

RANA SALTARINA

Daniel Rivas Torres
danimlg@gmail.com

Alfredo Prieto Ruiz
artetelo@hotmail.com

Antonio Fernández Chamorro
maschamorro@hotmail.com

1. Funcionamiento del juego:

El juego se puede generalizar para n ranas. Para mostrar el funcionamiento vamos a jugar con 4 ranas, por lo tanto $n=4$. Dos ranas verdes y otras dos marrones.

Para comenzar necesitamos una charca con 5 piedras situadas una al lado de la otra, en la primera y segunda piedra se colocan las dos ranas verdes y en la cuarta y quinta piedra las ranas marrones, dejando la tercera piedra vacía, tal y como muestra en la viñeta 1. La finalidad del juego es pasar las ranas verdes al lado derecho y las marrones al izquierdo como muestra la viñeta 9.



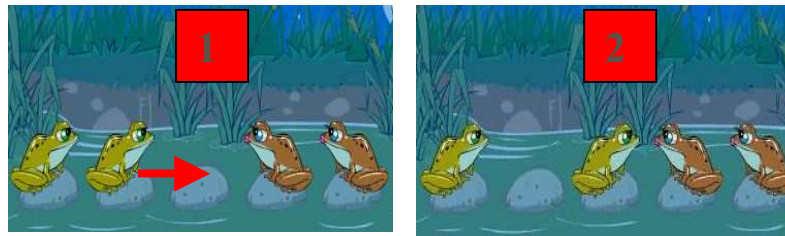
INICIO



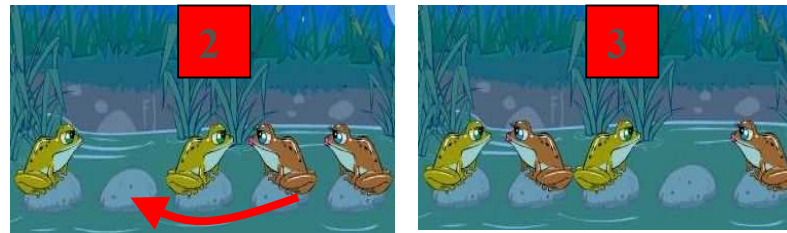
FINAL

¿Cómo podemos mover una rana y que movimientos están permitidos?

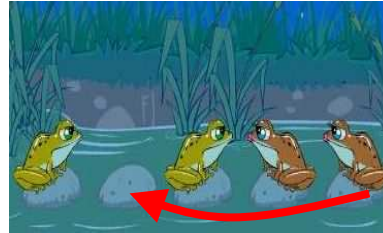
Las ranas pueden saltar a la piedra contigua siempre que dicha piedra esté vacía (ejemplo viñeta 1 y 2)



Pueden saltar sobre otra rana siempre y cuando caiga sobre una piedra vacía (ej. viñeta 2 y 3)



No está permitido saltar 2 ranas o más



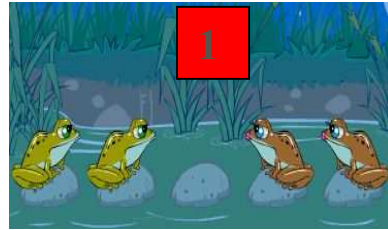
Empezamos moviendo la rana verde



Una rana puede volver a su piedra si ve que se ha equivocado



2. Solución gráfica para $n=4$:



Viñeta 1: Ranas en posición inicial posiciones de la 1 a la 5

Viñeta 2: Movemos rana verde posición 2 a posición 3 $\rightarrow S2 : d_{(2,3)}$

Viñeta 3: Movemos rana marrón posición 4 a posición 2 $\rightarrow S13 : s_{(4,2)}$



Viñeta 4: Movemos rana marrón posición 5 a posición 4 $\rightarrow S8 : d_{(5,4)}$

Viñeta 5: Movemos rana verde posición 3 a posición 5 $\rightarrow S11 : s_{(3,5)}$

Viñeta 6: Movemos rana verde posición 1 a posición 3 $\rightarrow S9 : s_{(1,3)}$



Viñeta 7: Movemos rana marrón posición 2 a posición 1 → S5 : d_(2,1)

Viñeta 8: Movemos rana marrón posición 4 a posición 2 → S13 : s_(4,2)

Viñeta 9: Movemos rana verde posición 3 a posición 4 → S3 : d_(3,4)

Juego terminado.

