# Relazione lavoro svolto

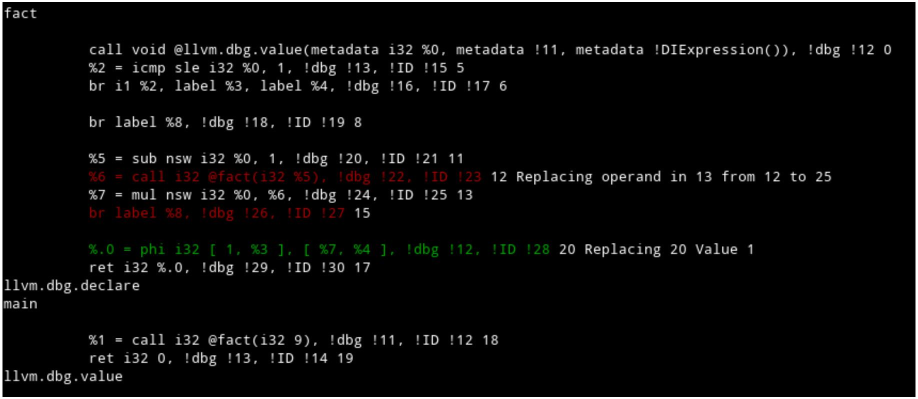
Pietro Ghiglio

October 15, 2020

## Table of Contents

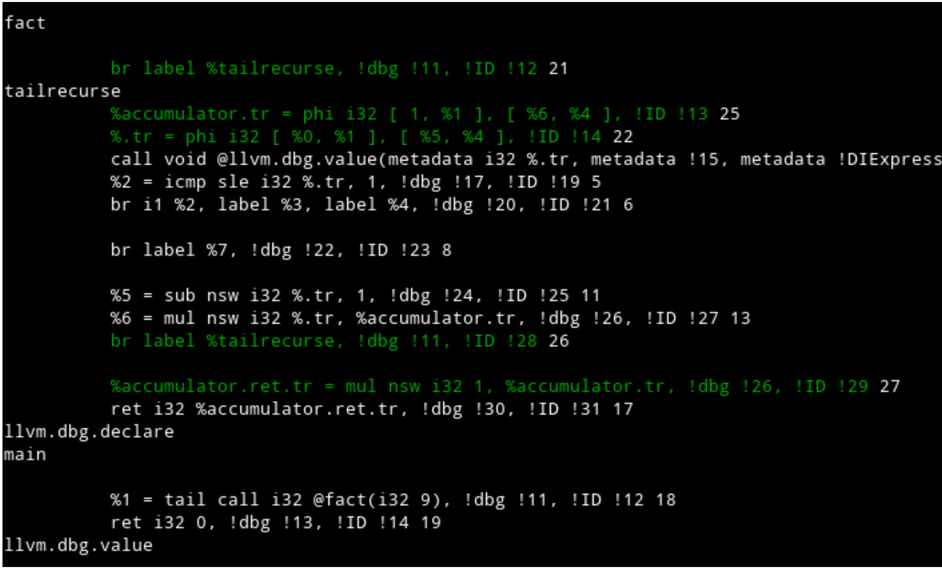
## Table of Contents

## Prima di TailCallElim



```
fact

        call void @llvm.dbg.value(metadata i32 %0, metadata !11, metadata !DIExpression()), !dbg !12 0
        %2 = icmp sle i32 %0, 1, !dbg !13, !ID !15 5
        br i1 %2, label %3, label %4, !dbg !16, !ID !17 6

        br label %8, !dbg !18, !ID !19 8

        %5 = sub nsw i32 %0, 1, !dbg !20, !ID !21 11
        %6 = call i32 @fact(i32 %5), !dbg !22, !ID !23 12 Replacing operand in 13 from 12 to 25
        %7 = mul nsw i32 %0, %6, !dbg !24, !ID !25 13
        br label %8, !dbg !26, !ID !27 15

        %.0 = phi i32 [ 1, %3 ], [ %7, %4 ], !dbg !12, !ID !28 20 Replacing 20 Value 1
        ret i32 %.0, !dbg !29, !ID !30 17
llvm.dbg.declare
main

        %1 = call i32 @fact(i32 9), !dbg !11, !ID !12 18
        ret i32 0, !dbg !13, !ID !14 19
llvm.dbg.value
```

