3C TIC *(Edición 23)* *Vol.6 – Nº4*
Diciembre'17 – marzo'18, 19 - 26
Área de Innovación y Desarrollo, S.L.
ISSN: 2254 – 6529
DOI: http://dx.doi.org/10.17993/3ctic.2017.58.19-26

Recepción: 10 de mayo de 2017
Aceptación: 5 de octubre de 2017
Publicación: 29 de diciembre de 2017

ALGORITMO DE BOOTH EN ARIDAD CON MÚLTIPLES OPERANDOS

BOOTH ALGORITHM IN ARITY WITH MULTIPLE OPERANDS

Jesús Ayuso Pérez[1]

1. Compositor musical y desarrollador. Licenciado en Ingeniería Informática por la Universidad Carlos III de Madrid (UC3M). E-mail: ayusoperez@terra.com

Citación sugerida:

Ayuso Pérez, J. (2017). Algoritmo de Booth en aridad con múltiples operandos . *3C TIC: Cuadernos de desarrollo aplicados a las TIC,* 6(4), 19-26. DOI: <http://dx.doi.org/10.17993/3ctic.2017.58.19-26/>.

3C TIC *(Edición 23)* Vol.6 – Nº4
Diciembre'17 – marzo'18, 19 - 26
Área de Innovación y Desarrollo, S.L.
ISSN: 2254 – 6529
DOI: http://dx.doi.org/10.17993/3ctic.2017.58.19-26

RESUMEN

El algoritmo dado por Andrew Donald Booth en 1950 (Booth, 1951) para la multiplicación es potencialmente más rico y adaptable (Ayuso 2015, pp. 113-119) que sólo como fue descrito en principio, pudiendo ser aplicado a más de un único operando en una misma operación que tenga una aridad mayor que 1 (Ayuso 2015, pp. 255-221). De ahí que en el presente documento se mostrarán algunas implementaciones que, basándose en dicho concepto, hacen uso simultaneo de la simplificación de Booth en varios de sus operandos. Se verán las mejoras de rendimiento que puede ofrecer el utilizar, en un mismo cálculo, el algoritmo de Booth siendo éste aplicado a la vez a varios de los operandos que entran en juego.

ABSTRACT

The algorithm given by Andrew Donald Booth in 1950 (Booth, 1951) for multiplication is possibly richer and adaptable (Ayuso 2015, pp. 113-119) than just as it was in the beginning, and can be applied to more than one single operation in a single operation having a greater 1 (Ayuso 2015, pp. 255-221). We will show some implementations that, based on this concept, make simultaneous use of the simplification of Booth in several of its operands. You will see the performance improvements that can be offered by using, in a single calculation, the Booth algorithm being applied simultaneously to several of the operands that come into play.

PALABRAS CLAVE

Booth, Algoritmo, Adición, Aridad, Múltiple.

KEYWORDS

Booth, Algorithm, Addition, Multiple Arity.

3C TIC (Edición 23) Vol.6 – Nº4
Diciembre'17 – marzo'18, 19 - 26
Área de Innovación y Desarrollo, S.L.
ISSN: 2254 – 6529
DOI: http://dx.doi.org/10.17993/3ctic.2017.58.19-26

1. INTRODUCCIÓN

El concepto de Booth es perfectamente aplicable a más de un operando dentro de la misma operación (Ayuso 2015, pp. 255-221), permitiendo extender el rendimiento descrito por Booth (Booth, 1951) a varios de los elementos de los que conste la operación algebraica que se esté calculando. Por consiguiente, la reducción de los cálculos necesarios para realizar ese cómputo, que se conseguía usando dicho concepto, puede ampliar su alcance; es decir, reducir aún más el número de operaciones parciales para lograr el resultado final.

La aplicabilidad del algoritmo de Booth depende de la naturaleza matemática con la que se pueda definir la operación que deseamos calcular. Por ello, y de igual manera, la posibilidad de poder extender su uso a varios de los elementos que intervienen en los cálculos también estará supeditada a esa definición comportamental.

Partiendo de esto, a lo largo del presente documento, se expondrán algunas implementaciones que delaten esa capacidad de uso del algoritmo de Booth, además de mostrar que efectivamente implican un aumento considerable del rendimiento y la gran potencia que atesora esta técnica.

2. METODOLOGÍA

Para este apartado, nos decantaremos por una operación sencilla y bien conocida, a la que le aplicaremos la mejora que se trata de describir. Por ser una operación ligera y simple permitirá más fácilmente ver exactamente lo que se busca describir en este documento. Será la operación de adición expuesta en la referencia bibliográfica titulada: '*Booth algorithm hardware operations addition and subtraction*' (Ayuso 2017, pp. 1-9). La versión del algoritmo de suma de dos números, *a* y *b*, de longitud *n* que usamos es:

```
result = 0; // ELEMENTO NEUTRO
for(int i = 0; i < n; i++)  {
      if(a[i] == 1) result = successor(result, i);
      if(b[i] == 1) result = successor(result, i);
}
```

Figura 1. Algoritmo de suma 1.

