On the robustness of the 2Sum and Fast2Sum algorithms

Sylvie Boldo Stef Graillat Jean-Michel Muller

Toulouse, May 2016

https://hal-ens-lyon.archives-ouvertes.fr/ensl-01310023

2Sum and Fast2Sum

- Kahan/Dekker and Møller/Knuth;
- used (implicitely or explicitely) in most compensated algorithms (such as compensated summation, dot product or polynomial evaluation), for manipulating Floating-Point expansions, etc.
- presented assuming round-to-nearest (RN), and no overflow;
- much more robust than usually believed: the result makes sense even when the rounding function is not RN, and they are *almost* immune to overflow.

First (implicit) use of Fast2Sum

In the following program S2 is an estimate of the error caused when S = T was last rounded or truncated, and is used in statement 13 to compensate for that error. The parentheses in statement 23 must not be omitted; they cause the difference (S−T) to be evaluated first and hence, in most cases, without error because the difference is normalized before it is rounded or truncated.

```
 1  S = 0.0
    S2 = 0.0
 2  DO 4 I = 1, N
 3  YI = ...
13  S2 = S2 + YI
    T = S + S2
23  S2 = (S−T) + S2
 4  S = T
 5  ....
```

Until double-precision arithmetic was made a standard feature of the FORTRAN language, the author and his students used this trick on a 7090 in FORTRAN II programs to perform quadrature, solve differential equations and sum infinite series.

- Kahan, Comm. ACM, Jan. 1965;
- error of the addition $S + S_2$, re-injected in the calculation.

First appearance of Twosum

QUASI DOUBLE-PRECISION IN FLOATING POINT ADDITION 43

and making allowance for the specific treatment of $\text{abs}(v) \leqq \frac{1}{2}E_u$ as e.g. takes place in the GIER computer, we conclude that provided $|v| \leqq |u|$ c gives the exact correction sought for except in the following special cases:

a1 u: $10.0000 \sim -2$ and $-E_u < v < -\frac{1}{2}E_u$ (consequently $\text{lev}(s) > \text{lev}(u)$) causing the error to be the last figure of v.

a2 u: $01.11..1xx..x1$
v: $00.00..011..11x_1x_2x_3\ldots x_f$ (and consequently $\text{lev}(s) > \text{lev}(u)$) in which cases the last figure of v, x_f (sub f for finis), is lost.

Even more briefly we can state that in the rare cases in which Process A is incorrect the error is a cut away of the last figure of v and will never exceed $\frac{1}{2}E_u$. And when the error equals $\frac{1}{2}E_u$ the result is equivalent to using no Process A ($c = 0$), i.e. we never do things worse than the bare addition $s := u + v$.

It should be noticed that only when $\text{lev}(s) > \text{lev}(u)$ and combination III or IV occurs need we evaluate eu; in all other cases $c := ev$ is sufficient.

The process could be put in a compact form by writing

$$s := u + v;$$
$$c := \big(v - (s - u)\big) + \big(u - (s - (s - u))\big);$$

For the sake of completeness we shall present an extension of (3) which masters cases a1, a2.

What we do is to evaluate $vlabi := v - (v1 + ev)$ ($vlabi$ for last bit of v) and add this quantity to c:

$$c := ev + eu + vlabi;$$

- Møller, *Quasi double-precision in floating-point addition*, BIT, 1965;
- c is the error of the addition $s := u + v$.

Error of a FP addition

- barring overflow, the error of a rounded-to-nearest FP addition or subtraction *is a FPN;*
- RN is necessary: radix-2, precision-p, rounding toward $-\infty$, if $a = 1$ and $b = -2^{-3p}$, then

$$s = \mathrm{RD}(a+b) = 0.\underbrace{111111\cdots11}_{p} = 1 - 2^{-p},$$

and

$$a + b - s = \underbrace{1.1111111111\cdots11}_{2p} \times 2^{-p-1},$$

$\rightarrow$ cannot be exactly represented with precision p.

- directed rounding functions are very useful (upper/lower bounds, stochastic arithmetic);
- a piece of code designed with RN in mind may be used in another context, willingly or not;
- furthermore,

  Let a and b be binary, precision-p, FPNs. Let $s \in \{\mathrm{RD}(a+b), \mathrm{RU}(a+b)\}$. If $|e_a - e_b| \leq p-1$, then $s-(a+b)$ is a binary, precision-p, FPN.