3. Modelado del juego:

Empieza saltando la rana verde, Las piedras están enumeradas del 1 a $n+1$.

n = numero de ranas total

Los saltos posibles se registran con las variables $S_1, S_2, S_3, \dots, S_{4n-2}$ formando un vector de salto :Ejemplo $n=4$:

$S = [S_1; S_2; S_3; S_4; S_5; S_6; S_7; S_8; S_9; S_{10}; S_{11}; S_{12}; S_{13}; S_{14}]$

Generamos dos vectores de partida, el inicial y el final de tamaño $2*(n+1)$. Ejemplo $n=4$:

$I = [0; 0; 0; 0; 1; 1; 1; 1; 0; 0; 0; 0]$

$F = [1; 1; 0; 0; 0; 0; 0; 0; 0; 0; 1; 1]$

Construimos la tabla de posibles desplazamientos y saltos. En la que $d_{\{1,2\}}$ por ejemplo significa que la rana se desplaza desde la posición 1 a la 2, y $s_{\{3,5\}}$ significa que la rana salta desde la posición 3 a la 5. Ejemplo sacado de $n=4$:

Salto	Movimiento	Rana que se mueve
S1	$d_{\{1,2\}}$	Verde
S2	$d_{\{2,3\}}$	Verde
S3	$d_{\{3,4\}}$	Verde
S4	$d_{\{4,5\}}$	Verde
S5	$d_{\{2,1\}}$	marrón
S6	$d_{\{3,2\}}$	marrón
S7	$d_{\{4,3\}}$	marrón
S8	$d_{\{5,4\}}$	marrón
S9	$s_{\{1,3\}}$	Verde
S10	$s_{\{2,4\}}$	Verde
S11	$s_{\{3,5\}}$	Verde
S12	$s_{\{3,1\}}$	marrón
S13	$s_{\{4,2\}}$	marrón
S14	$s_{\{5,3\}}$	marrón

Construimos la matriz de desplazamientos, en la que las filas son las diferentes posiciones en las que puede estar una rana de un color, y las de otro , y las columnas son los distintos saltos que puede realizar. La matriz se rellena con los datos de la tabla de movimientos poniendo un -1 en la casilla origen y un 1 en la casilla destino. Ejemplo matriz para $n=4$:

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14
M1					1							1		
M2					-1	1							1	
M3						-1	1					-1		1
M4							-1	1					-1	
R5								-1						-1
V1	-1								-1					
V2	1	-1								-1				
V3		1	-1						1		-1			
V4			1	-1						1				
V5				1							1			

Matriz desplazamientos



Ejemplo de modelado para $n=4$

OBJETIVO:

Debemos minimizar la suma de los posibles saltos.

$$S1 + S2 + S3 + S4 + S5 + S6 + S7 + S8 + S9 + S10 + S11 + S12 + S13 + S14$$

RESTRICCIONES

Sujeto a las restricciones que resultan de multiplicar la matriz por el vector de saltos ($S1...S14$) e igualarlo al vector formado por (vector final – vector inicial):

$$\begin{array}{lll} \text{ResM1: } S5 + S12 & = & 1 \\ \text{ResM2: } -S5 + S6 + S13 & = & 1 \\ \text{ResM3: } -S6 + S7 - S12 + S14 & = & 0 \\ \text{ResM4: } -S7 + S8 - S13 & = & -1 \\ \text{ResM5: } -S8 - S14 & = & -1 \\ \text{ResV1: } -S1 - S9 & = & -1 \\ \text{ResV2: } S1 - S2 - S10 & = & -1 \\ \text{ResV3: } S2 - S3 + S9 - S11 & = & 0 \\ \text{ResV4: } S3 - S4 + S10 & = & 1 \\ \text{ResV5: } S4 + S11 & = & 1 \end{array}$$



Acotamos el numero de veces que puede repetirse una variable a por ejemplo 20.

S1<=20
S2<=20
S3<=20
S4<=20
S5<=20
S6<=20
S7<=20
S8<=20
S9<=20
S10<=20
S11<=20
S12<=20
S13<=20
S14<=20

Sin olvidarnos de que para cualquier numero de ranas siempre empieza moviéndose la rana situada a la izquierda de la piedra central, por lo tanto obligatoriamente debe ser mayor que 1, también debemos tener en cuenta que los saltos de las ranas marrones – saltos ranas verdes = 0 para $(n/2)$ =valor par, y saltos marrones - saltos verdes = 1 para $(n/2)$ =valor impar.

S0: S2>=1

ST: $(S12+S13+S14)-(S9+S10+S11) = 0$

4. Modelo en formato LP para LP_Solve con n=4

Min: $S1 + S2 + S3 + S4 + S5 + S6 + S7 + S8 + S9 + S10 + S11 + S12 + S13 + S14$;

S0: $S2 \geq 1$;

ST: $+ S12 + S13 + S14 - S9 - S10 - S11 = 0$;

M1: $S5 + S12 = 1$;

M2: $- S5 + S6 + S13 = 1$;

M3: $- S6 + S7 - S12 + S14 = 0$;

M4: $- S7 + S8 - S13 = -1$;

M5: $- S8 - S14 = -1$;

V1: $- S1 - S9 = -1$;

V2: $S1 - S2 - S10 = -1$;

V3: $S2 - S3 + S9 - S11 = 0$;

V4: $S3 - S4 + S10 = 1$;

V5: $S4 + S11 = 1$;

$S1 \leq 20$;

$S2 \leq 20$;

$S3 \leq 20$;

$S4 \leq 20$;

$S5 \leq 20$;

$S6 \leq 20$;

$S7 \leq 20$;

$S8 \leq 20$;

$S9 \leq 20$;

$S10 \leq 20$;

$S11 \leq 20$;

$S12 \leq 20$;

$S13 \leq 20$;

