redhat

Loop Strip Mining in C2

Roland Westrelin

Motivations:

- Counted loops have no safepoints by default: hurts low latency GCs
- -XX:+UseCountedLoopSafepoints lets one safepoint per iteration: hurt throughput
- Loop strip mining should be best of both worlds

Loop Strip Mining

```
for (int i = start; i < stop; i += inc) {
  // body
}
```

↓

```
i = start;
if (i < stop) {
  do {
    int next = MIN(stop, i+LoopStripMiningIter*inc);
    do {
      // body
      i += inc;
    } while (i < next);
    safepoint();
  } while (i < stop);
}
```

Implementation

- When CountedLoop is built, construct skeleton OuterStripMinedLoop around it:
  - No extra data edges to not disrupt loop opts
  - Most optimizations simply step over OuterStripMinedLoop
  - Loop cloning requires extra care
- Once loop opts are over:
  - Inner loop limit adjusted
  - Data edges for outer loop are added

Limits

- If inner loop runs for few iterations on average, cost of outer loop may hurt performance
- Profiling doesn’t really help because it gives average number of iterations
- Even if we had max number of iterations from profiling, no guarantee the loop doesn’t run for more iterations