## Dopo TailCallElim



```
fact

         br label %tailrecurse, !dbg !11, !ID !12 21
tailrecurse
         %accumulator.tr = phi i32 [ 1, %1 ], [ %6, %4 ], !ID !13 25
         %.tr = phi i32 [ %0, %1 ], [ %5, %4 ], !ID !14 22
         call void @llvm.dbg.value(metadata i32 %.tr, metadata !15, metadata !DIExpress
         %2 = icmp sle i32 %.tr, 1, !dbg !17, !ID !19 5
         br i1 %2, label %3, label %4, !dbg !20, !ID !21 6

         br label %7, !dbg !22, !ID !23 8

         %5 = sub nsw i32 %.tr, 1, !dbg !24, !ID !25 11
         %6 = mul nsw i32 %.tr, %accumulator.tr, !dbg !26, !ID !27 13
         br label %tailrecurse, !dbg !11, !ID !28 26

         %accumulator.ret.tr = mul nsw i32 1, %accumulator.tr, !dbg !26, !ID !29 27
         ret i32 %accumulator.ret.tr, !dbg !30, !ID !31 17
llvm.dbg.declare
main

         %1 = tail call i32 @fact(i32 9), !dbg !11, !ID !12 18
         ret i32 0, !dbg !13, !ID !14 19
llvm.dbg.value
```

## Cosa fare con nuove istruzioni?

- Sapere che una nuova istruzione è stata creata non basta per attribuire (automaticamente) info di debug mancanti.
- Guardare a quali istruzioni vengono sostituite da quelle appena aggiunte.
- Relazione "Replaced By" tra istruzioni, rappresentata da un grafo.

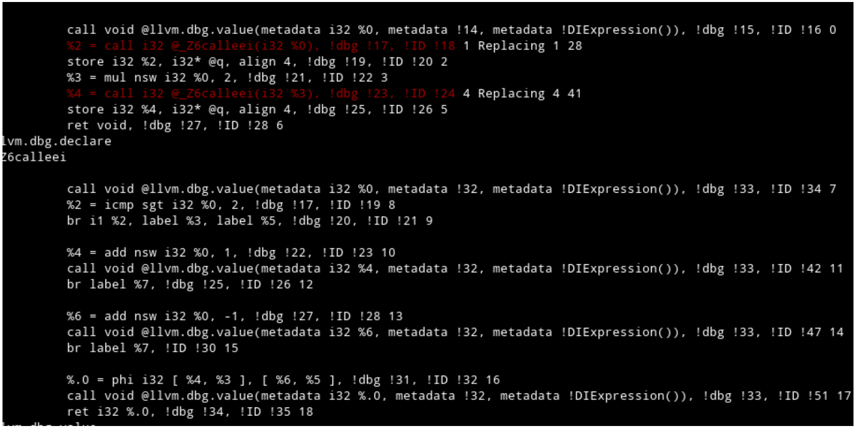
```
struct RepVertex {
  unsigned long InstID;
  std::set<SourceLocation> Locations;
};
```

## Propagare Source Locations

```
inline void propagateLocations(RepGraph& G){
  //Topological sort the graph
  std::deque<Vertex> Sorted;
  topological_sort(G, std::front_inserter(Sorted));

  //Following the topological sort, insert the locations
  //of the source in the target.
  for(auto V : Sorted){
    for(auto E : make_iterator_range(out_edges(V,G)))
      for(auto& Loc : G[V].Locations)
  G[target(E,G)].Locations.insert(Loc);
  }
}
```