De acuerdo, ahora se repasará la tabla dada por Booth para reducir el número de operaciones necesarias, apoyándonos en la propiedad de invertibilidad de la operación con la que se construye el cálculo para la estructura algebraica en la que nos encontramos:

3C TIC (Edición 23) Vol.6 – Nº4
Diciembre'17 – marzo'18, 19 - 26
Área de Innovación y Desarrollo, S.L.
ISSN: 2254 – 6529
DOI: http://dx.doi.org/10.17993/3ctic.2017.58.19-26

bit menos significativo	bit extra	Interpretación	Acción
0	**0**	intermedio cadena de 0s	ninguna
0	**1**	final cadena de 1s	operación
1	**0**	comienzo cadena de 1s	operación inversa / inverso misma operación
1	**1**	intermedio cadena de 1s	ninguna

Tabla 1. Acciones de Booth.

Ahora lo que se hace es aplicar esa tabla de reducción de operaciones a los dos operandos que componen la suma, de **aridad 2**, en lugar de a uno sólo de los operandos, como se había hecho siempre hasta ahora, y como hizo Booth cuando describió su método de multiplicación (Booth, 1951). De modo que, la suma de dos números, *a* y *b*, de longitud *n* sería:

```
result = 0;
bitExtra1 = 0;
bitExtra2 = 0;

for(int i = 0; i < n; i++)  {
     switch(actionBooth(a[i], bitExtra1) {
          case ( 0 1 ):
          result = successor(result, i);
          break;
     case ( 1 0 ):
          result = predecessor(result, i);
          break;
     }

     switch(actionBooth(b[i], bitExtra2) {
          case ( 0 1 ):
          result = successor(result, i);
          break;
     case ( 1 0 ):
          result = predecessor(result, i);
          break;
     }
```

3C TIC (Edición 23) Vol.6 – Nº4
Diciembre'17 – marzo'18, 19 - 26
Área de Innovación y Desarrollo, S.L.
ISSN: 2254 – 6529
DOI: http://dx.doi.org/10.17993/3ctic.2017.58.19-26

```
      bitExtra1 = a[i];
      bitExtra2 = b[i];
}
```

Figura 2. Algoritmo de suma 2.

Puede apreciarse que en realidad, la variación que se ha dado del algoritmo no dista mucho de la que figura en la bibliografía titulada: '*Booth algorithm operations addition and subtraction*' (Ayuso 2015, pp. 113-119). Lo único que se hace, como avanzábamos, es que en lugar de aplicar la reducción descrita por Booth a uno solo de los elementos al ser operado contra el otro, la aplicamos en los 2 elementos de los que consta la operación al ser operados contra el elemento neutro de la misma. Por lo demás, se trata del concepto descrito por Booth para la multiplicación en estado puro.

Seguimos haciendo hincapié en la potencia del concepto de Booth aprovechando el caso que acabamos de explicar, veámoslo ahora con una operación de otra aridad: con **aridad 3**; de hecho, usaremos la misma operación de suma, pero en un contexto modular (Ayuso 2015, pp. 222-229). Es decir, la suma de dos números, *a* y *b* módulo *m*, de longitud *n* quedaría:

```
result = 0;
bitExtra1 = 0;
bitExtra2 = 0;
bitExtra3 = 0;

for(int i = 0; i < n; i++)  {
      switch(actionBooth(a[i], bitExtra1) {
            case ( 0 1 ):
            result = successor(result, i);
            break;
      case ( 1 0 ):
            result = predecessor(result, i);
            break;
      }

      switch(actionBooth(b[i], bitExtra2) {
            case ( 0 1 ):
            result = successor(result, i);
```

3C TIC (Edición 23) Vol.6 – Nº4
Diciembre'17 – marzo'18, 19 - 26
Área de Innovación y Desarrollo, S.L.
ISSN: 2254 – 6529
DOI: http://dx.doi.org/10.17993/3ctic.2017.58.19-26

```
            break;
      case ( 1 0 ):
            result = predecessor(result, i);
            break;
      }
      switch(actionBooth(m[i], bitExtra3) {
            case ( 0 1 ):
            result = predecessor(result, i);
            break;
      case ( 1 0 ):
            result = successor(result, i);
            break;
      }

      bitExtra1 = a[i];
      bitExtra2 = b[i];
      bitExtra3 = m[i];
}
```

Figura 3. Algoritmo de resta 2.

