RESOLUCIÓN JUSTA DE PROBLEMAS: EL PASTEL, LA PIZZA Y EL CAMPEONATO DE SURF

Giovanni Sanabria Brenes
Instituto Tecnológico de Costa Rica –
Universidad de Costa Rica, Costa Rica
gsanabria@itcr.ac.cr

Resumen: La resolución justa de diversas situaciones ha intrigado a la humanidad y a la vez ha permitido a la matemática y la computación mostrar su potencial para brindar algoritmos que permitan resolverlos. En el presente trabajo se analizan tres problemas: repartición de un pastel, formar los equipos para cada eliminatoria en un torneo de surf, elegir quién va por la pizza con una moneda. Para el problema del pastel, resuelto en el 2017 con un algoritmo muy complejo, se expone una solución si el pastel se reparte a tres personas. Para cada uno de los otros dos problemas, se expone un intento fallido y una solución dada por el autor. Estos problemas pueden ser interesantes para motivar a nuestros estudiantes a que los modelen matemáticamente y traten de resolverlos.

Palabras clave: resolución de problemas, justicia, distribución uniforme, repartición.

Abstract: The fair resolution of various situations has intrigued humanity and at the same time has allowed mathematics and computing to show their potential to provide algorithms that allow them to be solved. In this paper, three problems are analyzed: distribution of a cake, forming the teams for each tie in a surf tournament, choosing who goes for pizza with a coin. For the cake problem, solved in 2017 with a very complex algorithm, a solution is exposed if the cake is distributed to three people. For each of the other two problems, a failed attempt and a solution given by the author are presented. These problems can be interesting to motivate our students to model them mathematically and try to solve them.

Keywords: problem solving, justice, uniform distribution, distribution.

1. Introducción

La resolución justa de una determinada situación ha sido un problema que ha intrigado a la humanidad desde tiempos antiguos. Por ejemplo, el reparto justo entre dos personas se solucionó desde tiempos antiguos bajo el lema “yo corto, tú escoges” (Klarreich, 2017), y consistía en que una de las personas reparte en dos partes el objeto y la otra escoge qué parte elegir. Aquí la justicia se daba en el sentido de que debía ser la persona que no repartía la escogiera con qué parte quedarse, pues de lo contrario, se cae en la injusticia: “El qué parte y reparte se queda con la mejor parte”.

A lo largo de la historia se han desarrollado diferentes algoritmos para resolver diferentes situaciones problemáticas de la manera más justa, como por ejemplo, el reparto de una herencia.

En el caso del reparto justo y equitativo de un objeto, en las últimas décadas, este problema ha sido abordado metafóricamente como "el problema de repartir un pastel entre n personas". Este problema fue resuelto para n=3, Bosch (1992) indica:

> *Este problema interesó a los matemáticos a principios de este siglo. En 1948, el matemático polaco Hugo Steinhaus escribió en uno de sus apuntes: "Al haber encontrado una solución para el problema del reparto del pastel entre tres personas, les propuse la generalización a mis compañeros B. Knaster y S. Banach." Los tres fueron matemáticos de excelente reputación internacional.*

En el problema de repartición del pastel, el concepto de justicia está subordinado a que cada persona considere que le tocó un pedazo justo.

Además del problema del pastel (problema 1), otros tipos de problemas que buscan una resolución justa son:

- Problema 2: ¿Cómo distribuir, de manera lo más justa posible, los jugadores de un campeonato en equipos, que en la medida de lo posible, tenga la misma cantidad de personas?
- Problema 3: ¿Cómo elegir de manera justa la persona que debe ir a comprar la pizza?

El problema 2, es un problema determinista y no hay justicia absoluta cuando el número de jugadores no es divisible entre el número de equipos a formar. El tercer problema, es aleatorio, y para que la elección sea justa, las personas deben tener igual probabilidad de ser elegidas para ir por la pizza.

Seguidamente se presentan un caso particular para cada uno de estos tipos de problemas y su respectiva solución.

