        end_locals
      end_states
      begin_HIR
  .47 1  "[R201|I]" i31 a15[i29] (C) [rc] <|@
  .49 0  32 if i31 != i13 then B11 else B10 <|@
      end_HIR
      begin_LIR
    132 label [label:0xf8020778] <|@
    134 convert [i21] [R196|I] [R200|J]  <|@
    136 cmp [R196|I] [Base:[R183|L] Disp: 12|I]  <|@
    138 branch [AE] [RangeCheckStub: 0xf80296d8] [bci:47] <|@
    140 move [Base:[R183|L] Index:[R200|J] * 2 Disp: 16|C] [R201|I]  <|@
    142 cmp [R201|I] [R178|I]  <|@
    144 move [metadata:0x3301c100|M] [R202|M]  <|@
    146 cmove [NE] [lng:304|J] [lng:320|J] [R203|J] <|@
    148 move [Base:[R202|M] Index:[R203|J] Disp: 0|J] [R204|J]  <|@
    150 leal [Base:[R204|J] Disp: 1|I] [R204|J]  <|@
    152 move [R204|J] [Base:[R202|M] Index:[R203|J] Disp: 0|J]  <|@
    154 branch [NE] [B11]  <|@
    156 branch [AL] [B10]  <|@
      end_LIR
    end_block
    begin_block
      name "B11"
      from_bci 55
      to_bci 58
      predecessors "B8"
```

```
      successors "B7"
      xhandlers
      flags "lle"
      dominator "B8"
      loop_index 0
      loop_depth 1
      first_lir_id 158
      last_lir_id 190
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   1  i13 "[R178|I]"
   3  i16 "[R184|I]"
   4  a15 "[R183|L]"
   5  i29 "[R196|I]"
        end_locals
      end_states
      begin_HIR
  55 1 i34 1 <|@
  .55 1  "[R205|I]" i35 i29 + i34 <|@
  .58 0  36 goto B7 (safepoint) <|@
      end_HIR
      begin_LIR
   158 label [label:0xf80210d8] <|@
   160 move [R196|I] [R205|I]  <|@
   162 add [R205|I] [int:1|I] [R205|I] <|@
   164 move [metadata:0x3301c100|M] [R206|M]  <|@
   166 move [Base:[R206|M] Disp: 112|I] [R207|I]  <|@
   168 add [R207|I] [int:8|I] [R207|I] <|@
   170 move [R207|I] [Base:[R206|M] Disp: 112|I]  <|@
   172 move [metadata:0x32dd2238|M] [R208|M]  <|@
   174 logic_and [R207|I] [int:65528|I] [R207|I] <|@
   176 cmp [R207|I] [int:0|I]  <|@
   178 branch [EQ] [CounterOverflowStub: 0xf802a488] <|@
   180 label [label:0xf802a4b0] <|@
   182 safepoint    [bci:58] <|@
   184 move [metadata:0x3301c100|M] [R209|M]  <|@
   186 add [Base:[R209|M] Disp: 336|J] [int:1|I] [Base:[R209|M] Disp: 336|J] <|@
   188 move [R205|I] [R196|I]  <|@
   190 branch [AL] [B7]  <|@
      end_LIR
    end_block
    begin_block
      name "B10"
      from_bci 52
      to_bci 54
      predecessors "B8"
      successors
      xhandlers
      flags
      dominator "B8"
      first_lir_id 192
      last_lir_id 196
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
   5  i29 "[R196|I]"
        end_locals
      end_states
      begin_HIR
  .54 0 i33 ireturn i29 <|@
      end_HIR
      begin_LIR
   192 label [label:0xf8020db8] <|@
   194 move [R196|I] [rax|I]  <|@
   196 return [rax|I]   <|@
      end_LIR
    end_block
    begin_block
      name "B9"
      from_bci 61
      to_bci 62
      predecessors "B7"
      successors
      xhandlers
      flags
      dominator "B7"
      first_lir_id 198
      last_lir_id 202
      begin_states
        begin_locals
          size 6
          method "virtual jint java.lang.String.indexOf(jint, jint)"
        end_locals
```