Si nos fijamos, todas las variaciones anteriores lo único que aportan es explotar el concepto ideado por Booth, pero lo aplican a todos los operandos que intervienen en la operación, lo cual dependerá de la aridad de la misma. Parece sano comentar que, en el caso particular que figura sobre estas líneas, para los operandos *a* y *b*, se realiza la misma operación en función de la tabla de acciones de Booth, pero en el caso del módulo *m*, se realizan las operaciones a la inversa, ya que ese operando reduce, mientras que los otros dos adicionan. También indicar que la implementación anterior da por sentado que el resultado de *a* sumado *b* siempre desborda al módulo *m*. Pero lo realmente importante, y es lo que aporta el presente documento, es que, lo que no cambia, en el caso de los 3 números, *a*, *b* y *m*, es que reducimos sus secuencias consecutivas de 1s existentes en su representación binaria, para realizar menos cálculos. Tal y como Booth nos enseñó (Booth, 1951), pero haciéndolo de forma simultánea, dentro de la misma operación, para el caso de los 3 operandos que intervienen en el cómputo. Y eso es lo novedoso, o lo que aporta la explicación del presente documento.

3C TIC *(Edición 23)* Vol.6 – Nº4
Diciembre'17 – marzo'18, 19 - 26
Área de Innovación y Desarrollo, S.L.
ISSN: 2254 – 6529
DOI: http://dx.doi.org/10.17993/3ctic.2017.58.19-26

3. CONCLUSIONES

Aplicar el algoritmo de Booth a más de un elemento de la operación que se desea calcular, nos permite explotar la idea de reducir el número de 1s existente en la representación binaria en varios o en todos los elementos que intervienen en dicha operación: dependerá de su aridad y de su composición matemática. Esa disminución de los 1s, como sabemos (Booth, 1951), se traduce directamente en reducir el número de operaciones totales que se han de realizar para conseguir el resultado final. En definitiva, si el algoritmo de Booth basaba su eficiencia en la reducción de operaciones por eliminar los 1s consecutivos de un único operando, si hacemos eso mismo sobre todos los operandos que intervienen en los cálculos, es de esperar que la reducción de operaciones totales sea aún menor.

En conclusión, la idea propuesta por Booth, simplifica los cálculos de aquellas operaciones en las que puede ser utilizada de una forma aún más profunda, pudiendo llegar a ser aplicada sobre todos los operandos que intervienen en nuestro cómputo, consiguiéndonos un rendimiento considerablemente mejor que si no hacemos uso de ella.

Como posible futura línea de investigación, se podría valorar el aplicar simultáneamente con la propuesta actual las mejoras existentes para optimizar la idea tenida por Booth, como por ejemplo la representación que soluciona su peor caso, con los 1s individuales (Ayuso 2016, pp. 33-43). O la versión que da la representación NAF (Reitwiesner 1960, pp. 231-308), mucho más óptima aunque de cálculo más pesado y menos acoplable en los algoritmos clásicos.

3C TIC *(Edición 23)* Vol.6 – Nº4
Diciembre'17 – marzo'18, 19 - 26
Área de Innovación y Desarrollo, S.L.
ISSN: 2254 – 6529
DOI: http://dx.doi.org/10.17993/3ctic.2017.58.19-26

6. REFERENCIAS BIBLIOGRÁFICAS

Booth, A.D. (1945). A method of calculating reciprocal spacings for X-ray reflections from a monoclinic crystal. *J. Sci. Instr,* 22, p. 74.

Burks, A., Goldstein, H. and Von Neumann, J. (1946). Logical Design of an Electronic Computing Instrument.

Booth, A.D. and Britten, K. H. V. (1947). General Considerations in the Design of an Electronic Computer.

Booth, A.D. (1951). A signed binary multiplication technique. *Q. J. Mech. and Appl. Math,* 4(2), pp.236-240.

W. Reitwiesner, G. (1960). Binary Arithmetic, 231-308.

Ayuso, J. (2015). Booth algorithm operations addition and subtraction. *3C TIC,* 4(2), pp. 113-119.

Ayuso, J. (2015). Booth algorithm modular arithmetic operations of addition and subtraction. *3C TIC,* 4(3), pp. 222-229.

Ayuso, J. (2015). Booth algorithm modular arithmetic operations of multiplication. *3C TIC,* 4(4), pp. 255-221.

Ayuso, J. (2016). Booth algorithm in signed-digit representation. *3C TIC,* 5(3), pp. 33-43.

Ayuso, J. (2017). Booth algorithm hardware operations addition and subtraction. *3C TIC*, 6(3), pp. 1-9.