Idee su errore
e precisione arbitraria

`github.com/nrizzo`

8 luglio 2018

1 Introduzione

Nello studio del calcolo della variante di Ackermann su insiemi ereditariamente finiti (e non) definita come

$$\mathbb{R}_A(x) = \sum_{y \in x} 2^{-\mathbb{R}_A(y)},$$

sorgono problemi di analisi numerica, tra i quali come modellare l'incertezza su numeri reali e la rappresentazione di un calcolatore di numeri razionali a precisione arbitraria.

2 Considerazioni su errore e precisione

Definizione 1. Dati $x, \tilde{x} > 0$, diciamo che $\tilde{x}$ è un numero razionale che approssima (in modo forte) x fino a $k \in \mathbb{N}$ cifre binarie dopo la virgola se

$$\tilde{x} = x - e_x, \qquad \text{con} \quad 0 \leq e_x < 2^{-k} \quad \text{e} \quad \tilde{x} = \frac{j}{2^k} \;\; j \in \mathbb{N},$$

cioè $\tilde{x}$ è un'approssimazione per difetto.

Tuttavia, questo modello risulta poco maneggevole nel calcolare i risultati di operazioni aritmetiche tra dati approssimati: sommando due approssimazioni, il risultato può essere più incerto dei dati di partenza. La prossima definizione è più generale, ma perde in precisione.

Definizione 2. Dati $x, \tilde{x}, e_x^{\max} > 0$ con $x_{\min}$, $x_{\max}$ razionali tali che $x_{\min} \leq x_{\max}$, diciamo che $\tilde{x}$ approssima (in modo debole) x se e solo se

$$\tilde{x} = x - e_x, \qquad \text{con} \quad 0 \leq e_x < e_x^{\max}.$$

Però nel calcolo dell'approssimazione di $2^{-\tilde{x}}$ deve necessariamente venire introdotto **resto negativo**, poiché indipendentemente dall'algoritmo utilizzato un resto negativo compare nel resto di Lagrange o nel polinomio di MacLaurin. Una rappresentazione alternativa può basarsi su intervalli di numeri reali ed essere più comoda nella stima dell'errore.

Definizione 3. Diciamo che x è approssimato dall'intervallo (topologicamente) chiuso $X = [a, b]$ se e solo se $x \in X$.[1]

Nella teoria di Hansen [1], X è chiamato un numero intervallare, i valori precisi sono intervalli degeneri e viene notato che gli estremi possono non essere rappresentabili da un numero di macchina: in tal caso si effettua *outward rounding*, cioè l'approssimazione per difetto dell'estremo sinistro con il più grande numero di macchina più piccolo dell'estremo stesso, e l'approssimazione per eccesso dell'estremo destro con il più piccolo numero di macchina più grande di esso.

La definizione di intervalli che contengono i valori esatti semplifica il calcolo e lo studio dell'errore, poiché calcolando operazioni aritmetiche sugli estremi si maneggiano dati non affetti da errore.

Teorema 1. Dato $x \in X = [a, b]$, con $a, b \in \mathbb{R}$, il prefisso in comune delle rappresentazioni binaria dei due estremi (se esiste) è anche prefisso di x.

Teorema 2. Dato $x \in X$ e dato $Y \supseteq X$, il prefisso in comune tra gli estremi di Y (se esiste) è anche prefisso di x e degli estremi di X.