- even when the error of $+$ is not a FPN, can we get a good approximation to that error ? Would frequently suffice;
- partial answers: Demmel & Nguyen; Graillat, Jézéquel, & Picot (Fast2Sum), Martin-Dorel et al. (RN with possible double roundings);
- can we have spurious overflows ?

Some notation

- radix-2, precision-p, FP arithmetic, of extremal exponents $e_{\min}$ and $e_{\max}$ (set F_p);
- Ω = largest representable FPN:

$$\Omega = (2 - 2^{1-p}) \cdot 2^{e_{\max}}.$$

- FP predecessor and successor of x: $\text{pred}(x)$ and $\text{succ}(x)$;
- if $x \in \mathbb{R}$, $2^k \leq |x| < 2^{k+1}$,

$$\text{ulp}(x) = 2^{\max(k, e_{\min}) - p + 1}.$$

- rounding functions RN, RD, RU, RZ (for RN the choice of the tie-breaking rule is not important);
- the FP number $\hat{x}$ is a faithful rounding of $x \in \mathbb{R}$ if $\hat{x} \in \{\text{RD}(x), \text{RU}(x)\}$.

Rounding functions

Definition 1 (Rounding function—“optimal rounding” in [Kulisch71])

Function $\circ$ from $\mathbb{R}$ to F_p is a rounding function if

- $\forall x \in F_p, \circ(x) = x;$
- $\forall (x, y) \in \mathbb{R}^2, x \leq y \Rightarrow \circ(x) \leq \circ(y).$

Remark 2

If $\circ$ *is a rounding function, then for any* x, $\circ(x) \in \{\mathrm{RD}(x), \mathrm{RU}(x)\}.$

Introduced by Dekker in 1971 (yet used by Kahan in 1965).

Algorithm 1: Conventional Fast2Sum Algorithm.

1: $s \leftarrow \mathrm{RN}(a+b)$
2: $z \leftarrow \mathrm{RN}(s-a)$
3: $t \leftarrow \mathrm{RN}(b-z)$

In the absence of overflow, if the radix of the FP system is ≤ 3, and if the FP exponents e_a and e_b of a and b satisfy $e_a \geq e_b$, then

$$s+t=a+b.$$

TwoSum

Knuth (1969?) transforms Møller′ trick (1965) into a Theorem.

Algorithm 2: Conventional 2Sum algorithm.

$s \leftarrow \mathrm{RN}(a+b)$
$a' \leftarrow \mathrm{RN}(s-b)$
$b' \leftarrow \mathrm{RN}(s-a')$
$\delta_a \leftarrow \mathrm{RN}(a-a')$
$\delta_b \leftarrow \mathrm{RN}(b-b')$
$t \leftarrow \mathrm{RN}(\delta_a+\delta_b)$

In the absence of overflow,

$$s+t=a+b.$$

Without knowing the respective orders of magnitude of a & b, calling 2Sum in general more efficient than comparing them, swapping them if needed, and calling Fast2Sum.

We won't say anything about underflows

Theorem 3 (Hauser)

If x and y are radix-β FP numbers, and if the number $\mathrm{RN}(x+y)$ is subnormal, then $x+y$ is a FP number (so that $\circ(x+y) = x+y$ for any rounding function).

Proof. x and y are multiples of the smallest nonzero FPN $\alpha = \beta^{e_{\min}-p+1} \rightarrow x+y$ is a multiple of α. If it is in the subnormal range, then it is $< \beta^{e_{\min}} \Rightarrow$ exactly representable.

What we are going to discuss...

Algorithm 3: Fast2Sum with faithful roundings: $\circ_1, \circ_2, \circ_3$ are rounding functions.

1: $s \leftarrow \circ_1(a+b)$
2: $z \leftarrow \circ_2(s-a)$
3: $t \leftarrow \circ_3(b-z)$

Algorithm 4: 2Sum with faithful roundings: $\circ_i$, for $i = 1, \ldots, 6$, are rounding functions.

1: $s \leftarrow \circ_1(a+b)$
2: $a' \leftarrow \circ_2(s-b)$
3: $b' \leftarrow \circ_3(s-a')$
4: $\delta_a \leftarrow \circ_4(a-a')$
5: $\delta_b \leftarrow \circ_5(b-b')$
6: $t \leftarrow \circ_6(\delta_a + \delta_b)$

Some preliminary results

Lemma 4 (Sterbenz)

If x and y are finite floating-point numbers such that

$$\frac{y}{2} \leq x \leq 2y,$$

then $x - y$ is a FP number.