2. ¿Cómo repartir un pastel entre n personas?

Una de las maneras más conocida para repartir el pastel entre 3 personas es dada en Bosch (1992), donde llama a las tres personas Sofía, Pablo y Claudia:

1. Sofía corta el pastel en dos partes que piensa son mitades.
2. Pablo elije y Sofía se queda con la otra mitad.
3. Pablo y Sofía dividen sus pedazos respectivos en tres partes que consideran iguales.
4. Claudia elije una tercera parte de cada uno.
5. Pablo y Sofía se quedan con lo restante.

El problema para n personas fue solucionado en el 2017. Al respecto Klarreich (2017) indica:

> *En abril del año pasado, sin embargo, dos teóricos de la computación superaron todas las expectativas cuando publicaron un artículo en línea en el que describían un algoritmo de reparto justo cuyo tiempo de cómputo solo dependía del número de jugadores, no de sus preferencias individuales. Uno de ellos, Simon Mackenzie, por entonces investigador posdoctoral de 27 años en Carnegie Mellon, presentó el resultado el pasado 10 de octubre en el Simposio sobre Fundamentos de la Teoría de la Computación del Instituto de Ingeniería Eléctrica y Electrónica (IEEE), uno de los congresos anuales más prestigiosos en ciencias de la computación. El algoritmo resulta extraordinariamente complejo: dividir un pastel entre n comensales puede requerir hasta n^n^n^n^n^n pasos (donde el símbolo ^ se usa aquí para expresar «elevado a») y un número equiparable de tajadas. Incluso para unos pocos jugadores, dicha cifra supera la cantidad de átomos existentes en el universo.*

Se infiere que no es una solución práctica pues para pocos jugadores parece que habrá que partir el pastel en boronas. Sin duda el reto ahora es hallar una mejor solución que sea más factible en la práctica.

3. ¿Cómo definir los equipos en un torneo de surf?

En el 2017 la Asociación Costarricense de Surf (ACOS) me planteó la siguiente problemática que tenían: la asociación organiza torneos de Surf e incluso minutos antes del inicio de un torneo se aceptan inscripciones por lo que no se sabe con antelación cuántos participantes tendrá el torneo. En cada eliminatoria se realizan grupos de 4 participantes o eventualmente de 3 participantes (se intenta que sean de 4). Cada grupo sale a surfear y es evaluado por tres jueces. El juez evalúa diferentes olas de cada participante. Para cada participante, se toman las puntaciones de las dos mejores olas evaluadas por cada juez, estás seis puntuaciones se promedian. De cada grupo clasifican, a la siguiente eliminatoria, los dos participantes con mejores promedios. Al final en la última eliminatoria debe haber 2 ganadores. El problema es cómo desarrollar un programa computacional que determine cuántos grupos se debe hacer y en general realice el proceso de clasificación en cada ronda.

El problema se intentó resolver bajo las condiciones dadas. Es decir, no se consideró la opción de hacer la competición de otra forma para generar más equidad en el torneo a la hora de elegir los ganadores. Esto, pues la forma de realizarlo responde quizás a una tradición o a un reglamento o a la naturaleza del deporte.

El gran problema era: Si se inscriben n participantes, ¿cuántos grupos se deben hacer (de 3 o 4 participantes) en cada eliminatoria para que al final “todo cierre bien” y se obtengan los 2 ganadores?

Note que no hay equidad perfecta si el número de competidores en cada eliminatoria no es múltiplo de cuatro. Esto pues al haber grupos de tres, es más fácil clasificar en un grupo de tres que en uno de cuatro, pues de acuerdo a las reglas, de cada grupo clasifican dos competidores. Además, como regla se deben evitar los grupos de tres, esto quizás por un asunto de no alargar el torneo.