```
    end_states
    begin_HIR
61 1 i37 -1 <|@
.62 0 i38 ireturn i37 <|@
    end_HIR
    begin_LIR
 198 label [label:0xf8020a98] <|@
 200 move [int:-1|I] [rax|I]  <|@
 202 return [rax|I]   <|@
    end_LIR
  end_block
  begin_block
    name "B6"
    from_bci 63
    to_bci 69
    predecessors "B3"
    successors
    xhandlers
    flags
    dominator "B3"
    first_lir_id 204
    last_lir_id 222
    begin_states
      begin_locals
        size 6
        method "virtual jint java.lang.String.indexOf(jint, jint)"
 0  a12 "[R177|L]"
 1  i13 "[R178|I]"
 2  i22 "[R189|I]"
      end_locals
    end_states
    begin_HIR
.66 0 a39 null_check(a12) (eliminated) <|@
.66 0 v41 profile a12 java/lang/String.indexOf) <|@
.66 1  "[R213|I]" i42 a12.invokespecial(i13, i22)
                   java/lang/String.indexOfSupplementary(II)I <|@
.69 0 i43 ireturn i42 <|@
    end_HIR
    begin_LIR
 204 label [label:0xf8020138] <|@
 206 move [R177|L] [R212|L]  <|@
 208 profile_call indexOf.java/lang/String @ 66 [R210|L] [R212|L] [R211|J]  <|@
 210 move [R178|I] [rdx|I]  <|@
 212 move [R189|I] [rcx|I]  <|@
 214 move [R177|L] [rsi|L]  <|@
 216 optvirtual call: [addr: 0x0] [recv: [rsi|L]] [result: [rax|I]] [bci:66] <|@
 218 move [rax|I] [R213|I]  <|@
 220 move [R213|I] [rax|I]  <|@
 222 return [rax|I]   <|@
    end_LIR
  end_block
  begin_block
    name "B4"
    from_bci 20
    to_bci 21
    predecessors "B2"
    successors
    xhandlers
    flags
    dominator "B2"
    first_lir_id 224
    last_lir_id 228
    begin_states
      begin_locals
        size 6
        method "virtual jint java.lang.String.indexOf(jint, jint)"
      end_locals
    end_states
    begin_HIR
20 1 i23 -1 <|@
.21 0 i24 ireturn i23 <|@
    end_HIR
    begin_LIR
 224 label [label:0xf801faf8] <|@
 226 move [int:-1|I] [rax|I]  <|@
 228 return [rax|I]   <|@
    end_LIR
  end_block
end_cfg
```

分配物理寄存器

LIR生成后,因为LIR目前使用的是虚拟寄存器,所以需要通过线性扫描算法分配物理寄存器.

Call stack to create register allocation:

```
int Compilation::compile_java_method()
    void Compilation::emit_lir()
        LinearScan::LinearScan(IR* ir, LIRGenerator* gen, FrameMap* frame_map)
```

```
1807              void LinearScan::do_linear_scan()
1808
1809  //代码在c1_LinearScan.cpp中.
1810  //方法先调用build_intervals()生成所有寄存器的活动区间,然后调用allocate_registers()为所以的虚拟寄存器分配物理寄存器.
1811  void LinearScan::do_linear_scan() {
1812    NOT_PRODUCT(_total_timer.begin_method());
1813
1814    number_instructions();
1815
1816  ......
1817    CHECK_BAILOUT();
1818
1819    build_intervals();
1820    CHECK_BAILOUT();
1821    sort_intervals_before_allocation();
1822
1823    NOT_PRODUCT(print_intervals("Before Register Allocation"));
1824    NOT_PRODUCT(LinearScanStatistic::compute(this, _stat_before_alloc));
1825
1826    allocate_registers();
1827    CHECK_BAILOUT();
1828
1829    resolve_data_flow();
1830  ......
1831  }
1832
1833  //循环每个block的每个instruction,生成每个寄存器的活动区间(interval）
1834  void LinearScan::build_intervals() {
1835    TIME_LINEAR_SCAN(timer_build_intervals);
1836
1837  ......
1838    LIR_OpVisitState visitor;
1839
1840    // iterate all blocks in reverse order
1841    for (i = block_count() - 1; i >= 0; i--) {
1842      BlockBegin* block = block_at(i);
1843      LIR_OpList* instructions = block->lir()->instructions_list();
1844      int         block_from =   block->first_lir_instruction_id();
1845      int         block_to =     block->last_lir_instruction_id();
1846
1847  ......
1848
1849      // iterate all instructions of the block in reverse order.
1850      // skip the first instruction because it is always a label
1851      // definitions of intervals are processed before uses
1852      assert(visitor.no_operands(instructions->at(0)), "first operation must always be a label");
1853      for (int j = instructions->length() - 1; j >= 1; j--) {
1854        LIR_Op* op = instructions->at(j);
1855        int op_id = op->id();
1856
1857  ......
1858
1859        // visit definitions (output and temp operands)
1860        //如果此instruction有output,则生成寄存器使用定义,默认为当前instruction位置到+1.
1861        int k, n;
1862        n = visitor.opr_count(LIR_OpVisitState::outputMode);
1863        for (k = 0; k < n; k++) {
1864          LIR_Opr opr = visitor.opr_at(LIR_OpVisitState::outputMode, k);
1865          assert(opr->is_register(), "visitor should only return register operands");
1866          add_def(opr, op_id, use_kind_of_output_operand(op, opr));
1867        }
1868
1869        n = visitor.opr_count(LIR_OpVisitState::tempMode);
1870        for (k = 0; k < n; k++) {
1871          LIR_Opr opr = visitor.opr_at(LIR_OpVisitState::tempMode, k);
1872          assert(opr->is_register(), "visitor should only return register operands");
1873          add_temp(opr, op_id, mustHaveRegister);
1874        }
1875
1876        // visit uses (input operands)
1877        //如果此instruction有intput,则添加寄存器的使用区间,默认从block开始到当前instruction位置.
1878        n = visitor.opr_count(LIR_OpVisitState::inputMode);
1879        for (k = 0; k < n; k++) {
1880          LIR_Opr opr = visitor.opr_at(LIR_OpVisitState::inputMode, k);
1881          assert(opr->is_register(), "visitor should only return register operands");
1882          add_use(opr, block_from, op_id, use_kind_of_input_operand(op, opr));
1883        }
1884
1885        // Add uses of live locals from interpreter's point of view for proper
1886        // debug information generation
1887        // Treat these operands as temp values (if the life range is extended
1888        // to a call site, the value would be in a register at the call otherwise)
1889        n = visitor.info_count();
1890        for (k = 0; k < n; k++) {
1891          CodeEmitInfo* info = visitor.info_at(k);
1892          ValueStack* stack = info->stack();
```