Lemma 5

Let a and b be two binary FPN of exponents e_a and e_b. Let $s \in \{\mathrm{RD}(a+b), \mathrm{RU}(a+b)\}$. If the exponent e_s of s is $\leq \min(e_a, e_b)$ then $s = a + b$.

Proof. a and b are multiple of $2^{e_a - p+1}$ and $2^{e_b - p+1}$, respectively. Since $e_s \leq \min(e_a, e_b)$, $a + b$ is a multiple of $2^{e_s - p+1}$. By rounding it (through any rounding function) to a multiple of $2^{e_s - p+1}$ we just get it.

Accuracy of Fast2Sum assuming no overflow

1: $s \leftarrow \circ_1(a+b)$
2: $z \leftarrow \circ_2(s-a)$
3: $t \leftarrow \circ_3(b-z)$

Lemma 6

Let a and b be two binary FPNs, with $e_a \geq e_b$. Let $s \in \{\mathrm{RD}(a+b), \mathrm{RU}(a+b)\}$. The number $s-a$ is a FP number ($\rightarrow$ computed exactly, with any rounding function).

Proof. Adaptation of Dekker's original proof. If $|b| \leq |a|$ the proof becomes straightforward: assuming $a \geq 0$ (symmetry), we have $-a \leq b \leq a$, and

- if $-a \leq b \leq -a/2$ then Sterbenz $\Rightarrow s = a+b \Rightarrow s-a = b$ is a FPN;
- if $-a/2 \leq b \leq a$ then $a/2 \leq a+b \leq 2a$ hence $a/2 \leq s \leq 2a \Rightarrow s-a$ is aFPN.

Accuracy of Fast2Sum assuming no overflow

1: $s \leftarrow \circ_1(a + b)$
2: $z \leftarrow \circ_2(s - a)$
3: $t \leftarrow \circ_3(b - z)$

Lemma 6 only holds in radix 2.
Example: In radix 3 with $p = 4$ and $\circ_i = \text{RU}$, if $a = 1002_3 = 29_{10}$ and $b = 2222_3 = 80_{10}$, then

$$s = \text{RU}(a + b) = 11010_3 = 111_{10},$$

so that $s - a = 10001_3 = 82_{10}$ is not exactly representable with precision 4.

1: $s \leftarrow \circ_1(a+b)$
2: $z \leftarrow \circ_2(s-a) = s - a$
3: $t \leftarrow \circ_3(b-z) = \circ_3(b-(s-a)) = \circ_3((a+b)-s)$

Theorem 7

If no overflow occurs, and $e_a \geq e_b$ then the values s and t returned by Algorithm 3 satisfy

$$t = \circ_3((a+b)-s),$$

i.e., t is a faithful rounding of the error of the FP addition $s \leftarrow \circ_1(a+b)$.

$\rightarrow$ if the difference of the exponents of a and b does not exceed $p-1$ (will occur in many practical cases), then $t = (a+b) - s$.

2Sum: more tricky…

1: $s \leftarrow \circ_1(a+b)$
2: $a' \leftarrow \circ_2(s-b)$
3: $b' \leftarrow \circ_3(s-a')$
4: $\delta_a \leftarrow \circ_4(a-a')$
5: $\delta_b \leftarrow \circ_5(b-b')$
6: $t \leftarrow \circ_6(\delta_a+\delta_b)$

---

t not always faithful rounding of $(a+b)-s$:
$p = 24$, $a = 3076485 \cdot 2^{-21}$, $b = -6130317 \cdot 2^{-49}$,
$\circ_1 = \circ_2 = \circ_5 = \text{RU}$, $\circ_3 = \circ_4 = \circ_6 = \text{RD}$, gives

$$\begin{aligned} s &= a = 3076485 \cdot 2^{-21} \rightarrow (a+b)-s = b; \\ t &= -1532579 \cdot 2^{-47}; \end{aligned}$$

With any rounding function $\circ$, $\circ((a+b)-s) = b \neq t$.
However, $(a+b-s)-t = -2^{-49} \rightarrow t$ remains a very good approximation to $(a+b)-s$.

2Sum: more tricky...

