Notes on Kahan's *A Distillation Program*

Gideon Yuval

ABSTRACT

In the 8th set of Kahan course notes, there was a 1-page entry "A Distillation Program" (entry 3.5). My copy was almost illegible, which did not help any in trying to understand it.

The following is a recapitulation of the ideas in that Kahan note. It is more legible, and may or may not be clearer.

Exact 2-operand addition: Let Z be p+q (machine precision, rounded), abs(p)>abs(q). Let z be (p-Z)+q (operations done at machine precision). Then Z+z is exactly p+q; and z is as small as possible, among all pairs Z+z for which this holds.

Exact N-operand addition: We try to span the nonzero bits of the exact N-operand sum with (approximately) as few terms as possible. This is called "distillation". We use two data-structures for this: one ("the list") holds all current terms, in order of absolute value. The other ("the heap") other holds pairs of successive list-entries, sorted in order of floor(log2(p))-floor(log2(q)). In fact, all we need in the 2nd data-structure is the heap-property, i.e. the ability to find its smallest member fast.

A balanced tree (2-3 or AVL) will do fine for both data-structures (logN cost for any operation).

We can assume all entries are initially nonzero; otherwise, we just delete the zeros.

```
Repeat
 find smallest number in heap. Locate the two terms p,q, in list that
    correspond to it.
 Convert p+q into Z+z, as above.
 Delete p and q from list (and update the heap accordingly)
 Insert Z into list       (updating the heap accordingly)
 if z!=0, insert in list  (updating the heap accordingly)
Until "convert" step is a no-op.
```

Termination: At every step, the values in the list become "more unequal": their sum remains constant, but the new list is "more exact" in the following sense: there are numbers such that

Σ Xi over |Xi|>Z

becomes more accurate after the transformation than it was before. But the reverse does not occur.

Since this "more accurate" relation is a strict ordering, the iteration must terminate.

Final state: In the final state of the list, it contains N floating-point numbers, whose bit-positions never overlap (for otherwise, p+q->Z+z would not have been a no-op). This number is thus within one of the optimum.