```
          for_each_state_value(stack, value,
            add_use(value, block_from, op_id + 1, noUse);
          );
        }

        // special steps for some instructions (especially moves)
        handle_method_arguments(op);
        handle_doubleword_moves(op);
        add_register_hints(op);

      } // end of instruction iteration
    } // end of block iteration

......
}
```

最后,列出寄存器分配前后的对比.

```
begin_intervals
  name "Before Register Allocation"
//格式为:
regNum type opr parentRegNum regHintRegNum (rngStart_rngEnd)+ (usePosAndKind)* spillState
0 fixed "[rsi|I]" 0 177 [0, 4[ [214, 217[  "no spill store"
1 fixed "[rdi|I]" 1 -1 [0, 1[ [216, 217[  "no definition"
2 fixed "[rbx|I]" 2 -1 [0, 1[ [216, 217[  "no definition"
3 fixed "[rax|I]" 3 196 [0, 1[ [194, 196[ [200, 202[ [216, 218[ [220, 222[ [226, 228[  "no optimization"
4 fixed "[rdx|I]" 4 178 [0, 6[ [210, 217[  "no spill store"
5 fixed "[rcx|I]" 5 189 [0, 8[ [212, 217[  "no spill store"
6 fixed "[r8|I]" 6 -1 [0, 1[ [216, 217[  "no definition"
7 fixed "[r9|I]" 7 -1 [0, 1[ [216, 217[  "no definition"
8 fixed "[r11|I]" 8 -1 [0, 1[ [216, 217[  "no definition"
9 fixed "[r13|I]" 9 -1 [0, 1[ [216, 217[  "no definition"
10 fixed "[r14|I]" 10 -1 [0, 1[ [216, 217[  "no definition"
177 object 177 0 [4, 108[ [204, 214[ 4 M 36 M 206 S 214 S  "no spill store"
178 int 178 4 [6, 192[ [204, 210[ 6 M 92 M 142 S 191 L 210 S  "no spill store"
179 int 179 5 [8, 56[ [66, 86[ 8 M 40 M 68 M 86 S  "no spill store"
180 *metadata* 180 -1 [10, 16[ 10 M 12 M 16 M  "no spill store"
181 int 181 -1 [12, 22[ 12 M 14 M 16 M 20 M 22 M  "no optimization"
182 *metadata* 182 -1 [18, 24[ 18 M 24 M  "no spill store"
183 object 183 -1 [36, 192[ 36 M 38 M 136 M 140 M 191 L  "no spill store"
184 int 184 -1 [38, 192[ 38 M 68 S 116 S 191 L  "no spill store"
185 *metadata* 185 -1 [42, 50[ 42 M 46 M 50 M  "no spill store"
186 long 186 -1 [44, 50[ 44 M 46 M 50 M  "no spill store"
187 long 187 -1 [46, 50[ 46 M 48 M 50 M  "no spill store"
188 *metadata* 188 -1 [58, 60[ 58 M 60 M  "no spill store"
189 int 189 179 [62, 66[ [86, 110[ [204, 212[ 62 M 86 M 110 S 212 S  "no optimization"
190 *metadata* 190 -1 [70, 78[ 70 M 74 M 78 M  "no spill store"
191 long 191 -1 [72, 78[ 72 M 74 M 78 M  "no spill store"
192 long 192 -1 [74, 78[ 74 M 76 M 78 M  "no spill store"
193 *metadata* 193 -1 [94, 102[ 94 M 98 M 102 M  "no spill store"
194 long 194 -1 [96, 102[ 96 M 98 M 102 M  "no spill store"
195 long 195 -1 [98, 102[ 98 M 100 M 102 M  "no spill store"
196 int 196 189 [110, 160[ [188, 194[ 110 M 116 M 134 M 136 M 138 M 160 S 188 M 191 L 194 S  "no optimization"
197 *metadata* 197 -1 [118, 126[ 118 M 122 M 126 M  "no spill store"
198 long 198 -1 [120, 126[ 120 M 122 M 126 M  "no spill store"
199 long 199 -1 [122, 126[ 122 M 124 M 126 M  "no spill store"
200 long 200 196 [134, 140[ 134 M 140 M  "no spill store"
201 int 201 -1 [140, 142[ 140 M 142 M  "no spill store"
202 *metadata* 202 -1 [144, 152[ 144 M 148 M 152 M  "no spill store"
203 long 203 -1 [146, 152[ 146 M 148 M 152 M  "no spill store"
204 long 204 -1 [148, 152[ 148 M 150 M 152 M  "no spill store"
205 int 205 196 [160, 188[ 160 M 162 M 188 S  "no spill store"
206 *metadata* 206 -1 [164, 170[ 164 M 166 M 170 M  "no spill store"
207 int 207 -1 [166, 176[ 166 M 168 M 170 M 174 M 176 M  "no optimization"
208 *metadata* 208 -1 [172, 178[ 172 M 178 M  "no spill store"
209 *metadata* 209 -1 [184, 186[ 184 M 186 M  "no spill store"
210 object 210 -1 [208, 209[ 208 M  "no definition"
211 long 211 -1 [208, 209[ 208 M  "no definition"
212 object 212 177 [206, 207[ [208, 209[ 206 M 208 M  "no spill store"
213 int 213 3 [218, 220[ 218 M 220 S  "no spill store"
end_intervals
begin_intervals
  name "After Register Allocation"
0 fixed "[rsi|I]" 0 177 [0, 4[ [214, 217[  "no spill store"
1 fixed "[rdi|I]" 1 -1 [0, 1[ [216, 217[  "no definition"
2 fixed "[rbx|I]" 2 -1 [0, 1[ [216, 217[  "no definition"
3 fixed "[rax|I]" 3 196 [0, 1[ [194, 196[ [200, 202[ [216, 218[ [220, 222[ [226, 228[  "no optimization"
4 fixed "[rdx|I]" 4 178 [0, 6[ [210, 217[  "no spill store"
5 fixed "[rcx|I]" 5 189 [0, 8[ [212, 217[  "no spill store"
6 fixed "[r8|I]" 6 -1 [0, 1[ [216, 217[  "no definition"
7 fixed "[r9|I]" 7 -1 [0, 1[ [216, 217[  "no definition"
8 fixed "[r11|I]" 8 -1 [0, 1[ [216, 217[  "no definition"
9 fixed "[r13|I]" 9 -1 [0, 1[ [216, 217[  "no definition"
10 fixed "[r14|I]" 10 -1 [0, 1[ [216, 217[  "no definition"
//寄存器177分配后,使用物理寄存器rsi.
177 object "[rsi|L]" 177 0 [4, 108[ [204, 214[ 4 M 36 M 206 S 214 S  "no spill store"
178 int "[rdx|I]" 178 4 [6, 192[ [204, 210[ 6 M 92 M 142 S 191 L 210 S  "no spill store"
```