1: $s \leftarrow \circ_1(a+b)$
2: $a' \leftarrow \circ_2(s-b)$
3: $b' \leftarrow \circ_3(s-a')$
4: $\delta_a \leftarrow \circ_4(a-a')$
5: $\delta_b \leftarrow \circ_5(b-b')$
6: $t \leftarrow \circ_6(\delta_a+\delta_b)$

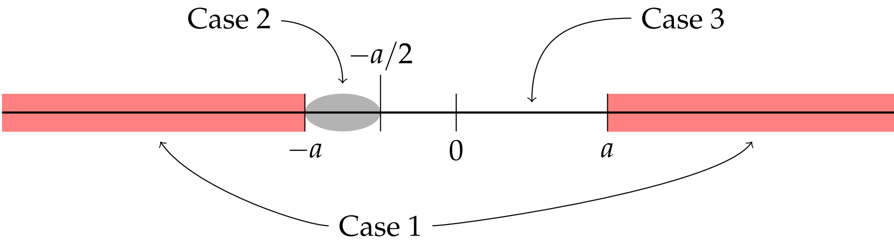
Theorem 8

If $p \geq 4$ and no overflow occurs, then s and t satisfy

$$t = (a+b) - s + \alpha,$$

with $|\alpha| < 2^{-p+1} \cdot \mathrm{ulp}(a+b) \leq 2^{-p+1} \cdot \mathrm{ulp}(s)$. Furthermore, if e_s and e_b satisfy $e_s - e_b \leq p-1$ then t is a faithful rounding of $(a+b)-s$.

Case splitting based on the location of b



An easy case: $|b| \geq a$

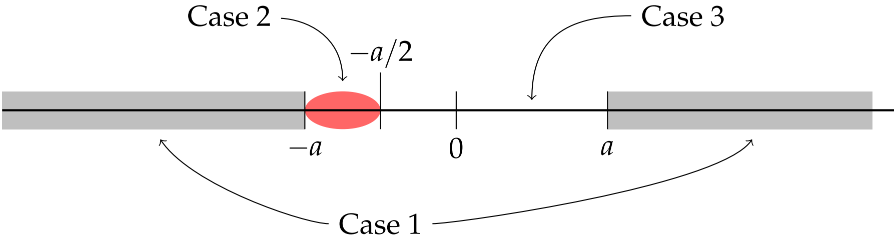
1: $s \leftarrow \circ_1(a+b)$
2: $a' \leftarrow \circ_2(s-b)$
3: $b' \leftarrow \circ_3(s-a')$
4: $\delta_a \leftarrow \circ_4(a-a')$
5: $\delta_b \leftarrow \circ_5(b-b')$
6: $t \leftarrow \circ_6(\delta_a+\delta_b)$

$|b| \geq a \Rightarrow$ lines (1), (2), and (4) of Algorithm 4 are Fast2Sum(b,a).

$\rightarrow$ we have $a' = s - b$ and $\delta_a = \circ_4(a+b-s)$. An immediate consequence of $a' = s - b$ is $b' = b$ and $\delta_b = 0$. From this, we find

$$t = \circ_4(a+b-s)$$

A straightforward case: $-a < b \leq -a/2$



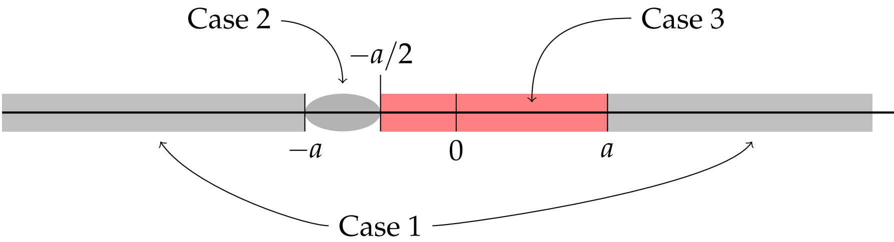
A straightforward case: $-a < b \leq -a/2$

1: $s \leftarrow \circ_1(a+b)$
2: $a' \leftarrow \circ_2(s-b)$
3: $b' \leftarrow \circ_3(s-a')$
4: $\delta_a \leftarrow \circ_4(a-a')$
5: $\delta_b \leftarrow \circ_5(b-b')$
6: $t \leftarrow \circ_6(\delta_a+\delta_b)$