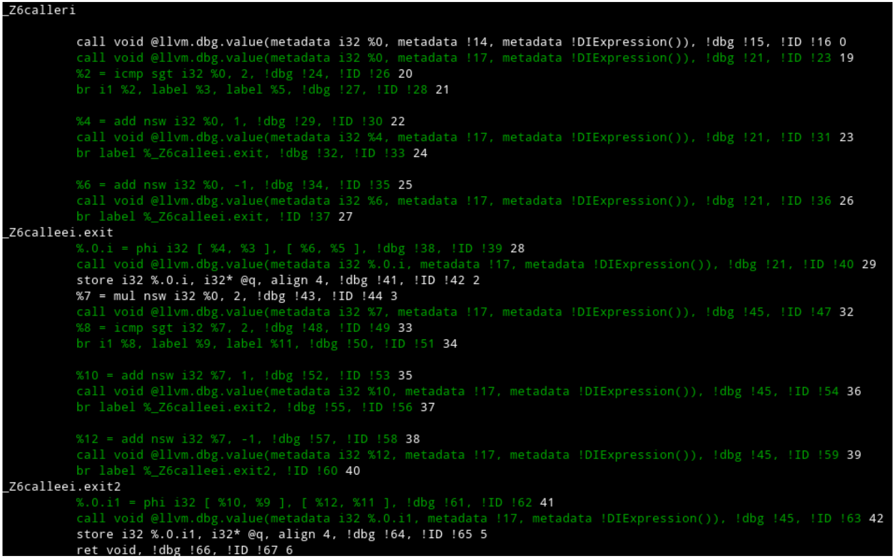
## Table of Contents

## Pre Inlining



```
        call void @llvm.dbg.value(metadata i32 %0, metadata !14, metadata !DIExpression()), !dbg !15, !ID !16 0
        %2 = call i32 @_Z6calleei(i32 %0), !dbg !17, !ID !18 1 Replacing 1 28
        store i32 %2, i32* @q, align 4, !dbg !19, !ID !20 2
        %3 = mul nsw i32 %0, 2, !dbg !21, !ID !22 3
        %4 = call i32 @_Z6calleei(i32 %3), !dbg !23, !ID !24 4 Replacing 4 41
        store i32 %4, i32* @q, align 4, !dbg !25, !ID !26 5
        ret void, !dbg !27, !ID !28 6
lvm.dbg.declare
Z6calleei

        call void @llvm.dbg.value(metadata i32 %0, metadata !32, metadata !DIExpression()), !dbg !33, !ID !34 7
        %2 = icmp sgt i32 %0, 2, !dbg !17, !ID !19 8
        br i1 %2, label %3, label %5, !dbg !20, !ID !21 9

        %4 = add nsw i32 %0, 1, !dbg !22, !ID !23 10
        call void @llvm.dbg.value(metadata i32 %4, metadata !32, metadata !DIExpression()), !dbg !33, !ID !42 11
        br label %7, !dbg !25, !ID !26 12

        %6 = add nsw i32 %0, -1, !dbg !27, !ID !28 13
        call void @llvm.dbg.value(metadata i32 %6, metadata !32, metadata !DIExpression()), !dbg !33, !ID !47 14
        br label %7, !ID !30 15

        %.0 = phi i32 [ %4, %3 ], [ %6, %5 ], !dbg !31, !ID !32 16
        call void @llvm.dbg.value(metadata i32 %.0, metadata !32, metadata !DIExpression()), !dbg !33, !ID !51 17
        ret i32 %.0, !dbg !34, !ID !35 18
```

# Post Inlining



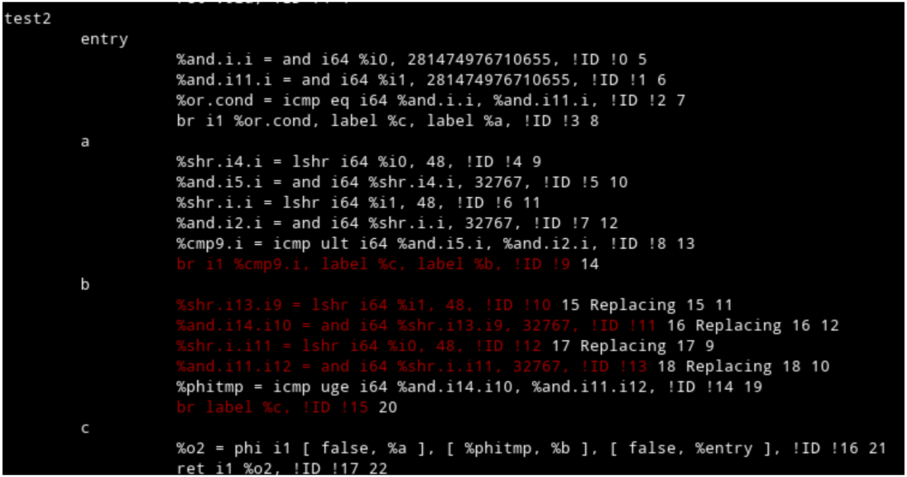
```
_Z6calleri

        call void @llvm.dbg.value(metadata i32 %0, metadata !14, metadata !DIExpression()), !dbg !15, !ID !16 0
        call void @llvm.dbg.value(metadata i32 %0, metadata !17, metadata !DIExpression()), !dbg !21, !ID !23 19
        %2 = icmp sgt i32 %0, 2, !dbg !24, !ID !26 20
        br i1 %2, label %3, label %5, !dbg !27, !ID !28 21

        %4 = add nsw i32 %0, 1, !dbg !29, !ID !30 22
        call void @llvm.dbg.value(metadata i32 %4, metadata !17, metadata !DIExpression()), !dbg !21, !ID !31 23
        br label %_Z6calleei.exit, !dbg !32, !ID !33 24

        %6 = add nsw i32 %0, -1, !dbg !34, !ID !35 25
        call void @llvm.dbg.value(metadata i32 %6, metadata !17, metadata !DIExpression()), !dbg !21, !ID !36 26
        br label %_Z6calleei.exit, !ID !37 27
_Z6calleei.exit
        %.0.i = phi i32 [ %4, %3 ], [ %6, %5 ], !dbg !38, !ID !39 28
        call void @llvm.dbg.value(metadata i32 %.0.i, metadata !17, metadata !DIExpression()), !dbg !21, !ID !40 29
        store i32 %.0.i, i32* @q, align 4, !dbg !41, !ID !42 2
        %7 = mul nsw i32 %0, 2, !dbg !43, !ID !44 3
        call void @llvm.dbg.value(metadata i32 %7, metadata !17, metadata !DIExpression()), !dbg !45, !ID !47 32
        %8 = icmp sgt i32 %7, 2, !dbg !48, !ID !49 33
        br i1 %8, label %9, label %11, !dbg !50, !ID !51 34

        %10 = add nsw i32 %7, 1, !dbg !52, !ID !53 35
        call void @llvm.dbg.value(metadata i32 %10, metadata !17, metadata !DIExpression()), !dbg !45, !ID !54 36
        br label %_Z6calleei.exit2, !dbg !55, !ID !56 37

        %12 = add nsw i32 %7, -1, !dbg !57, !ID !58 38
        call void @llvm.dbg.value(metadata i32 %12, metadata !17, metadata !DIExpression()), !dbg !45, !ID !59 39
        br label %_Z6calleei.exit2, !ID !60 40
_Z6calleei.exit2
        %.0.i1 = phi i32 [ %10, %9 ], [ %12, %11 ], !dbg !61, !ID !62 41
        call void @llvm.dbg.value(metadata i32 %.0.i1, metadata !17, metadata !DIExpression()), !dbg !45, !ID !63 42
        store i32 %.0.i1, i32* @q, align 4, !dbg !64, !ID !65 5
        ret void, !dbg !66, !ID !67 6
```