```
  179 int "[rcx|I]" 179 5 [8, 56[ [66, 86[ 8 M 40 M 68 M 86 S  "no spill store"
  180 *metadata* "[rax|M]" 180 -1 [10, 16[ 10 M 12 M 16 M  "no spill store"
  181 int "[rdi|I]" 181 -1 [12, 22[ 12 M 14 M 16 M 20 M 22 M  "no optimization"
  182 *metadata* "[rax|M]" 182 -1 [18, 24[ 18 M 24 M  "no spill store"
  183 object "[rax|L]" 183 -1 [36, 192[ 36 M 38 M 136 M 140 M 191 L  "no spill store"
  184 int "[rdi|I]" 184 -1 [38, 192[ 38 M 68 S 116 S 191 L  "no spill store"
  185 *metadata* "[rbx|M]" 185 -1 [42, 50[ 42 M 46 M 50 M  "no spill store"
  186 long "[r8r8|J]" 186 -1 [44, 50[ 44 M 46 M 50 M  "no spill store"
  187 long "[r9r9|J]" 187 -1 [46, 50[ 46 M 48 M 50 M  "no spill store"
  188 *metadata* "[rcx|M]" 188 -1 [58, 60[ 58 M 60 M  "no spill store"
  189 int "[rcx|I]" 189 179 [62, 66[ [86, 110[ [204, 212[ 62 M 86 M 110 S 212 S  "no optimization"
  190 *metadata* "[rbx|M]" 190 -1 [70, 78[ 70 M 74 M 78 M  "no spill store"
  191 long "[r8r8|J]" 191 -1 [72, 78[ 72 M 74 M 78 M  "no spill store"
  192 long "[r9r9|J]" 192 -1 [74, 78[ 74 M 76 M 78 M  "no spill store"
  193 *metadata* "[rbx|M]" 193 -1 [94, 102[ 94 M 98 M 102 M  "no spill store"
  194 long "[r8r8|J]" 194 -1 [96, 102[ 96 M 98 M 102 M  "no spill store"
  195 long "[r9r9|J]" 195 -1 [98, 102[ 98 M 100 M 102 M  "no spill store"
  196 int "[rcx|I]" 196 189 [110, 160[ [188, 194[ 110 M 116 M 134 M 136 M 138 M 160 S 188 M 191 L 194 S  "no optimization"
  197 *metadata* "[rsi|M]" 197 -1 [118, 126[ 118 M 122 M 126 M  "no spill store"
  198 long "[rbxrbx|J]" 198 -1 [120, 126[ 120 M 122 M 126 M  "no spill store"
  199 long "[r8r8|J]" 199 -1 [122, 126[ 122 M 124 M 126 M  "no spill store"
  200 long "[rsirsi|J]" 200 196 [134, 140[ 134 M 140 M  "no spill store"
  201 int "[rsi|I]" 201 -1 [140, 142[ 140 M 142 M  "no spill store"
  202 *metadata* "[rsi|M]" 202 -1 [144, 152[ 144 M 148 M 152 M  "no spill store"
  203 long "[rbxrbx|J]" 203 -1 [146, 152[ 146 M 148 M 152 M  "no spill store"
  204 long "[r8r8|J]" 204 -1 [148, 152[ 148 M 150 M 152 M  "no spill store"
  205 int "[rcx|I]" 205 196 [160, 188[ 160 M 162 M 188 S  "no spill store"
  206 *metadata* "[rsi|M]" 206 -1 [164, 170[ 164 M 166 M 170 M  "no spill store"
  207 int "[rbx|I]" 207 -1 [166, 176[ 166 M 168 M 170 M 174 M 176 M  "no optimization"
  208 *metadata* "[rsi|M]" 208 -1 [172, 178[ 172 M 178 M  "no spill store"
  209 *metadata* "[rsi|M]" 209 -1 [184, 186[ 184 M 186 M  "no spill store"
  210 object "[rbx|L]" 210 -1 [208, 209[ 208 M  "no definition"
  211 long "[raxrax|J]" 211 -1 [208, 209[ 208 M  "no definition"
  212 object "[rdi|L]" 212 177 [206, 207[ [208, 209[ 206 M 208 M  "no spill store"
  213 int "[rax|I]" 213 3 [218, 220[ 218 M 220 S  "no spill store"
  end_intervals