3 Principi di aritmerica intervallare

Dati $X = [a, b]$ e $Y = [c, d]$, se $\bullet$ denota una delle operazioni di addizione, sottrazione, moltiplicazione e divisione, allora il risultato dell'operazione corrispondente applicata a X e Y è

$$X \bullet Y = \{x \bullet y \mid x \in X, y \in Y\}.$$

3.1 Addizione e sottrazione

$$X + Y = [a + c, b + d]$$
$$X - Y = [a - d, b - c]$$

3.2 Immagine di funzione continua monotona

$$f(X) = \begin{cases} [f(a), f(b)] & \text{se } f \text{ è non decrescente} \\ [f(b), f(a)] & \text{se } f \text{ è non crescente} \end{cases}$$

4 Approssimazione della potenza di due con esponente negativo

In questa sezione studiamo lo schema per un possibile algoritmo di approssimazione, da parte di un calcolatore, del calcolo della potenza di due con esponente reale negativo rappresentato da un intervallo. Come numero di macchina

[1] Questa definizione per essere equivalente alla 2 deve avere l'estremo destro non incluso, e i calcoli da eseguire per risolvere il problema dell'approssimazione della codifica di Ackermann modificata "sbiadiscono" facilmente entrambi gli estremi: l'inclusione o meno di essi non causa problemi se ne viene tenuto conto.

supponiamo di avere a disposizione numeri razionali a precisione arbitraria, codificati come esponente e mantissa (di dimensione variabile) in base 2.

A grandi linee, dato X_0, l'algoritmo:

1. ne divide gli estremi in parte intera e frazionaria, spezzando così il calcolo della potenza di 2;
2. cambia la base delle potenze con esponente razionale da 2 a e (*outward rounding*);
3. approssima l'elevamento a potenza di e della parte frazionaria trasformata dei due estremi con un polinomio di MacLaurin, approssimando anche ad ogni operazione di moltiplicazione e divisione (*outward rounding*);
4. calcola gli estremi finali.

4.1 Parte intera e parte frazionaria

Per utilizzare al meglio il polinomio di MacLaurin il punto da calcolare deve essere vicino allo zero. Inoltre la divisione per una potenza di due è un'operazione semplice da eseguire in aritmetica di macchina (un `rshift`). Per semplicità[2], supponiamo

$$X_0 = \left[x_0 + x_{\min}, x_0 + x_{\max}\right] \qquad x_0 \in \mathbb{N}, \quad 0 \leq x_{\min} \leq x_{\max} < 1,$$

$$X_1 = X_0 - x_0 = [x_{\min}, x_{\max}];$$

allora l'operazione diventa

$$2^{-X} = 2^{x_0 + X_1} = 2^{-x_0} \cdot 2^{-X_1} = \left[2^{-x_0} \cdot 2^{-x_{\max}}, 2^{-x_0} \cdot 2^{-x_{\min}}\right]$$

4.2 Cambio di base

Il cambio a base e è di fatto obbligatorio, poiché nei successivi polinomi di MacLaurin compaiono potenze di $x_{\min} \log 2$ e di $x_{\max} \log 2$. In generale, definito $y = x \log 2$,

$$2^{-x} = e^{\log(2^{-x})} = e^{-x \log 2} = e^{-y},$$

quindi vanno calcolati $y_{\min}$ e $y_{\max}$ approssimando $x_{\min} \cdot \log 2$ per difetto e $x_{\max} \cdot \log 2$ per eccesso (*outward rounding*), con scarto $-\delta_{\min}$ e $\delta_{\max}$:

$$Y = [y_{\min}, y_{\max}] = [x_{\min} \cdot \log 2 - \delta_{\min},\ x_{\max} \cdot \log 2 + \delta_{\max}],$$

$$0 \leq y_{\min} \leq y_{\max} < \log 2,$$

$$Y \supseteq X_1$$

Il valore da calcolare va aggiornato quindi a

$$2^{-x_0} \cdot e^{-Y} \supseteq 2^{-X}.$$

[2] L'accuratezza di almeno la parte intera del dato sembrerebbe una richiesta legittima dato il problema da risolvere; anche se non è il caso, l'algoritmo è valido comunque.

