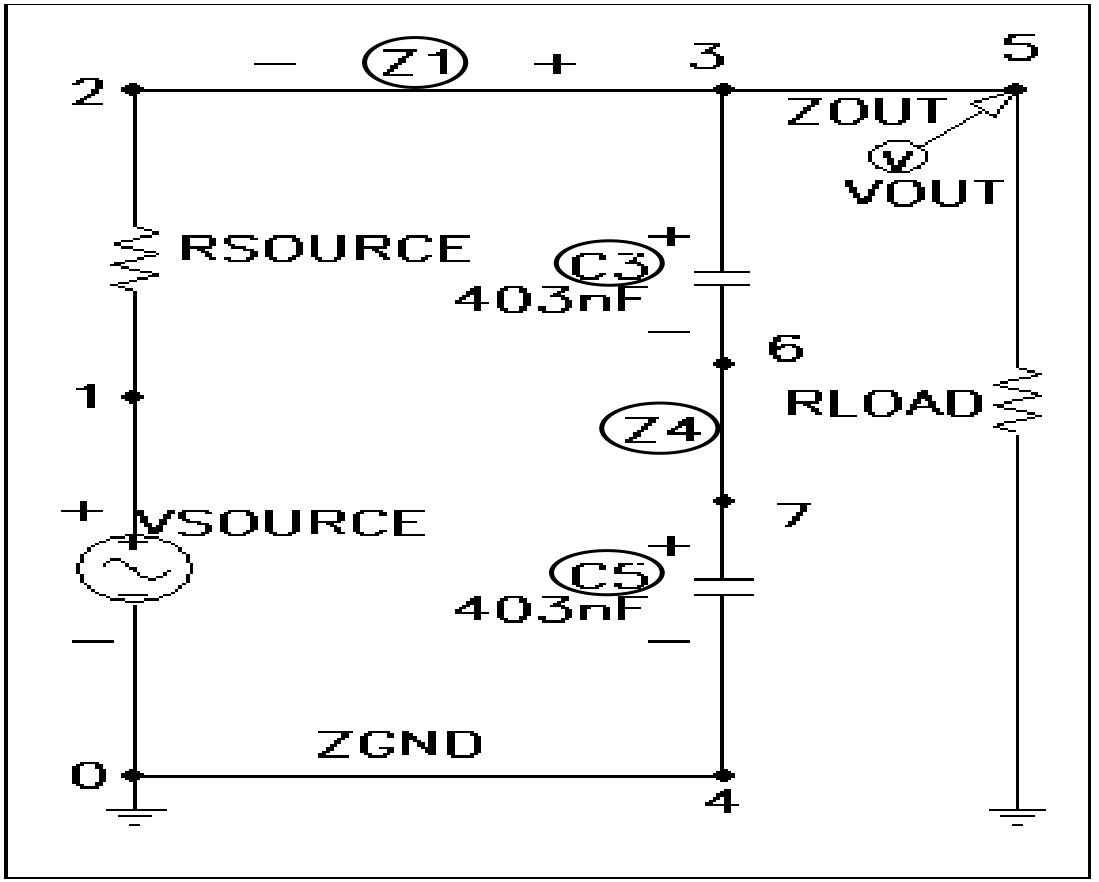
# Programación Genética Aplicada a la Síntesis de Circuitos Analógicos

## El Diseño de Circuitos Analógicos

- A diferencia del diseño digital, es un domino complicado, que hacen unos pocos expertos
- Elementos: resistencias (R), condensadores (C), inductancias (L), transistores

# Ejemplo de Circuito

Fuente de la imagen: Genetic Programming III: Darwinian Invention and Problem Solving by John R. Koza, Forrest H. Bennett III, David Andre, and Martin A. Keane. Morgan Kaufmann 1999.