  这个例子的物理寄存器刚好都够用,不需要split interval.下面给一个需要分割区间的例子.
  虚拟寄存器194的120-250区间就无法找到一个完整的物理寄存器.
  寄存器分配完后,区间分配为120-190, 190-201, 201-230, 230-250.
  其中,190-201, 201-230使用了桢栈空间,230以后又回到了rdi寄存器的使用.
  总之,算法是尽可能使用物理寄存器来加速程序执行.

  begin_intervals
    name "Before Register Allocation"
  0 fixed "[rsi|I]" 0 179 [0, 4[ [200, 203[ [356, 359[ [466, 469[  "no optimization"
  1 fixed "[rdi|I]" 1 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  2 fixed "[rbx|I]" 2 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  3 fixed "[rax|I]" 3 227 [0, 1[ [202, 204[ [358, 360[ [442, 444[ [448, 450[ [468, 470[ [472, 474[ [478, 480[  "no optimization"
  4 fixed "[rdx|I]" 4 202 [0, 6[ [198, 203[ [354, 359[ [462, 469[  "no optimization"
  5 fixed "[rcx|I]" 5 179 [0, 8[ [202, 203[ [358, 359[ [464, 469[  "no spill store"
  6 fixed "[r8|I]" 6 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  7 fixed "[r9|I]" 7 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  8 fixed "[r11|I]" 8 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  9 fixed "[r13|I]" 9 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  10 fixed "[r14|I]" 10 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  ......
  194 object 194 -1 [98, 116[ [120, 250[ [254, 274[ [452, 454[ [476, 478[ 98 M 100 M 122 M 146 M 232 M 256 M 454 M 478 S  "no spill store"
  ......
  end_intervals
  begin_intervals
    name "After Register Allocation"
  0 fixed "[rsi|I]" 0 179 [0, 4[ [200, 203[ [356, 359[ [466, 469[  "no optimization"
  1 fixed "[rdi|I]" 1 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  2 fixed "[rbx|I]" 2 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  3 fixed "[rax|I]" 3 227 [0, 1[ [202, 204[ [358, 360[ [442, 444[ [448, 450[ [468, 470[ [472, 474[ [478, 480[  "no optimization"
  4 fixed "[rdx|I]" 4 202 [0, 6[ [198, 203[ [354, 359[ [462, 469[  "no optimization"
  5 fixed "[rcx|I]" 5 179 [0, 8[ [202, 203[ [358, 359[ [464, 469[  "no spill store"
  6 fixed "[r8|I]" 6 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  7 fixed "[r9|I]" 7 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  8 fixed "[r11|I]" 8 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  9 fixed "[r13|I]" 9 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  10 fixed "[r14|I]" 10 -1 [0, 1[ [202, 203[ [358, 359[ [468, 469[  "no definition"
  ......
  194 object "[rdi|L]" 194 -1 [98, 116[ [120, 190[ 98 M 100 M 122 M 146 M  "one spill store"
  ......
  263 object "[stack:3|L]" 194 194 [190, 201[  "one spill store"
  264 object "[stack:3|L]" 194 194 [201, 230[  "one spill store"
  ......
  267 object "[rdi|L]" 194 194 [230, 250[ [254, 274[ [452, 454[ [476, 478[ 232 M 256 M 454 M 478 S  "one spill store"
  ......
  end_intervals
  生成代码
  最后,LIR分配物理寄存器后如下:

