Computer arithmetic's – Final project documentation

Date: 28/4/02
Written by:
Eliav Gnessin
Yaron Gorovitz

Chapter 1: Introduction

The goal of the project is to design a software tool to compute the upper bound of the intermediate errors of Goldschmidt's algorithm based on the analysis presented in the article "Cost and delay- optimal parameter choices for accurate division using Goldschmidt's algorithm" by Even, Seidel and Ferguson and based on the division algorithm that they present.

The user will be able to set the parameters of different variables that determine the accuracy of the exact errors during the intermediate, the number of iterations and the error of the initial guess. Using this input we will compute the bounds of the relative intermediate errors and will supply the user a graphic tool to be able to see the changes of the relative error due to changes in a free parameter.

The software is based on the Maple software with VB Gui interface.
We choose Maple as the main platform because it supports multi precision calculations and our numbers are very small.

The Maple program is open to the user and he can change parameters inside and to change the errors calculations in order to fulfill special requirements.

Chapter 2: Theoretical background

The theoretical background is presented in the article.

We had to add 2 items in order to be able to write the software:

1) Translate the precision of the intermediate operations to the exact error of the iteration.

2) Compute the first 2 iterations of the algorithm (The algorithms in the article calculate the errors for)

In this section we will explain the theoretical background of these 2 items.

Translate the precision of the intermediate operations to the exact error of the iteration.

The exact error of the iteration is a function of the circute and the algorithm that we are using.

As we see it the exact error are a function of the following parameters:

- Multipleir precision – Mp.
- Number of bits in N register – Np.
- Number of bits in D register – Dp.
- Number of bits in F register – Fp.
- One's complement error (if used) – Op.
- Injection error in MUL_RZ calculation-Ip.
- Evalulate F error in carry save representation – CSp.
- Error due to Multiplier representation - MEp

Calculation precision while passing the multipleir once

The relation ship between the exact erros and the circute parameters are:

$$\hat{\text{nepsi}} = f(\text{Mp}, \text{Np}, \text{Ip}, \text{MEp})$$

$$\hat{\text{depsi}} = f(\text{Mp}, \text{Dp}, \text{MEp})$$

$$\hat{\text{fepsi}} = f(\text{CSp}, \text{Fp}, \text{Op})$$

The following assumption are also relevant for the calculation

- Ip is a function of Mp
- Op is a function of Fp.
- CSp is a function of Fp.
- MEp is 0 if the representation is not redundent binary and 1 if the representation is Carry save.

According to the following we calcuale the exact error as follows:

$$\hat{nepsi} = \max(Np, (1+MEp)*Mp + Ip) = \max(Np, (2+MEp)*Mp)$$

$$\hat{depsi} = \max(Dp, (1+MEp)Mp)$$

$$\hat{fepsi} = Op + Bp = Fp * one's\_complement\_is\_used + 2*Fp$$

Calculation precision while passing the multipleir twice

we will use the same semantics as in the section above.

The difference is that now the minimum parameter is relevant.

In this case I assume that Ip is very small and I will not use it in the calculation.

$$\hat{nepsi} = f(Mp, Np, MEp)$$

$$\hat{depsi} = f(Mp, Dp, MEp)$$

$$\hat{fepsi} = f(CSp, Fp, Op) = f(\hat{depsi})$$

$$\hat{nepsi} = Min(Np, (1+MEp)*Mp)$$

$$\hat{depsi} = Min(Dp, (1+MEp)Mp)$$

$$\hat{fepsi} = \hat{depsi} * (one's\_complement\_is\_used + 1)$$

Compute the first 2 iterations of the algorithm

In order to compute the first 2 iterations of the algorithm we are using all the relevant claims presented in the article.

Assumptions

Our assumptions are:

$$n_i, d_i, f_i \geq 0;$$
$$0 \leq n_i + d_i + f_i \leq 1/4;$$
$$|e_0| + 3d_0/2 + f_0 < 1/2$$

To compute the iteration we are using definition 3:

Definition 3

$$n_i = \frac{neps_i}{N'_{i-1} \cdot F'_{i-1}}; d_i = \frac{deps_i}{D'_{i-1} \cdot F'_{i-1}}; f_i = \frac{feps_i}{2 - D'_{i-1}};$$