3.1. Primer acercamiento al problema

Bajo condiciones ideales, si vamos del final al inicio del torneo, para obtener dos ganadores, estos se obtuvieron de un grupo de 4, es decir la última eliminatoria está formada por un grupo de 4. Entonces en la penúltima hay dos grupos de 4 y en la antepenúltima hay 2^2 grupos de 4:

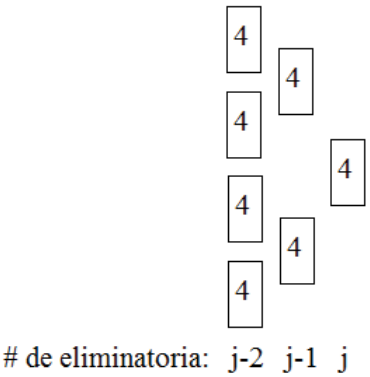

Figura 1. Intento de solución al problema de Surf

Bajo condiciones ideales, si en total hay j eliminatorias: en la eliminatoria j hay 4 competidores, en la eliminatoria j-1 hay $2^1 \cdot 4$ competidores, en la eliminatoria j-2 hay $2^2 \cdot 4$ competidores,…, en la primera eliminatoria hay $2^{j-1} \cdot 4$ competidores. Por lo tanto, si el número n de personas inscritas es de la forma $2^{j-1} \cdot 4$ el problema tiene una solución satisfactoria: en la primera eliminatoria se realizan 2^{j-1} grupos de 4, en la segunda se realizan 2^{j-2} grupos de 4,… y en la última eliminatoria un grupo de 4.

¿Qué sucede si n no es de esa forma? Por el principio del Buen Orden podemos hallar el natural j que cumpla que

$$2^{j-2} \cdot 4 < n < 2^{j-1} \cdot 4$$

Así, en la primera eliminatoria, no se pueden realizar 2^{j-2} grupos de 4 pues quedaría por fuera competidores, y si se realizan 2^{j-1} grupos de 4, faltarían competidores. En definitiva, se necesitan realizar grupos de tres. Dado que $2^{j-2} \cdot 4 = 2^{j-1} \cdot 2$, entonces $2^{j-1} \cdot 4$ es el doble de $2^{j-2} \cdot 4$, por lo tanto, n varía en un intervalo muy amplio.

Si insistimos en nuestro modelo de solución de hacer en la primera eliminatoria 2^{j-1} grupos, en la segunda se realizan 2^{j-2} grupos,… y la última eliminatoria un grupo, donde algunos grupos pueden ser 3, esto va a fracasar. En efecto, como

$$2^{j-1} \cdot 2 < n < 2^{j-1} \cdot 4$$

Suponga que se realizan 2^{j-1} grupos, algunos de 4 y otros de 3 competidores. Si nos vamos al caso extremo en que todos los grupos son de 3, entonces se tendrían $2^{j-1} \cdot 3$ competidores. El problema es que entre $2^{j-1} \cdot 2$ y $2^{j-1} \cdot 4$, justo en el centro se encuentra $2^{j-1} \cdot 3$ y si el n cumple que $2^{j-1} \cdot 2 < n < 2^{j-1} \cdot 3$, entonces haciendo solo grupos de 3 no es posible realizar 2^{j-1} grupos, algunos tendrán que ser de dos, lo cual es absurdo. Por lo tanto, este primer acercamiento no resulta útil.

3.2. Una solución al problema

En la solución anterior, más que buscar un número par de grupos para cada eliminatoria, si intentó que el número de grupos por eliminatoria fuera siempre una potencia de dos. En realidad, el número de grupos por eliminatoria no tiene que ser par.

Sean n_i el número de competidores en la eliminatoria i. Si dividimos este número entre cuatro, por el Algoritmo de la División existen c_i y r_i tales que:

$$n_i = 4c_i + r_i, \text{ con } 0 \leq r_i < 4$$

Si $c_i > 1$, el número de grupos será:

Valor del residuo	Grupos de 4	Grupos de 3	Total de grupos
$r_i = 0$	c_i	0	c_i
$r_i = 1$	$c_i - 2$	3	$c_i + 1$
$r_i = 2$	$c_i - 1$	2	$c_i + 1$
$r_i = 3$	c_i	1	$c_i + 1$

Tabla 1. Solución al problema de surf

Note que c_i no puede ser cero, salvo que se inscriban solo 2 o 3 competidores, de lo contrario esto no puede ocurrir en la última eliminatoria pues:

- Si solo quedan 2 competidores, estos serían los ganadores y el torneo terminó en la eliminatoria anterior.
- No puede haber tres competidores en la última eliminatoria, pues clasifica un número par de participantes.