  begin_cfg
    name "Before Code Generation"
```

```
    begin_block
      name "B12"
      from_bci 0
      to_bci 0
      predecessors
      successors "B13"
      xhandlers
      flags
      first_lir_id 0
      last_lir_id 28
      begin_LIR
     0 label [label:0xfc0043b8] <|@
     2 std_entry  <|@
    10 move [metadata:0x3301c100|M] [rax|M]  <|@
    12 move [Base:[rax|M] Disp: 108|I] [rdi|I]  <|@
    14 add [rdi|I] [int:8|I] [rdi|I] <|@
    16 move [rdi|I] [Base:[rax|M] Disp: 108|I]  <|@
    18 move [metadata:0x32dd2238|M] [rax|M]  <|@
    20 logic_and [rdi|I] [int:8184|I] [rdi|I] <|@
    22 cmp [rdi|I] [int:0|I]  <|@
    24 branch [EQ] [CounterOverflowStub: 0xf8025908] <|@
    26 label [label:0xf8025930] <|@
      end_LIR
    end_block
    begin_block
      name "B13"
      from_bci 0
      to_bci 0
      predecessors "B12"
      successors "B0"
      xhandlers
      flags "std"
      dominator "B12"
      first_lir_id 30
      last_lir_id 32
      begin_LIR
    30 label [label:0xfc0046c8] <|@
      end_LIR
    end_block
    begin_block
      name "B0"
      from_bci 0
      to_bci 7
      predecessors "B13"
      successors "B2" "B1"
      xhandlers
      flags "std"
      dominator "B13"
      first_lir_id 34
      last_lir_id 54
      begin_LIR
    34 label [label:0xf801ee78] <|@
    36 move [Base:[rsi|L] Disp: 12|L] [rax|L]  <|@
    38 move [Base:[rax|L] Disp: 12|I] [rdi|I]  [bci:4] <|@
    40 cmp [rcx|I] [int:0|I]  <|@
    42 move [metadata:0x3301c100|M] [rbx|M]  <|@
    44 cmove [GE] [lng:152|J] [lng:168|J] [r8r8|J] <|@
    46 move [Base:[rbx|M] Index:[r8r8|J] Disp: 0|J] [r9r9|J]  <|@
    48 leal [Base:[r9r9|J] Disp: 1|I] [r9r9|J]  <|@
    50 move [r9r9|J] [Base:[rbx|M] Index:[r8r8|J] Disp: 0|J]  <|@
    52 branch [GE] [B2]  <|@
      end_LIR
    end_block
    begin_block
      name "B1"
      from_bci 10
      to_bci 12
      predecessors "B0"
      successors "B3"
      xhandlers
      flags
      dominator "B0"
      first_lir_id 56
      last_lir_id 64
      begin_LIR
    56 label [label:0xf801f198] <|@
    58 move [metadata:0x3301c100|M] [rcx|M]  <|@
    60 add [Base:[rcx|M] Disp: 184|J] [int:1|I] [Base:[rcx|M] Disp: 184|J] <|@
    62 move [int:0|I] [rcx|I]  <|@
    64 branch [AL] [B3]  <|@
      end_LIR
    end_block
    begin_block
      name "B2"
      from_bci 15
      to_bci 17
```