For the first iteration we get:

$$\text{Recall that}: N'_{-1} = A; D'_{-1} = B; F'_{-1} = \frac{1-e_0}{B}; D'_0 \in [1-\delta_0, 1+\delta_0]$$

$$n_0 = \frac{neps_0}{N'_{-1} \cdot F'_{-1}} = \frac{neps_0}{A \cdot \frac{1-e_0}{B}} \le \frac{2 \cdot neps_0}{1-e_0} \le \frac{2 \cdot \hat{neps}_0}{1-e_0}$$

$$d_0 = \frac{deps_0}{D'_{-1} \cdot F'_{-1}} = \frac{deps_0}{B \cdot \frac{1-e_0}{B}} = \frac{deps_0}{1-e_0} \le \frac{\hat{deps}_0}{1-e_0}$$

$$f_0 = \frac{feps_0}{2 - D'_0} = \le \frac{\hat{feps}_0}{2-(1+\delta_0)} \le \frac{\hat{feps}_0}{1-\delta_0}$$

For the second iteration we get

Recall that: $\grave{D_0} \in [1-\delta_0, 1+\delta_0]$ where $\delta_0 = |e_0| + 3d_0/2$

Since $|e_0| + 3d_0/2 + f_0 < 1/2$ the bound $D_0 \in [1-\delta_0, 1+\delta_0] \subset (1/2, 1+|e_0|+3d_0/2)$
also implies that $\grave{F_0} \in (1-\delta_0-f_0, 1+\delta_0-f_0) \subset (1/2, 3/2)$

$$n_1 = \frac{neps_1}{\grave{N_0}\cdot\grave{F_0}} = \frac{neps_1}{(1-n_0)A\cdot\frac{1-e_0}{B}\cdot\grave{F_0}} \le \frac{2\cdot neps_1}{(1-n_0)\cdot(1-e_0)\cdot(1-\delta_0-f_0)} \le \frac{2\cdot\hat{neps}_1}{(1-n_0)\cdot(1-e_0)\cdot(1-\delta_0-f_0)}$$

$$d_1 = \frac{deps_1}{\grave{D_0}\cdot\grave{F_0}} \le \frac{deps_1}{(1-\delta_0)\cdot(1-\delta_0-f_0)} \le \frac{\hat{deps}_1}{(1-\delta_0)\cdot(1-\delta_0-f_0)}$$

No assumptions

$$f_1 = \frac{feps_1}{2-\grave{D_1}} = \frac{feps_1}{2-(1+d_1)\cdot\grave{D_0}\cdot\grave{F_0}} \le \frac{feps_1}{2-(1+d_1)\cdot(1+\delta_0)\cdot(1+\delta_0-f_0)} \le \frac{\hat{feps}_1}{2-(1+d_1)\cdot(1+\delta_0)\cdot(1+\delta_0-f_0)}$$

Di assumption
recall that $D_1 \in [1-\delta_1, 1+\delta_1]$ in the di assumption so we get

$$f_1 = \frac{feps_1}{2-\grave{D_1}} = \frac{feps_1}{2-(1+\delta_1)} \le \frac{\hat{feps}_1}{1-\delta_1}$$