$S14 \leq 20$;

int

$S1, S2, S3, S4, S5, S6, S7, S8, S9, S10, S11, S12, S13, S14$;

5. Modelo en formato LP para XpressMP con n=4

Minimize

$$S1 + S2 + S3 + S4 + S5 + S6 + S7 + S8 + S9 + S10 + S11 + S12 + S13 + S14$$

Subject To

$$S0: S2 \geq 1$$

$$M1: S5 + S12 = 1$$

$$M2: -S5 + S6 + S13 = 1$$

$$M3: -S6 + S7 - S12 + S14 = 0$$

$$M4: -S7 + S8 - S13 = -1$$

$$M5: -S8 - S14 = -1$$

$$V1: -S1 - S9 = -1$$

$$V2: S1 - S2 - S10 = -1$$

$$V3: S2 - S3 + S9 - S11 = 0$$

$$V4: S3 - S4 + S10 = 1$$

$$V5: S4 + S11 = 1$$

$$ST: +S12 + S13 + S14 - S9 - S10 - S11 = 0$$



Bounds

S1 <= 20

S2 <= 20

S3 <= 20

S4 <= 20

S5 <= 20

S6 <= 20

S7 <= 20

S8 <= 20

S9 <= 20

S10 <= 20

S11 <= 20

S12 <= 20

S13 <= 20

S14 <= 20

Integers

S1 S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 S12 S13 S14

End

7. Generador de ficheros .lp .mps

Hemos construido un programa en C++ que ofrece la posibilidad de crear los ficheros .lp para LP_Solve y el .lp y .mps para XPressMP. Para ello hay que introducirle el número total de ranas que queremos y elegir el correspondiente formato. El código de su funcionamiento queda reflejado en el fichero generador.cpp

8. Interpretación de soluciones

Una vez obtenidos los ficheros .lp procedemos a pasárselos al resolutor LP_Solve, el cual nos permite obtener con un simple clic el mismo modelo pero en formato .mps, o se lo pasamos al XPressMP. El resolutor una vez ejecutado nos ofrece las variables de la solución y el número de veces que se repite, en el caso que llevamos analizando desde el principio $n=4$ usando el resolutor XPressMP nos muestra la siguiente tabla de resultados:

Name	Solution	Reduced cost
S2	1	1e+030
S3	1	1e+030
S5	1	1e+030
S8	1	1e+030
S9	1	1e+030
S11	1	1e+030
S13	2	1e+030

De la que deducimos que las variables de la solución son S2,S3,S5,S8,S9,S11,S13 observando que S13 se repite dos veces. Estos valores están desordenados y no representan exactamente la secuencia de saltos correspondientes para resolver el problema, por lo que necesitamos un interpretador de soluciones.



El interpretador de soluciones que hemos creado sigue los siguientes pasos:

- 1.- Pide al usuario el numero de ranas del modelo.
- 2.- Pide al usuario el numero de variables obtenidas en el resolutor.
- 3.- Va pidiendo al usuario el índice de las variables de la solución.

Declaramos una matriz[numero soluciones][3] .

Para cada índice introducido mediante una función decodificaX y otra decodificaY , obtenemos los valores de la tabla de desplazamiento. Por ejemplo n=4:

S2 índice 2 valor x=2, valor y=3 d_(2,3)

S8 índice 8 valor x=5, valor y=4 d_(5,4)

Que vamos guardando en matriz[][0] la x, matriz[][1] la y, en matriz[][2] guardamos un 0, valor que nos servirá en el futuro para saber si un par ha sido seleccionado o no.

4.- Una vez rellena la matriz con todos los movimientos x,y de la solución. Comprobamos si en la matriz existe el valor $(n/2, (n/2)+1)$ que se corresponde con el primer movimiento obligatorio para cualquier valor de n. Si no existe, “solución incorrecta”. Si existe ponemos $m[][2] = 1$ lo que nos indica que lo hemos visitado.



5.- Vamos llamando a una función recursiva que nos comprueba si existe un movimiento siguiente al actual en la matriz.

Si existe lo marca a 1, y hace la llamada recursiva.

Si no existe y están todos marcados a 1 hemos terminado “solución correcta”.

Si existe pero está marcado a 1 y hay alguno a 0 “solución incorrecta”.

Si tenemos por ejemplo el par (2,3), en la llamada recursiva busca el par que tenga como valor $y=2$

Como resultado el interpretador muestra la secuencia de movimientos que hay que seguir para resolver el problema y si corresponden con las variables introducidas al principio da como correcta la solución, sino la da como incorrecta.



9. Referencias bibliográficas:

Enlace para jugar con $n=6$, en esta versión no se puede deshacer un movimiento

<http://infantiles.juegorama.com/infantiles/ver-juego-1286.html>

Página de donde surge la idea de hacer este juego

http://www.sinewton.org/numeros/numeros/65/matematicas_01.php

10. Agradecimientos:

Queremos dar las gracias al profesor Pablo Guerrero por todas las ideas, comentarios, rectificaciones y mejoras que nos ha ofrecido para poder terminar y modelar correctamente la rana saltarina. Sin su desinteresada colaboración este proyecto no se podría haber terminado a tiempo. Muchas Gracias.