Sterbenz Lemma $\rightarrow s = a + b$. Successively implies $a' = a$, $b' = b$, $\delta_a = \delta_b = t = 0$, so that

$$t = (a+b) - s.$$

We are left with the painful part: $-a/2 < b < a$



We are left with the painful part: $-a/2 < b < a$

$s \leftarrow \circ_1(a+b)$
$a' \leftarrow \circ_2(s-b)$
$b' \leftarrow \circ_3(s-a')$
$\delta_a \leftarrow \circ_4(a-a')$
$\delta_b \leftarrow \circ_5(b-b')$
$t \leftarrow \circ_6(\delta_a+\delta_b)$

Let $u = 2^{1-p}$. We have

$$\begin{array}{rcl} s & = & (a+b)\cdot(1+\epsilon_1);\ \text{with } |\epsilon_1| \le u; \\ a' & = & (s-b)\cdot(1+\epsilon_2);\ \text{with } |\epsilon_2| \le u. \end{array}$$

$$\rightarrow a' = (a + a\epsilon_1 + b\epsilon_1)\cdot(1+\epsilon_2).$$

$|b| < a \rightarrow a\epsilon_1 + b\epsilon_1 = 2a\epsilon_3$, with $|\epsilon_3| \le u$. Therefore $a' = a\cdot(1+\eta)$, with $|\eta| \le 3u + 2u^2$. As soon as $p \ge 4$, $|\eta| < 1/2$ so that $a/2 \le a' \le 2a \rightarrow \delta_a = a - a'$ by Sterbenz Lemma.

We are left with the painful part: $-a/2 < b < a$

1: $s \leftarrow \circ_1(a+b)$
2: $a' \leftarrow \circ_2(s-b)$
3: $b' \leftarrow \circ_3(s-a')$
4: $\delta_a \leftarrow \circ_4(a-a') = a-a'$
5: $\delta_b \leftarrow \circ_5(b-b')$
6: $t \leftarrow \circ_6(\delta_a+\delta_b)$

$s \geq |b| \rightarrow$ Lines (2), (3), and (5) are equivalent to Fast2Sum$(s, -b)$, so that

$$b' = s - a', \tag{1}$$

and

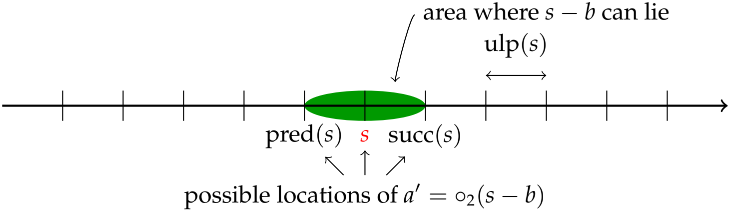
$$\delta_b = \circ_5(a' - (s-b)). \tag{2}$$

If $e_s - e_b \leq p-1$, then (2) implies $\delta_b = a' - (s-b)$, from which we deduce $t = \circ_6(a+b-s)$.

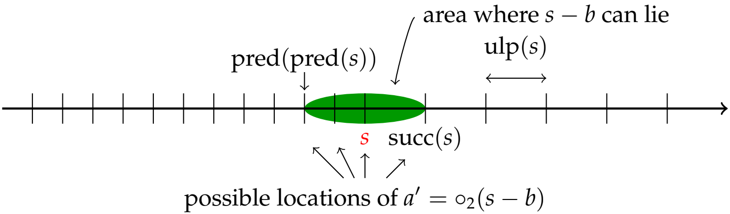
We are left with $-a/2 < b < a$ and $e_s - e_b \geq p$

$s \leftarrow \circ_1(a+b)$
$a' \leftarrow \circ_2(s-b)$
$b' \leftarrow \circ_3(s-a') = s - a'$
$\delta_a \leftarrow \circ_4(a-a') = a - a'$
$\delta_b \leftarrow \circ_5(b-b')$
$t \leftarrow \circ_6(\delta_a + \delta_b)$

$e_s - e_b \geq p \Rightarrow |b| < \mathrm{ulp}(s) \Rightarrow a' \in \{\mathrm{succ}(s), s, \mathrm{pred}(s), \mathrm{pred}(\mathrm{pred}(s))\}$, and the case $a' = \mathrm{pred}(\mathrm{pred}(s))$ can occur only when s is a power of 2.