Además, si $c_i = 1$ entonces la tabla anterior funciona bien excepto cuando $r_i = 1$, en este caso $n_i = 5$. Esto no puede ocurrir en ninguna eliminatoria excepto en la primera, pues en cada eliminatoria, después de la primera, hay un número par de competidores.
Así, los casos no cubiertos por el algoritmo presentado son cuando inicialmente se inscriben solamente 2,3 o 5 competidores.

¿Siempre se obtendrá dos ganadores en la última eliminatoria? En efecto, observe que:

Valor del residuo	Total de grupos	# de competidores en eliminatoria i	# de competidores en eliminatoria i
$r_i = 0$	c_i	$n_i = 4c_i$	$n_{i+1} = 2c_i$
$r_i = 1$	$c_i + 1$	$n_i = 4c_i + 1$	$n_{i+1} = 2c_i + 2$
$r_i = 2$	$c_i + 1$	$n_i = 4c_i + 2$	$n_{i+1} = 2c_i + 2$
$r_i = 3$	$c_i + 1$	$n_i = 4c_i + 3$	$n_{i+1} = 2c_i + 2$

Tabla 2. Decrecimiento par del número de competidores por eliminatoria

Como c_i no es nulo, se puede asegurar $n_{i+1} < n_i$. Es decir, la cantidad de competidores va decreciendo estrictamente. Dado que, en cada eliminatoria después de la primera, hay un número par de competidores y esté va decreciendo estrictamente, necesariamente existirá un k tal que $n_k = 2$. Es decir, el proceso tiene fin (hay un número finito de pares entre 1 y n_1) y la última eliminatoria será la k-1.

El algoritmo se programó en Excel y se ha venido utilizando en los torneos de ACOS. En la primera hoja se ingresan los datos:

1		**Total de competidores:**		**22**		# de eliminatoria:		**1**	
2	**Nombre de competidores**					# de grupos		Total de	
3	#	**Nombre**		Eliminatoria	# competidores	De 4 personas	De 3 personas	grupos	
4	1	luis		1	22	4	2	6	
5	2	ana		2	12	3	0	3	
6	3	juan		3	6	0	2	2	
7	4	beto		4	4	1	0	1	
8	5	jose		5	2				
9	6	Luisa							

Figura 2. Programa de cantidad de grupos por eliminatoria diseñado en Excel

En la siguiente hoja aparecen los diferentes grupos con los espacios para que los jueces ingresen las calificaciones:

Grupo 1	Nombre	color		ola1	ola2	ola3	ola4	ola5	ola6	ola7	ola8	ola9	ola10	ola11	ola12	Las 2 olas mejores	promedio
	luis		juez1	1	2	1										2	1
			juez2	1	1	4										4	1
			juez3													0	0
			Promed:														1,333333
	ana		juez1													0	0
			juez2													0	0
			juez3													0	0
			Promed:	0	0	0	0	0	0	0	0	0	0	0	0		0
	juan		juez1													0	0
			juez2													0	0
			juez3													0	0
			Promed:	0	0	0	0	0	0	0	0	0	0	0	0		0
	beto		juez1													0	0
			juez2													0	0
			juez3													0	0
			Promed:	0	0	0	0	0	0	0	0	0	0	0	0		0

Posición	Competidor	Puntaje	Diferencia para subir una posición
1	luis	1,333333	
2	ana	0	1,333333333
3	juan	0	0
4	beto	0	0

Figura 3. Programa para calificación de jueces diseñado en Excel

La tercera hoja indica los clasificados a la siguiente eliminatoria y los puntajes obtenidos:

Clasificados de la eliminatoria

#	Nombre	Pts
1	ana	2,166666667
2	luis	1,333333333
3	jose	1,333333333
4	Luisa	0

Figura 4. Lista de clasificados diseñada en Excel

Actualmente ACOS está en proceso de contratar un profesional en informática que programe el algoritmo como una aplicación independiente que permita el ingreso de datos de forma remota, esto para que cada juez ingrese los datos desde su computadora personal.