```
      predecessors "B0"
      successors "B3" "B4"
      xhandlers
      flags
      dominator "B0"
      first_lir_id 66
      last_lir_id 82
      begin_LIR
     66 label [label:0xf801f4b8] <|@
     68 cmp [rcx|I] [rdi|I]  <|@
     70 move [metadata:0x3301c100|M] [rbx|M]  <|@
     72 cmove [LT] [lng:208|J] [lng:224|J] [r8r8|J] <|@
     74 move [Base:[rbx|M] Index:[r8r8|J] Disp: 0|J] [r9r9|J]  <|@
     76 leal [Base:[r9r9|J] Disp: 1|I] [r9r9|J]  <|@
     78 move [r9r9|J] [Base:[rbx|M] Index:[r8r8|J] Disp: 0|J]  <|@
     80 branch [GE] [B4]  <|@
      end_LIR
    end_block
    begin_block
      name "B3"
      from_bci 22
      to_bci 25
      predecessors "B1" "B2"
      successors "B6" "B7"
      xhandlers
      flags
      dominator "B0"
      first_lir_id 90
      last_lir_id 106
      begin_LIR
     90 label [label:0xf801f7d8] <|@
     92 cmp [rdx|I] [int:65536|I]  <|@
     94 move [metadata:0x3301c100|M] [rbx|M]  <|@
     96 cmove [GE] [lng:240|J] [lng:256|J] [r8r8|J] <|@
     98 move [Base:[rbx|M] Index:[r8r8|J] Disp: 0|J] [r9r9|J]  <|@
    100 leal [Base:[r9r9|J] Disp: 1|I] [r9r9|J]  <|@
    102 move [r9r9|J] [Base:[rbx|M] Index:[r8r8|J] Disp: 0|J]  <|@
    104 branch [GE] [B6]  <|@
    106 branch [AL] [B7]  <|@
      end_LIR
    end_block
    begin_block
      name "B8"
      from_bci 43
      to_bci 49
      predecessors "B7"
      successors "B11" "B10"
      xhandlers
      flags "bb"
      dominator "B7"
      loop_index 0
      loop_depth 1
      first_lir_id 132
      last_lir_id 156
      begin_LIR
    132 label [label:0xf8020778] <|@
    134 convert [i2l] [rcx|I] [rsirsi|J]  <|@
    136 cmp [rcx|I] [Base:[rax|L] Disp: 12|I]  <|@
    138 branch [AE] [RangeCheckStub: 0xf80296d8] [bci:47] <|@
    140 move [Base:[rax|L] Index:[rsirsi|J] * 2 Disp: 16|C] [rsi|I]  <|@
    142 cmp [rsi|I] [rdx|I]  <|@
    144 move [metadata:0x3301c100|M] [rsi|M]  <|@
    146 cmove [NE] [lng:304|J] [lng:320|J] [rbxrbx|J] <|@
    148 move [Base:[rsi|M] Index:[rbxrbx|J] Disp: 0|J] [r8r8|J]  <|@
    150 leal [Base:[r8r8|J] Disp: 1|I] [r8r8|J]  <|@
    152 move [r8r8|J] [Base:[rsi|M] Index:[rbxrbx|J] Disp: 0|J]  <|@
    154 branch [EQ] [B10]  <|@
      end_LIR
    end_block
    begin_block
      name "B11"
      from_bci 55
      to_bci 58
      predecessors "B8"
      successors "B7"
      xhandlers
      flags "lle"
      dominator "B8"
      loop_index 0
      loop_depth 1
      first_lir_id 158
      last_lir_id 190
      begin_LIR
    158 label [label:0xf80210d8] <|@
    162 add [rcx|I] [int:1|I] [rcx|I] <|@
    164 move [metadata:0x3301c100|M] [rsi|M]  <|@
```