4.3 Polinomio di MacLaurin

Gli elevamenti a potenza $e^{-y_{\max}}$ ed $e^{-y_{\min}}$ vengono approssimati dal polinomio di MacLaurin g di cui vanno implementate due versioni, che approssimino una per eccesso e una per difetto: $\tilde{g}_{\text{ecc}}$ e $\tilde{g}_{\text{dif}}$. Il calcolo finale sarà quello di

$$\left[2^{-x_0}\tilde{g}_{\text{dif}}(y_{\max}), 2^{-x_0}\tilde{g}_{\text{ecc}}(y_{\min})\right] \supseteq 2^{-x_0} \cdot e^{-Y}.$$

4.3.1 Horner

L'algoritmo per g potrebbe essere quello indicato in figura, in pseudocodice, ricalcando l'algoritmo di Horner per il calcolo di un polinomio dal suo fondo.

```
0  reciprocal_exp_aux(y,n)
1  {
2          res = 0;
3          for (i = n; i > 0; i--) {
4                  if (i%2 == 0)
5                          res += +1;
6                  else
7                          res += -1;
8
9                  res *= y;
10                 res /= i;
11         }
12         res += 1;
13
14         return res;
15 }
```

Figura 1:

Computazione parziale "migliorabile"? Partire dal fondo del polinomio di Maclaurin permette di risparmiare operazioni e rendere il calcolo della serie lineare rispetto al grado del polinomio, però per implementare le due versioni di g le operazioni devono essere tutte per eccesso o per difetto. Si potrebbe mantenere circa lo stesso numero di operazioni semplici partendo anche dai primi addendi, potendo rendere il calcolo "migliorabile in seguito", salvandosi l'ultimo addendo calcolato, ma bisognerebbe maneggiare due copie: l'approssimazione per difetto e l'approssimazione per eccesso, perché gli addendi hanno segno alternato.

4.3.2 Quanto e dove approssimare?

Le operazioni in cui si restituisce un risultato approssimato sono:

1. moltiplicazione (si può);
2. divisione per intero (spesso si deve);

ed entrambe possono essere troncate o arrotondate ad una precisione arbitraria, per essere approssimate rispettivamente per difetto o per eccesso.

Segue uno schema del calcolo di $\tilde{g}_{\text{dif}}(y_{\max})$, in cui, in ogni passo della computazione, a sinistra del simbolo di uguaglianza c'è il risultato parziale effettivamente calcolato, a destra la somma del valore corretto con gli scarti accumulati. Perché sia un'approssimazione per difetto, il resto di Lagrange deve essere **positivo**, cioè il grado del polinomio di MacLaurin deve essere dispari (allo stesso modo nel calcolo di $\tilde{g}_{\text{ecc}}(y_{\min})$ il grado del polinomio sarà pari).

Approssimazione (difetto)	Approssimato e resto	Prossima operazione
-1	$= -1$	$\times y_{\max}$
$a(-y_{\max})$	$= -y_{\max}$	$/n$
$a\Big(-\frac{y_{\max}}{n}\Big)$	$= -\frac{y_{\max}}{n} - \alpha_n$	$+1$
$a\Big(-\frac{y_{\max}}{n} + 1\Big)$	$= -\frac{y_{\max}}{n} + 1 - \alpha_n$	$\times y_{\max}$
$a\Big(-\frac{y_{\max}^2}{n} + y_{\max}\Big)$	$= -\frac{y_{\max}^2}{n} + y_{\max} - \alpha_n \cdot y_{\max} - \beta_n$	$/(n-1)$
$a\Big(-\frac{y_{\max}^2}{n(n-1)} + \frac{y_{\max}}{n-1}\Big)$	$= -\frac{y_{\max}^2}{n(n-1)} + \frac{y_{\max}}{n-1} - \frac{\alpha_n}{n-1} y_{\max} - \frac{\beta_n}{n-1}$	-1
	$\vdots$	
$a\Big(\sum_{i=0}^{n}(-1)^i \frac{(y_{\max})^i}{(i)!}\Big)$	$= \sum_{i=0}^{n}(-1)^i \frac{(y_{\max})^i}{(i)!} - \sum_{i=2}^{n} \frac{(y_{\max})^{i-1}}{(i-1)!}\alpha_i - \sum_{i=2}^{n} \frac{(y_{\max})^{i-2}}{(i-1)!}\beta_i$	–
	$= T_n(y_{\max}) - \sum_{i=2}^{n} \frac{(y_{\max})^{i-1}}{(i-1)!}\alpha_i - \sum_{i=2}^{n} \frac{(y_{\max})^{i-2}}{(i-1)!}\beta_i$	–