4. ¿Quién va por la pizza?

El siguiente problema lo plantearon unos estudiantes de Ingeniería Computación del Tecnológico de Costa Rica en el primer semestre del 2019 al autor de este trabajo. Al parecer una empresa, para definir a quiénes va a contratar, les dio una lista de problemas a los oferentes y éstos tenían un plazo para entregarlos.

El problema dice (se modificó levemente): En una reunión hay n personas, numeradas del 1 al n, y desean comer pizza. Defina un algoritmo que permita elegir de manera justa a la persona que debe ir por la pizza, por medio de una moneda.

No se puede dar un algoritmo determinista para resolver el problema, pues al introducir la utilización de la moneda, nos introduce en un mundo azaroso. El concepto de justicia se operacionaliza en que todas las personas tengan la misma posibilidad de ser elegidas. En realidad, el problema matemáticamente se reduce a hallar una distribución uniforme para la variable X (número de persona que debe ir por la pizza) de rango {1,2,3,…,n}, utilizando una moneda.

4.1. Primer acercamiento al problema

Una solución común a este problema sería lanzar la moneda n-1 veces y sea Y la variable aleatoria que indica el número de escudos obtenidos, entonces va por la pizza la persona número X=Y+1. Por ejemplo, si son cinco personas, lanzamos la moneda 4 veces, si se obtiene 3 escudos, entonces la persona número 4 es la elegida.

El gran problema de esta solución es que la distribución de X no es uniforme. Note que si Y es el número de escudos en n –1 lanzamientos, entonces:

$$Y \sim B(n-1,½)$$

Así, la función de distribución de X es $f_X(k) = f_Y(k-1)$, la cual no es uniforme. Por ejemplo, si n=5 el histograma de X es:

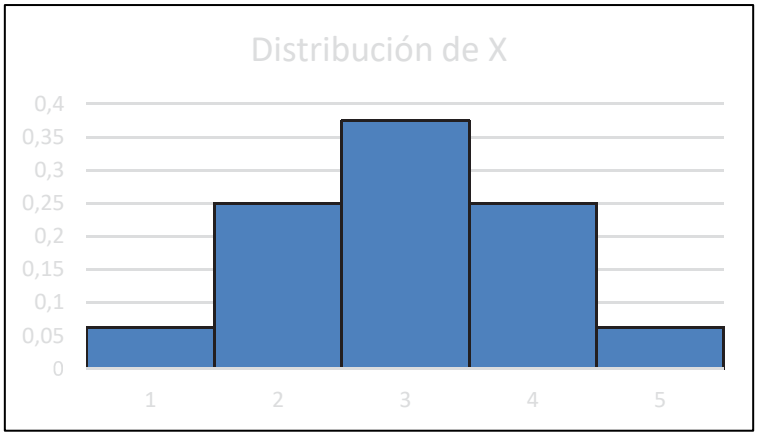

Figura 5. Histograma de X: # de persona que va por la pizza para n=5

Así, en el ejemplo con n=5, la elección de la persona que debe ir por la pizza no es justa, pues la persona 3 tendrá más posibilidad de ser elegida que las otras. Por lo tanto, se descarta esta forma de solucionar el problema.

4.2. Una solución al problema

El formar una distribución uniforme para una variable de rango {1,2,3,…,n} con una moneda no es para nada sencillo. De hecho, se cuestiona si realmente existirá dicho método.