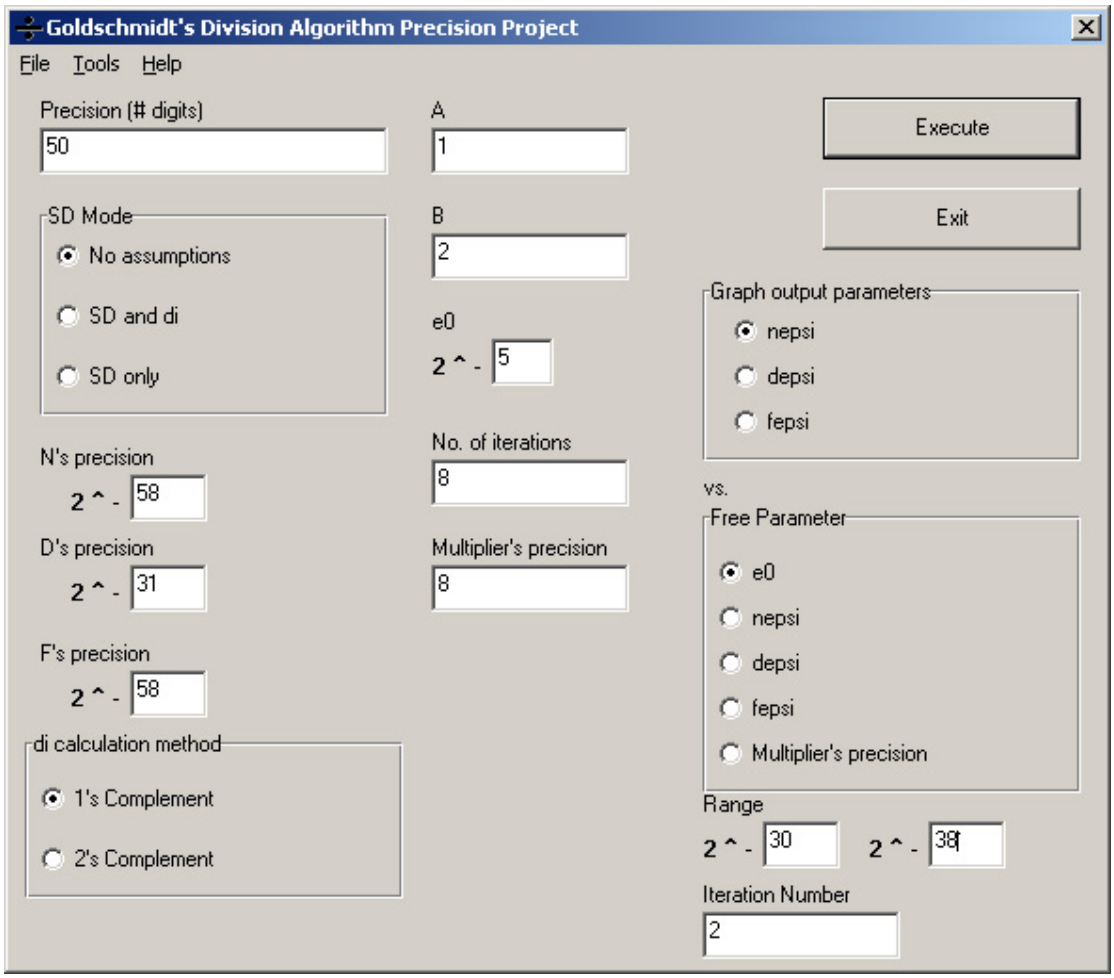
strict rounding assumption
recall that $D_1 \in [1-\delta_1, 1]$

$$f_1 = \frac{feps_1}{2-\grave{D_1}} \le \hat{feps}_1$$

Chapter 3: using the software

As we mentioned before the tool is build from 2 applications: a VB application to input the parameters and a Maple application to compute the errors and display graphs.

The Gui of the system is as follows:



The parameters that the user can control are:

Input parameters:

- Precision – The precision of the Maple Program.
- SD Mode – The desired rounding mode of the algorithm.
- N's precision – The precision of the N register.
- D's precision – The precision of the D register.

- F's precision – The precision of the F register.
- fi calculcation method – use one's or two's complement in the calculcation of 2-Di
- A,B – A and B inputs for the system.
- e0 – the initial guess error.
- No' of iterations – the number of iterations that the algorithm uses.
- Multiplier precision – The precision of the multipleir.

Output parameters

- Graph output parameter – select the desired error in the output graph.
- Free parameter – The free parameter that we want to alter in order to create the graph.
- Range – the range that the free parameter needs to alter – from Min to Max.
- Iteration number – the relevant iteration that we want to see in the graph.

Execute

Execute button will create an input file for the maple problem and will run the maple problem.

Since the maple program is not automated, the user need to click <enter> to run the different sections and in order to see the graph.

Note: if only the maple program is used the the input file needs to be at the directory as specified in the maple program (fopen function)

Input file structure

At this section we will descrive the structure of the input file :

For the following input file:

Parameter	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Example value	50	1.1	1.3	64	58	31	58	5	0	1	5	0	1	10	20	2

1. Precision – The precision of the Maple Program.

2. A input.
3. B input.
4. Multiplier precision – The precision of the multipleir.
5. N's precision – The precision of the N register.
6. F's precision – The precision of the F register.
7. D's precision – The precision of the D register.
8. e0 – the initial guess error.
9. SD Mode – The desired rounding mode of the algorithm.
10. fi calculcation method – use one's or two's complement in the calculcation of 2-Di
11. No' of iterations – the number of iterations that the algorithm uses.
12. Free parameter – The free parameter that we want to alter in order to create the graph.
13. Graph output parameter – select the desired error in the output graph.
14. Min Range – the range that the free parameter needs to alter – Min.
15. Max Range – the range that the free parameter needs to alter – Max.
16. Iteration number – the relevant iteration that we want to see in the graph.