General case: s is not a power of 2



Special case: s is a power of 2

We are left with $-a/2 < b < a$ and $e_s - e_b \geq p$

$s \leftarrow \circ_1(a+b)$
$a' \leftarrow \circ_2(s-b)$
$b' \leftarrow \circ_3(s-a') = s - a'$
$\delta_a \leftarrow \circ_4(a-a') = a - a'$
$\delta_b \leftarrow \circ_5(b-b')$
$t \leftarrow \circ_6(\delta_a + \delta_b)$

Remember: $a' \in \{\mathrm{succ}(s), s, \mathrm{pred}(s), \mathrm{pred}(\mathrm{pred}(s))\}$,

1. **If $\mathbf{a'} = \mathbf{s}$ then** $b' = 0$. It follows that $\delta_b = b$ and $\delta_a = a - s$, for which we deduce $t = \circ_6(a + b - s)$,
2. **If $\mathbf{a'} \neq \mathbf{s}$ then**

$$a' = s - \sigma \cdot \mathrm{ulp}(s), \text{with } \sigma \in \{-1, +1/2, +1\},$$

and we have

$$b' = \sigma \cdot \mathrm{ulp}(s); \delta_a = a - s + \sigma \cdot \mathrm{ulp}(s); \text{ and } \delta_b = \circ_5(b - \sigma \cdot \mathrm{ulp}(s)).$$

$-a/2 < b < a$, $e_s - e_b \geq p$, and $a' = s - \sigma \cdot \mathrm{ulp}(s)$

1: $s \leftarrow \circ_1(a+b)$
2: $a' \leftarrow \circ_2(s-b) = s - \sigma \cdot \mathrm{ulp}(s)$
3: $b' \leftarrow \circ_3(s-a') = s - a' = \sigma \cdot \mathrm{ulp}(s)$
4: $\delta_a \leftarrow \circ_4(a-a') = a - a'$
5: $\delta_b \leftarrow \circ_5(b-b') = \circ_5(b - \sigma \cdot \mathrm{ulp}(s))$
6: $t \leftarrow \circ_6(\delta_a + \delta_b)$

Remember: $|b| < \mathrm{ulp}(s)$. Furthermore, b has the same sign as σ. Therefore

- either $|b| \geq |\sigma|/2 \cdot \mathrm{ulp}(s)$, in which case Sterbenz Lemma $\rightarrow \delta_b = b - \sigma \cdot \mathrm{ulp}(s) \Rightarrow \delta_a + \delta_b = a + b - s \Rightarrow t = \circ_6(a+b-s)$
- or $|b| < |\sigma|/2 \cdot \mathrm{ulp}(s)$, in which case, from

$$|b - \sigma \cdot \mathrm{ulp}(s)| < |\sigma| \cdot \mathrm{ulp}(s)$$

(unless $b = 0$ but that case is straightforwardly handled), we get (since $|\sigma| \cdot \mathrm{ulp}(s)$ is a power of 2),

$$|\delta_b - (b - \sigma \cdot \mathrm{ulp}(s))| < \frac{1}{2}\mathrm{ulp}(\sigma \cdot \mathrm{ulp}(s)) = \frac{|\sigma|}{2}\mathrm{ulp}(\mathrm{ulp}(s)) = |\sigma| \cdot 2^{-p}\mathrm{ulp}(s$$

$-a/2 < b < a$, $e_s - e_b \geq p$, and $a' = s - \sigma \cdot \mathrm{ulp}(s)$

1: $s \leftarrow \circ_1(a+b)$
2: $a' \leftarrow \circ_2(s-b) = s - \sigma \cdot \mathrm{ulp}(s)$
3: $b' \leftarrow \circ_3(s-a') = s - a' = \sigma \cdot \mathrm{ulp}(s)$
4: $\delta_a \leftarrow \circ_4(a-a') = a - a'$
5: $\delta_b \leftarrow \circ_5(b-b') = \circ_5(b - \sigma \cdot \mathrm{ulp}(s))$
6: $t \leftarrow \circ_6(\delta_a + \delta_b)$

Consequence:

$$\begin{array}{c} |(\delta_a + \delta_b) - (a+b-s)| < \ |\sigma| \cdot 2^{-p}\mathrm{ulp}(s). \\ |\delta_a + \delta_b| < \mathrm{ulp}(a+b) + |\sigma| \cdot 2^{-p}\mathrm{ulp}(s). \end{array} \quad (3)$$

- show that $\delta_a + \delta_b$ is a multiple of $|\sigma| \cdot 2^{-p}\mathrm{ulp}(s) \rightarrow |\delta_a + \delta_b| < \mathrm{ulp}(a+b)$;
- case splitting (is s a power of 2?, is it above or below $a+b$?)