Sin embargo, seguidamente se plantea una solución justa al problema pero un poco más compleja.

La solución surge al analizar la distribución de la variable aleatoria Y: # de escudos al lanzar la moneda m veces. Note que Y sigue una distribución binomial donde la probabilidad de éxito es ½ por lo tanto

$$F_Y(k) = \frac{C(m,k)}{2^m}, k = 0,1,2,\dots,m.$$

Donde el coeficiente binomial $C(n,k)$, en realidad agrupa resultados equiprobables. Simbolicemos los resultados al lanzar una moneda con E (escudo) y C (Corona). Por ejemplo para m=3, observe que:

K	Resultados favorables a Y=k	$F_Y(k)$
0	CCC	$\frac{1}{8}$
1	CCE, CEC, ECC	$3 \cdot \frac{1}{8}$
2	CEE, ECE, EEC	$3 \cdot \frac{1}{8}$
3	EEE	$\frac{1}{8}$

Tabla 3. Distribución de Y: # de escudos al lanzar la moneda m=3 veces

Así encontramos resultados equiprobables, para m=3 los posibles resultados al lanzar una moneda son 8 (CCC, …, EEE) y cada uno tiene una probabilidad de 1/8. Estos resultados nos ayudarán a resolver el problema.

Note que, si se lanza una moneda una vez, los posibles resultados son 2; si se lanza dos veces, los posibles resultados son 2^2; …; si se lanza m veces, los posibles resultados son 2^m. Así el problema tiene una solución perfecta si el número de personas reunidas es de la forma 2^m. En efecto, si hay 2^m personas, vamos a lanzar la moneda m veces, anotamos su resultado con C y E, luego cada E del resultado la cambiamos por 1 y cada C por cero. El resultado obtenido con los cambios hechos es un número en binario, que al pasarlo a base 10 y sumarle 1, da el número de la persona que debe ir por la pizza.

Por ejemplo, si hay 2^3 personas reunidas lanzamos la moneda 3 veces, hay 8 posibles resultados, cada uno está asociado aún número de persona:

CCC -> 000 -> 0+1=1

CCE -> 001 -> 1+1 =2

CEC -> 010 -> 2+1 =3

ECC -> 100 -> 4+1 =4

CEE -> 011 -> 3+1 =5

ECE -> 101 -> 5+1 =6

EEC -> 110 -> 6+1 =7

EEE -> 111 -> 7+1 =8

Así, si se lanza la moneda 3 veces y se obtiene CEE entonces la persona número 5 debe ir por la pizza.

¿Qué sucede si n no es de la forma 2^m? Por el principio del Buen Orden podemos hallar el natural m que cumpla que

$$2^{m-1} < n < 2^m$$

Por lo tanto, se lanzará la moneda m veces y se procederá de la misma manera que anteriormente: anotamos su resultado con C y E, luego cada E del resultado lo cambiamos por 1 y cada C por cero, obteniendo un número en binario, que al pasarlo a base 10 y sumarle 1, da un número N.

Como $n < 2^m$, entonces N no necesariamente está entre 1 y n. La regla será la siguiente: si $N \leq n$, la persona número N debe ir por la pizza, sino se debe volver a lanzar la moneda N veces y proceder de la misma manera hasta elegir quién va por la pizza.

¿Funciona este algoritmo? Primero note que cada persona tiene la misma probabilidad de ser elegida, esta es $1/2^m$. Por otro lado, note que la probabilidad de que ninguna persona sea elegida es

$$\frac{2^m - n}{2^m}$$

Como $n > 2^{m-1}$ entonces $-n < -2^{m-1}$ y la probabilidad de que ninguna persona sea elegida cumple:

$$\frac{2^m - n}{2^m} < \frac{2^m - 2^{m-1}}{2^m} = \frac{2^{m-1}}{2^m} = \frac{1}{2} = 50\%$$