```
   166 move [Base:[rsi|M] Disp: 112|I] [rbx|I]  <|@
   168 add [rbx|I] [int:8|I] [rbx|I] <|@
   170 move [rbx|I] [Base:[rsi|M] Disp: 112|I]  <|@
   172 move [metadata:0x32dd2238|M] [rsi|M]  <|@
   174 logic_and [rbx|I] [int:65528|I] [rbx|I] <|@
   176 cmp [rbx|I] [int:0|I]  <|@
   178 branch [EQ] [CounterOverflowStub: 0xf802a488] <|@
   180 label [label:0xf802a4b0] <|@
   182 safepoint    [bci:58] <|@
   184 move [metadata:0x3301c100|M] [rsi|M]  <|@
   186 add [Base:[rsi|M] Disp: 336|J] [int:1|I] [Base:[rsi|M] Disp: 336|J] <|@
     end_LIR
  end_block
  begin_block
     name "B7"
     from_bci 37
     to_bci 40
     predecessors "B11" "B3"
     successors "B9" "B8"
     xhandlers
     flags "plh" "llh"
     dominator "B5"
     loop_index 0
     loop_depth 1
     first_lir_id 114
     last_lir_id 130
     begin_LIR
   114 label [label:0xf8020458] <|@
   116 cmp [rcx|I] [rdi|I]  <|@
   118 move [metadata:0x3301c100|M] [rsi|M]  <|@
   120 cmove [GE] [lng:272|J] [lng:288|J] [rbxrbx|J] <|@
   122 move [Base:[rsi|M] Index:[rbxrbx|J] Disp: 0|J] [r8r8|J]  <|@
   124 leal [Base:[r8r8|J] Disp: 1|I] [r8r8|J]  <|@
   126 move [r8r8|J] [Base:[rsi|M] Index:[rbxrbx|J] Disp: 0|J]  <|@
   128 branch [GE] [B9]  <|@
   130 branch [AL] [B8]  <|@
     end_LIR
  end_block
  begin_block
     name "B10"
     from_bci 52
     to_bci 54
     predecessors "B8"
     successors
     xhandlers
     flags
     dominator "B8"
     first_lir_id 192
     last_lir_id 196
     begin_LIR
   192 label [label:0xf8020db8] <|@
   194 move [rcx|I] [rax|I]  <|@
   196 return [rax|I]   <|@
     end_LIR
  end_block
  begin_block
     name "B9"
     from_bci 61
     to_bci 62
     predecessors "B7"
     successors
     xhandlers
     flags
     dominator "B7"
     first_lir_id 198
     last_lir_id 202
     begin_LIR
   198 label [label:0xf8020a98] <|@
   200 move [int:-1|I] [rax|I]  <|@
   202 return [rax|I]   <|@
     end_LIR
  end_block
  begin_block
     name "B6"
     from_bci 63
     to_bci 69
     predecessors "B3"
     successors
     xhandlers
     flags
     dominator "B3"
     first_lir_id 204
     last_lir_id 222
     begin_LIR
   204 label [label:0xf8020138] <|@
   206 move [rsi|L] [rdi|L]  <|@
```

```
 208 profile_call indexOf.java/lang/String @ 66 [rbx|L] [rdi|L] [raxrax|J]  <|@
 216 optvirtual call: [addr: 0x0] [recv: [rsi|L]] [result: [rax|I]] [bci:66] <|@
 222 return [rax|I]   <|@
    end_LIR
  end_block
  begin_block
    name "B4"
    from_bci 20
    to_bci 21
    predecessors "B2"
    successors
    xhandlers
    flags
    dominator "B2"
    first_lir_id 224
    last_lir_id 228
    begin_LIR
 224 label [label:0xf801faf8] <|@
 226 move [int:-1|I] [rax|I]  <|@
 228 return [rax|I]   <|@
    end_LIR
  end_block
end_cfg
```

发射汇编代码的过程如：

```
int Compilation::compile_java_method()
    int Compilation::emit_code_body()
        void LIR_Assembler::emit_code(BlockList* hir)
            void LIR_Assembler::emit_block(BlockBegin* block)
                void LIR_Assembler::emit_lir_list(LIR_List* list)
                    virtual void emit_code(LIR_Assembler* masm)
                        void LIR_Op1::emit_code(LIR_Assembler* masm) //比如前面的lir_move

void LIR_Assembler::emit_op1(LIR_Op1* op) {
  switch (op->code()) {
    case lir_move:
      if (op->move_kind() == lir_move_volatile) {
        assert(op->patch_code() == lir_patch_none, "can't patch volatiles");
        volatile_move_op(op->in_opr(), op->result_opr(), op->type(), op->info());
      } else {
        move_op(op->in_opr(), op->result_opr(), op->type(),
                op->patch_code(), op->info(), op->pop_fpu_stack(),
                op->move_kind() == lir_move_unaligned,
                op->move_kind() == lir_move_wide);
      }
      break;
......
}

void LIR_Assembler::move_op(LIR_Opr src, LIR_Opr dest, BasicType type, LIR_PatchCode patch_code, CodeEmitInfo* info, bool pop_fpu_stack, bool
  if (src->is_register()) {
    if (dest->is_register()) {
      assert(patch_code == lir_patch_none && info == NULL, "no patching and info allowed here");
      reg2reg(src,  dest);
    } else if (dest->is_stack()) {
......
}

void LIR_Assembler::reg2reg(LIR_Opr src, LIR_Opr dest) {
......
  // move between cpu-registers
  if (dest->is_single_cpu()) {
......
    move_regs(src->as_register(), dest->as_register());

  } else if (dest->is_double_cpu()) {
......
}

void LIR_Assembler::move_regs(Register from_reg, Register to_reg) {
  if (from_reg != to_reg) __ mov(to_reg, from_reg);
}

//Code in assembler_x86.cpp
void Assembler::mov(Register dst, Register src) {
  LP64_ONLY(movq(dst, src)) NOT_LP64(movl(dst, src));
}
```