# ¿Cómo representar circuitos con árboles?

- Difícil: los circuitos son grafos
- Además, hay que garantizar que los circuitos evolucionados son correctos
- Solución: Programación Genética “con embriones” (*developmental GP*)
- Un individuo no es un circuito, sino un programa para construir un circuito

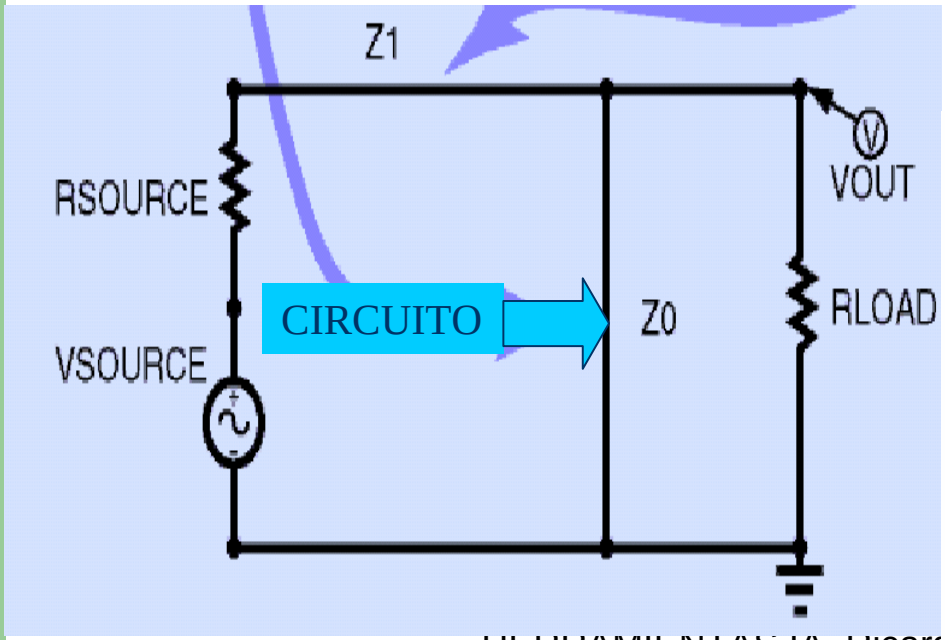
# Operaciones de desarrollo de “embriones” (circuitos)

- Modificación de la topología: serie, paralelo, flip (cambio de polaridad)
- Creación de componentes y sus valores: Resistencia, Condensador, L-inductancia
- Control del desarrollo: NOP, END
- Subrutinas
- Funciones aritméticas

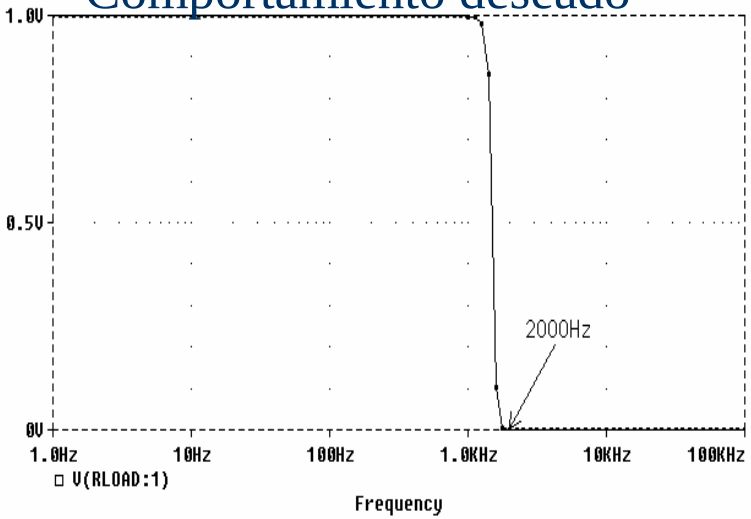
# Cálculo de la Fitness

- Se simula el circuito con un simulador de circuitos (SPICE)
- Se compara la onda de salida con la esperada.