Quiere decir que es más probable elegir a alguien que no elegirlo, al lanzar la moneda m veces. Por la ley de los grandes números, si la moneda es legal, el proceso deberá terminar. Es decir, existirá un grupo de m lanzamientos de la moneda en el cual se logre elegir la persona que irá por la pizza. Y dado que la probabilidad es mayor al 50%, es probable que para este grupo de m lanzamientos, suceda no se tenga que esperar mucho. Por ejemplo, si son n=5 personas, note que $2^2 < 5 < 2^3$, por lo tanto, m=3 y se lanza la moneda 3 veces. De los 8 posibles resultados, solo hay 3 que no asignan a la persona que va por la pizza, estos son: ECE, EEC, EEE. Si uno de estos resultados sucede, si seguirá

lanzando la moneda tres veces, hasta que uno de estos resultados no se dé y se pueda elegir a la persona. Note que en cada grupo de tres lanzamientos de la moneda, la probabilidad de que no se asigne a la persona es de 3/8.

El algoritmo lo podemos simular en algún software. Por ejemplo en Matemática

```
Pizza[n_] :=  Module[{m = 0, respuesta = n + 1, Lanzar, k = 0},
  (*Hallar el m *)
  While[2^m < n,  m = m + 1;   ];
  Print["El número de lanzamientos de la moneda es ", m];
  (*elegir a la persona *)
  While[respuesta > n - 1,
        Lanzar = RandomInteger[1, m];
        k = 0;
        respuesta = 0;
        While[k <= m - 1,
              respuesta = respuesta + (Lanzar[[k + 1]]*2^k);
              k = k + 1;
             ];
        Print["Se elige a la persona # ", respuesta + 1, " para ir por la pizza"];
         If[respuesta + 1 > n,
        Print["Como solo hay ", n, " personas, se vuelve a lanzar la moneda ", m," veces"];
            ];
          ];
  respuesta = respuesta + 1;
  respuesta
  ];
```

Para 50 personas el resultado fue el siguiente:

```
In[7]:= Pizza[50]
        El número de lanzamientos de la moneda es 6
        Se elige a la persona ¤ 55 para ir por la pizza
        Como solo hay 50 personas, se vuelve a lanzar la moneda 6 veces
        Se elige a la persona ¤ 63 para ir por la pizza
        Como solo hay 50 personas, se vuelve a lanzar la moneda 6 veces
        Se elige a la persona ¤ 16 para ir por la pizza
Out[7]= 16
```

Figura 6. Ejemplo del programa en Mathematica para el problema de la pizza

5. Conclusión

Sin duda la resolución justa de problemas ha intrigado a la humanidad. Desde el punto de vista matemático, son problemas muy interesantes que pueden despertar el interés de los estudiantes, de diferentes niveles. Aquí se presentaron tres tipos de estos problemas: repartición de un pastel, formar los equipos para cada eliminatoria en un torneo de surf, elegir quién va por la pizza con una moneda. Puede resultar interesante plantear estos problemas a estudiantes de primaria, secundaria y a nivel universitario y analizar como ellos utilizan las herramientas matemáticas que tienen para modelar los problemas y dar una solución parcial a estos.

Para estos problemas, se brindó una posible solución, pero no una solución óptima. De hecho, son problemas abiertos, donde el lector puede buscar mejores soluciones a las planteadas en este documento.

En el caso de los dos últimos problemas, se brindaron soluciones fallidas, pero muy ricas matemáticamente y pedagógicamente. En el torneo de surf, se muestra contundentemente que se debe renunciar a seguir el primer camino de solución. En el problema de la pizza, además de que se muestra que la primera aproximación es errónea, esta genera la solución brindada.

Por otro lado, es importante notar cómo se combina la matemática y el uso de software para resolver este tipo de problemas.

6. Referencias bibliográficas

Klarreich, E. (2017, abril). Un algoritmo para repartir una tarta. Revista Investigación y Ciencia. Volumen (487). España.

BOSCH, C. (1992) El que parte y reparte se queda con la mejor parte. Ciencias. Volumen (025). México.