$\rightarrow |t - (a+b-s)| < 2^{-p+1}\mathrm{ulp}(a+b)$.

Fast2Sum is immune to overflow

1: $s \leftarrow \circ_1(a+b)$ reminder: $e_a \geq e_b$
2: $z \leftarrow \circ_2(s-a)$
3: $t \leftarrow \circ_3(b-z)$

- assume no overflow at line 1;
- without l.o.g., assume $a > 0$;
- $s = a + b + \epsilon$, with $|\epsilon| < \text{ulp}(a+b) \leq 2\text{ulp}(a)$, hence

$$s - a = b + \epsilon$$

- therefore, if line (2) overflows then $b < -\Omega + 2\text{ulp}(a)$ or $b > \Omega - 2\text{ulp}(a)$
  - if $b < -\Omega + 2\text{ulp}(a)$ then $b < -\Omega + 2\text{ulp}(\Omega)$ and (since $e_a \geq e_b$), $\Omega/2 < 2^{e_{\max}} \leq a \leq \Omega$. Sterbenz Lemma $\rightarrow s = a + b \rightarrow z = b \rightarrow$ no overflow at line 2;
  - if $b > \Omega - 2\text{ulp}(a)$ is impossible: this and $e_a \geq e_b$ imply $a + b > \Omega/2 + \Omega - 2\text{ulp}(\Omega) \rightarrow$ line 1 overflows.

$\rightarrow$ Line 2 cannot overflow.

Fast2Sum is immune to overflow

1: $s \leftarrow \circ_1(a+b)$
2: $z \leftarrow \circ_2(s-a)$ no overflow $\rightarrow z = s - a$
3: $t \leftarrow \circ_3(b-z)$

Line 3 ?

Since line 2 does not overflow, $z = s - a$. Hence $b - z = (a+b) - s$, hence

$$|b - z| < |(a+b) - s| < \mathrm{ulp}(s) < |s|$$

$\rightarrow$ no overflow.

Theorem 9

Assuming $e_a \geq e_b$, if the computation of s (first line of Fast2Sum) does not overflow, then the other lines cannot overflow.

Of course, with 2Sum it is more tricky

1: $s \leftarrow \circ_1(a + b)$
2: $a' \leftarrow \circ_2(s - b)$
3: $b' \leftarrow \circ_3(s - a')$
4: $\delta_a \leftarrow \circ_4(a - a')$
5: $\delta_b \leftarrow \circ_5(b - b')$
6: $t \leftarrow \circ_6(\delta_a + \delta_b)$

Theorem 10

If $|a| < \Omega$ and if there is no overflow at line (1) of the algorithm, then there will be no overflow at lines (2) to (6).

Condition $|a| < \Omega$ is necessary. Assume all rounding functions are RN (ties-to-even). The choice $a = \Omega$ and $b = -(3/2) \cdot \mathrm{ulp}(\Omega)$ gives no overflow at line (1), and an overflow at line (2).

On the list stds-754@IEEE.ORG

```
De: Jason Riedy <jason.riedy@cc.gatech.edu>
Objet: Updated twoSum proposal
Date: 18 mai 2016 17:15:58 UTC+2

Attached is an updated twoSum proposal that provides more
thorough justification for exceptional behavior.
There also is some example C code for context,
although that code ignores the rounding mode.

...
                    -----------------------------------
                     PROPOSAL: TWOSUM OPERATION
                            Jason Riedy
                    -----------------------------------



Table of Contents
-------------------
...
.. 2.1 Properties of an existing software implementation
```

On the list stds-754@IEEE.ORG

```
De: Jason Riedy <jason.riedy@cc.gatech.edu>
Objet: Updated twoSum proposal
Date: 18 mai 2016 20:11:35 UTC+2

And Jean-Michel Muller writes:

> That analyses the behaviour of these algorithms with various
rounding modes, and shows that they are rather immune from
spurious overflow.

Thank you.  My "careful" version fail with your example.  augh
 I'll work on that.
```

Conclusion

In binary FP arithmetic, 2Sum and Fast2Sum are more "robust" than it is usually believed:

- even when the error of the initial FP addition is not a FP number, they return a very good approximation to that error ($\rightarrow$ can be used in many compensated algorithms);
- Fast2Sum totally immune to overflow;
- 2Sum almost totally immune to overflow: the only case where a "spurious" overflow may occur is when the absolute value of a is equal to Ω.