Fuente de la imagen: Genetic Programming III: Darwinian Invention and Problem Solving by John R. Koza, Forrest H. Bennett III, David Andre, and Martin A. Keane. Morgan Kaufmann 1999.



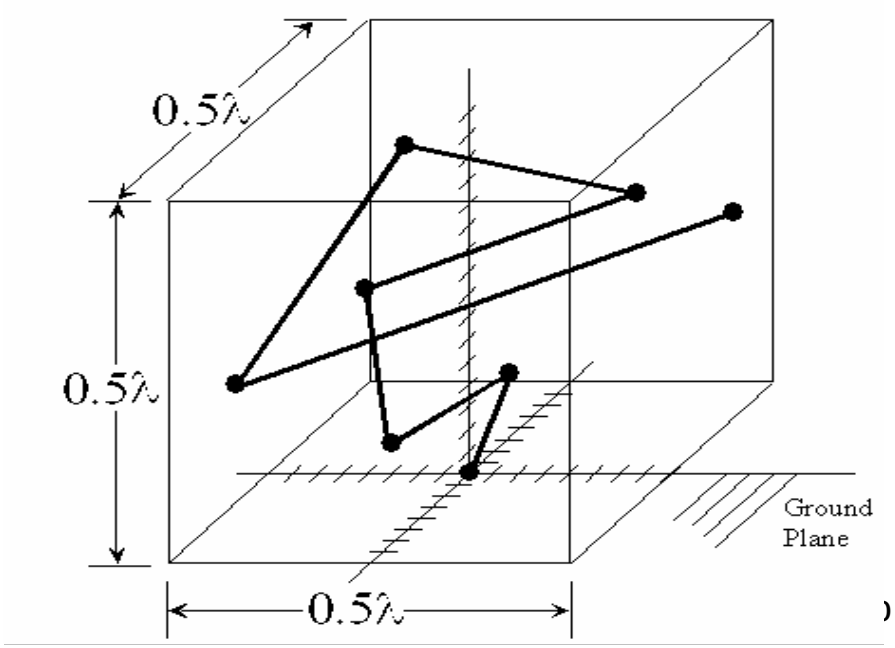

Comportamiento deseado

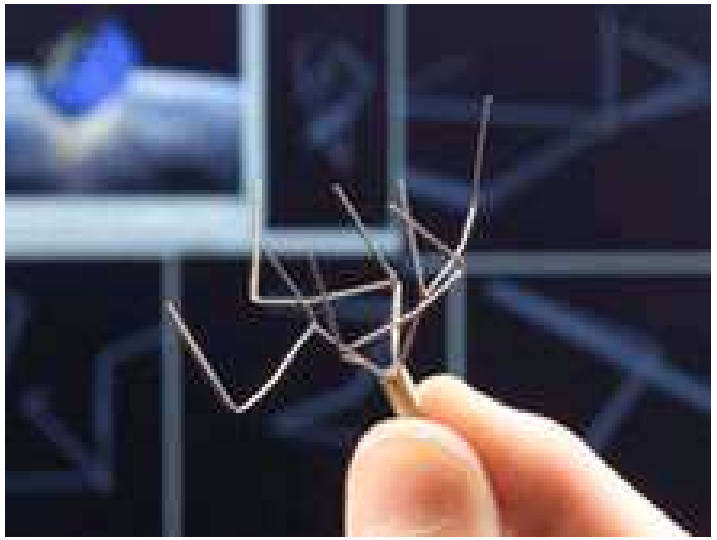


# PG aplicada al desarrollo de antenas

Fuente de la imagen: Genetic Programming IV: Darwinian Invention and Problem Solving by John R. Koza et al.. Springer 2005.