## Osservazioni

- La maggior parte delle istruzioni inserite ha info di debug corrette.
- I due phi-node che sostinuiscono le call hanno info di debug, ma hanno Line e Column a 0.
- Propagando le location, gli viene assegnata la location delle call.

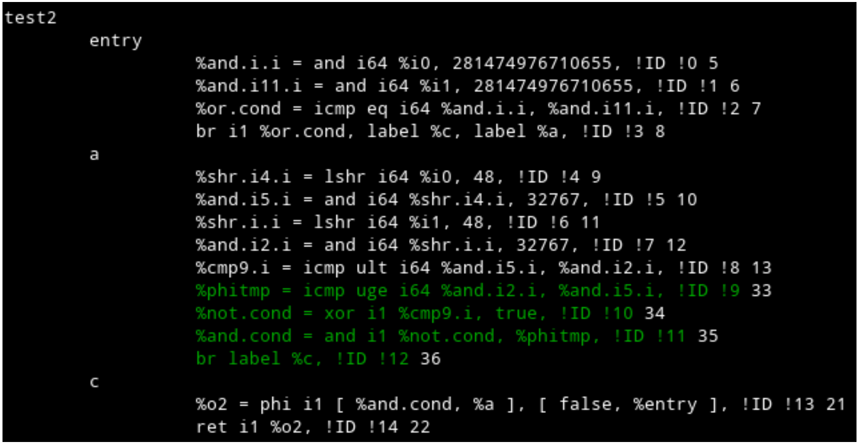
# Table of Contents

# Pre Simplify



```
test2
        entry
                  %and.i.i = and i64 %i0, 281474976710655, !ID !0 5
                  %and.i11.i = and i64 %i1, 281474976710655, !ID !1 6
                  %or.cond = icmp eq i64 %and.i.i, %and.i11.i, !ID !2 7
                  br i1 %or.cond, label %c, label %a, !ID !3 8
        a
                  %shr.i4.i = lshr i64 %i0, 48, !ID !4 9
                  %and.i5.i = and i64 %shr.i4.i, 32767, !ID !5 10
                  %shr.i.i = lshr i64 %i1, 48, !ID !6 11
                  %and.i2.i = and i64 %shr.i.i, 32767, !ID !7 12
                  %cmp9.i = icmp ult i64 %and.i5.i, %and.i2.i, !ID !8 13
                  br i1 %cmp9.i, label %c, label %b, !ID !9 14
        b
                  %shr.i13.i9 = lshr i64 %i1, 48, !ID !10 15 Replacing 15 11
                  %and.i14.i10 = and i64 %shr.i13.i9, 32767, !ID !11 16 Replacing 16 12
                  %shr.i.i11 = lshr i64 %i0, 48, !ID !12 17 Replacing 17 9
                  %and.i11.i12 = and i64 %shr.i.i11, 32767, !ID !13 18 Replacing 18 10
                  %phitmp = icmp uge i64 %and.i14.i10, %and.i11.i12, !ID !14 19
                  br label %c, !ID !15 20
        c
                  %o2 = phi i1 [ false, %a ], [ %phitmp, %b ], [ false, %entry ], !ID !16 21
                  ret i1 %o2, !ID !17 22
```