Dove a rappresenta una funzione che approssima per difetto e α_i e β_i sono gli errori introdotti nell'approssimazione di, rispettivamente, l'$n-i$-esima moltiplicazione e della divisione per i. Maggiorando il valore assoluto di questi errori con α e β, e notando che le rispettive sommatorie sono maggiorate dal polinomio di MacLaurin per $y = 2^x$,

$$a\Big(\sum_{i=0}^{n}(-1)^i \frac{(y_{\max})^i}{(i)!}\Big) = \tilde{g}_{\text{dif}}(y_{\max}) > T_n(y_{\max}) - \alpha \cdot e^{y_{\max}} - \beta \cdot e^{y_{\max}},$$

e poiché $y_{\max} \in [0, \log 2)$,

$$\tilde{g}_{\text{dif}}(y_{\max}) > T_n(y_{\max}) - 2(\alpha + \beta).$$

Quest'ultima disequazione mostra che il polinomio di MacLaurin si può approssimare con precisione arbitraria, a seconda della precisione delle operazioni di moltiplicazione e divisione per intero.

5 Conclusione

In conclusione, i calcoli importanti da considerare sugli intervalli sono la somma e l'elevamento a potenza di due con esponente cambiato di segno. Per quest'ultimo dovrebbe valere

$$2^{-X} \subseteq \left(2^{-x_0} \cdot \left(\frac{2^{-x_{\max}}}{e^{\delta_{\max}}} - \frac{(\log 2 \cdot x_{\max})^{2n+1}}{(2n+1)!} - 2(\alpha+\beta) \right), \right.$$
$$\left. 2^{-x_0} \cdot \left(2^{-x_{\min}} \cdot e^{\delta_{\min}} + \frac{(\log 2 \cdot x_{\min})^{2m}}{(2m)!} + 2(\alpha+\beta) \right) \right),$$

con m e n naturali e $\delta_{\max}$, $\delta_{\min}$, α, β arbitrariamente piccoli tali che:

- $\delta_{\max}$ e $\delta_{\min}$ dipendono dall'accuratezza dell'approssimazione nel cambio a base e;
- n e m determinano il grado dei polinomi di MacLaurin;
- α e β sono una maggiorazione degli errori massimi compiuti troncando e arrotondando le operazioni di moltiplicazione e divisione per intero.

Sharpness e dependency In [1] Hansen tratta di *sharpness*, cioè la possibilità di calcolare risultati (intervallari) esatti o più precisi possibili dato il numero di macchina, e il problema della *dependency*, cioè degli errori evitabili risultanti dal mancato sfruttamento delle dipendenze tra variabili. Dai risultati ottenuti sembra che il calcolo della potenza di due con esponente reale negativo possa essere *sharp* (dopotutto tutte le approssimazioni hanno accuratezza arbitraria), ma nel contesto della codifica di Ackermann modificata ciò non è più vero a causa della *dependency*: l'albero delle dipendenze degli insiemi è inevitabilmente interconnesso e sicuramente si propagheranno errori. Un algoritmo che risolva questo problema non sembra essere di facile scrittura, mentre si potrebbe studiare l'influenza della *dependency*.

References

[1] *Global Optimization Using Interval Analysis*, Second Edition, Revised and Expanded, E. Hansen, G. W. Walster, Marcel Dekker Inc., 2004.