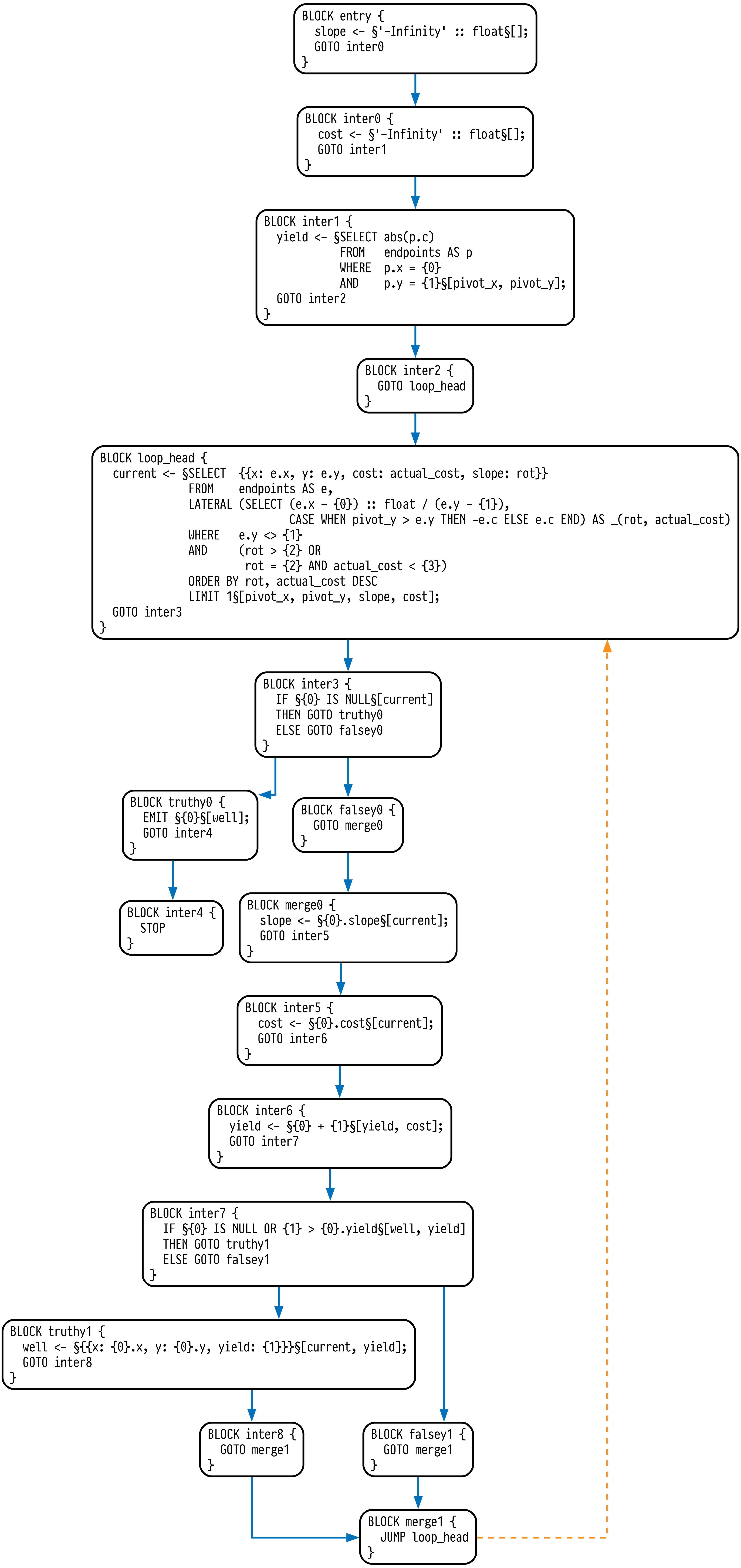

```
BLOCK entry {
  slope <- §'-Infinity' :: float§[];
  GOTO inter0
}

BLOCK inter0 {
  cost <- §'-Infinity' :: float§[];
  GOTO inter1
}

BLOCK inter1 {
  yield <- §SELECT abs(p.c)
            FROM   endpoints AS p
            WHERE  p.x = {0}
            AND    p.y = {1}§[pivot_x, pivot_y];
  GOTO inter2
}

BLOCK inter2 {
  GOTO loop_head
}

BLOCK loop_head {
  current <- §SELECT  {{x: e.x, y: e.y, cost: actual_cost, slope: rot}}
              FROM    endpoints AS e,
              LATERAL (SELECT (e.x - {0}) :: float / (e.y - {1}),
                              CASE WHEN pivot_y > e.y THEN -e.c ELSE e.c END) AS _(rot, actual_cost)
              WHERE   e.y <> {1}
              AND     (rot > {2} OR
                       rot = {2} AND actual_cost < {3})
              ORDER BY rot, actual_cost DESC
              LIMIT 1§[pivot_x, pivot_y, slope, cost];
  GOTO inter3
}

BLOCK inter3 {
  IF §{0} IS NULL§[current]
  THEN GOTO truthy0
  ELSE GOTO falsey0
}

BLOCK truthy0 {
  EMIT §{0}§[well];
  GOTO inter4
}

BLOCK falsey0 {
  GOTO merge0
}

BLOCK inter4 {
  STOP
}

BLOCK merge0 {
  slope <- §{0}.slope§[current];
  GOTO inter5
}

BLOCK inter5 {
  cost <- §{0}.cost§[current];
  GOTO inter6
}

BLOCK inter6 {
  yield <- §{0} + {1}§[yield, cost];
  GOTO inter7
}

BLOCK inter7 {
  IF §{0} IS NULL OR {1} > {0}.yield§[well, yield]
  THEN GOTO truthy1
  ELSE GOTO falsey1
}

BLOCK truthy1 {
  well <- §{{x: {0}.x, y: {0}.y, yield: {1}}}§[current, yield];
  GOTO inter8
}

BLOCK inter8 {
  GOTO merge1
}

BLOCK falsey1 {
  GOTO merge1
}

BLOCK merge1 {
  JUMP loop_head
}
```