The output of the system is as follows:

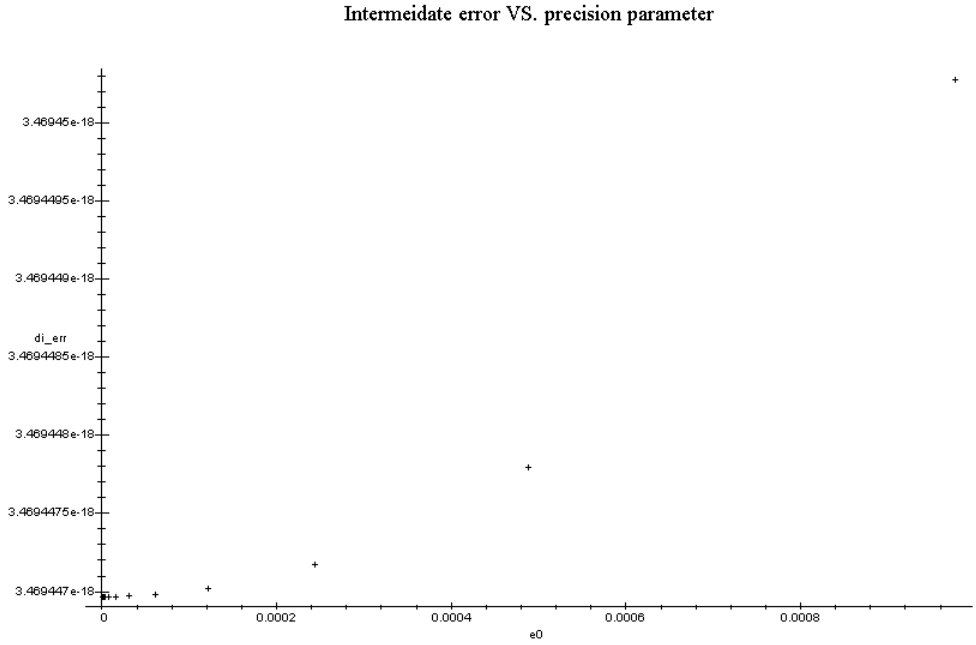
The output is a graph of the selected free parameter and the selected error. The user can also display the value of the error in all iteration and also display the value of all intermediate errors in the selected iteration.
The user can also add commands manually and look at all the errors for a specific iteration of for a specific free parameter value.

For example:
The input file is as follows:

Parameter	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Example value	50	1.1	1.3	64	58	31	58	5	0	1	5	0	1	10	20	2

in this case the free parameter is e0 and the error paramter is di. We will see the graph when e0 alter from 2^(-10) to 2^(-20) for iteration 2 of the algrithm.

The output graph in this case is:



The output also plots the values of the graph.
For example:

di_err[0,2]=.346945027523848767875316650138358498770584875543 74 10^(-17).
e0, 0 = 1/1024

di_err[1,2]=.346944779368418724877880389300210207902119871892 76 10^(-17).
e0, 1 = 1/2048

di_err[2,2]=.346944717329380023649416951017397503105994684716 35 10^(-17).
e0, 2 = 1/4096

di_err[3,2]=.346944701819505485998558298317107399614614383942 40 10^(-17).
e0, 3 = 1/8192

di_err[4,2]=.346944697941977903699551394512433917154948850433 74 10^(-17).
e0, 4 = 1/16384

di_err[5,2]=.346944696972566439384278702785438340001226371855 85 10^(-17).
e0, 5 = 1/32768

di_err[6,2]=.346944696730198783010020981945809622641052046267 13 10^(-17).
e0, 6 = 1/65536

di_err[7,2]=.346944696669599473398370151966138720990073296239 95 10^(-17).
e0, 7 = 1/131072

Open problems & questions

1) While calculating the algorithm without assumptions we need to calculate δ_i but according to the article we can calculate it only for i=1 ($\delta_{i+1} = \max(|e_0|, d_1), i = 0$) and we need it for i=0. at this time we made the following assumption: $\delta_0 = \max(|e_0|, d_0)$. We don't know if this assumption is correct.
2) When the numbers are very small the automatic scale of maple is incorrect the there is a need to change the scale manually. At this time we cannot overcome this problem. At this time we have the option to show the result as a bias from the first point in the graph. With this graph we can see the difference between the errors regarding the first point in the graph. We want to know your opinion on this matter.