## Post Simplify



```
test2
        entry
                  %and.i.i = and i64 %i0, 281474976710655, !ID !0 5
                  %and.i11.i = and i64 %i1, 281474976710655, !ID !1 6
                  %or.cond = icmp eq i64 %and.i.i, %and.i11.i, !ID !2 7
                  br i1 %or.cond, label %c, label %a, !ID !3 8
        a
                  %shr.i4.i = lshr i64 %i0, 48, !ID !4 9
                  %and.i5.i = and i64 %shr.i4.i, 32767, !ID !5 10
                  %shr.i.i = lshr i64 %i1, 48, !ID !6 11
                  %and.i2.i = and i64 %shr.i.i, 32767, !ID !7 12
                  %cmp9.i = icmp ult i64 %and.i5.i, %and.i2.i, !ID !8 13
                  %phitmp = icmp uge i64 %and.i2.i, %and.i5.i, !ID !9 33
                  %not.cond = xor i1 %cmp9.i, true, !ID !10 34
                  %and.cond = and i1 %not.cond, %phitmp, !ID !11 35
                  br label %c, !ID !12 36
        c
                  %o2 = phi i1 [ %and.cond, %a ], [ false, %entry ], !ID !13 21
                  ret i1 %o2, !ID !14 22
```

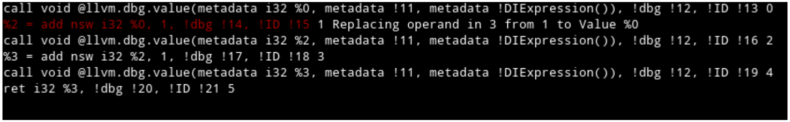
## Osservazioni

- La istruzioni del basic block b vengono sostituite con istruzioni di a.
- Propagando le locations, le istruzioni di a avranno due locations (quelle che avevano prima, e quelle di b).
  Cosa fare quando si attribuisce il costo energetico?

## Table of Contents

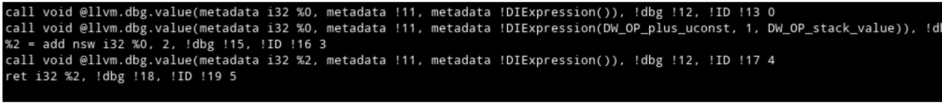
## InstCombine

Pre:



```
call void @llvm.dbg.value(metadata i32 %0, metadata !11, metadata !DIExpression()), !dbg !12, !ID !13 0
%2 = add nsw i32 %0, 1, !dbg !14, !ID !15 1 Replacing operand in 3 from 1 to Value %0
call void @llvm.dbg.value(metadata i32 %2, metadata !11, metadata !DIExpression()), !dbg !12, !ID !16 2
%3 = add nsw i32 %2, 1, !dbg !17, !ID !18 3
call void @llvm.dbg.value(metadata i32 %3, metadata !11, metadata !DIExpression()), !dbg !12, !ID !19 4
ret i32 %3, !dbg !20, !ID !21 5
```

Post:



```
call void @llvm.dbg.value(metadata i32 %0, metadata !11, metadata !DIExpression()), !dbg !12, !ID !13 0
call void @llvm.dbg.value(metadata i32 %0, metadata !11, metadata !DIExpression(DW_OP_plus_uconst, 1, DW_OP_stack_value)), !db
%2 = add nsw i32 %0, 2, !dbg !15, !ID !16 3
call void @llvm.dbg.value(metadata i32 %2, metadata !11, metadata !DIExpression()), !dbg !12, !ID !17 4
ret i32 %2, !dbg !18, !ID !19 5
```

## Osservazioni

- Per via dell'implementazione della trasformazione, dal log non si evince chiaramente che l'istruzione 3 diventa il merge di 1 e 3.
- Altri problemi: istruzioni senza usi (store, branch), replace con un value che non è un'istruzione.