- GP “con embriones”
- Lanzada en el satélite ST5, Marzo 2006

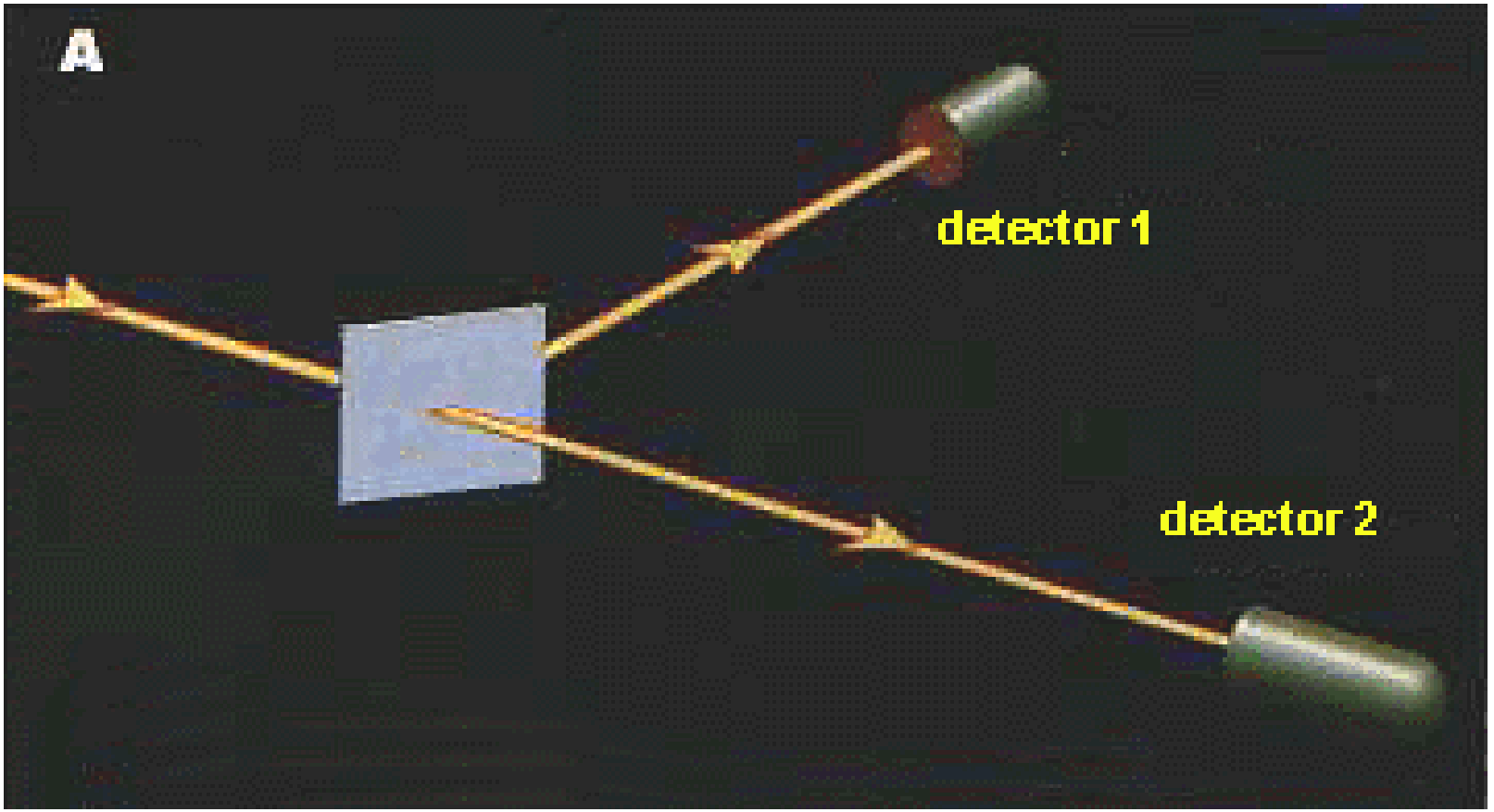


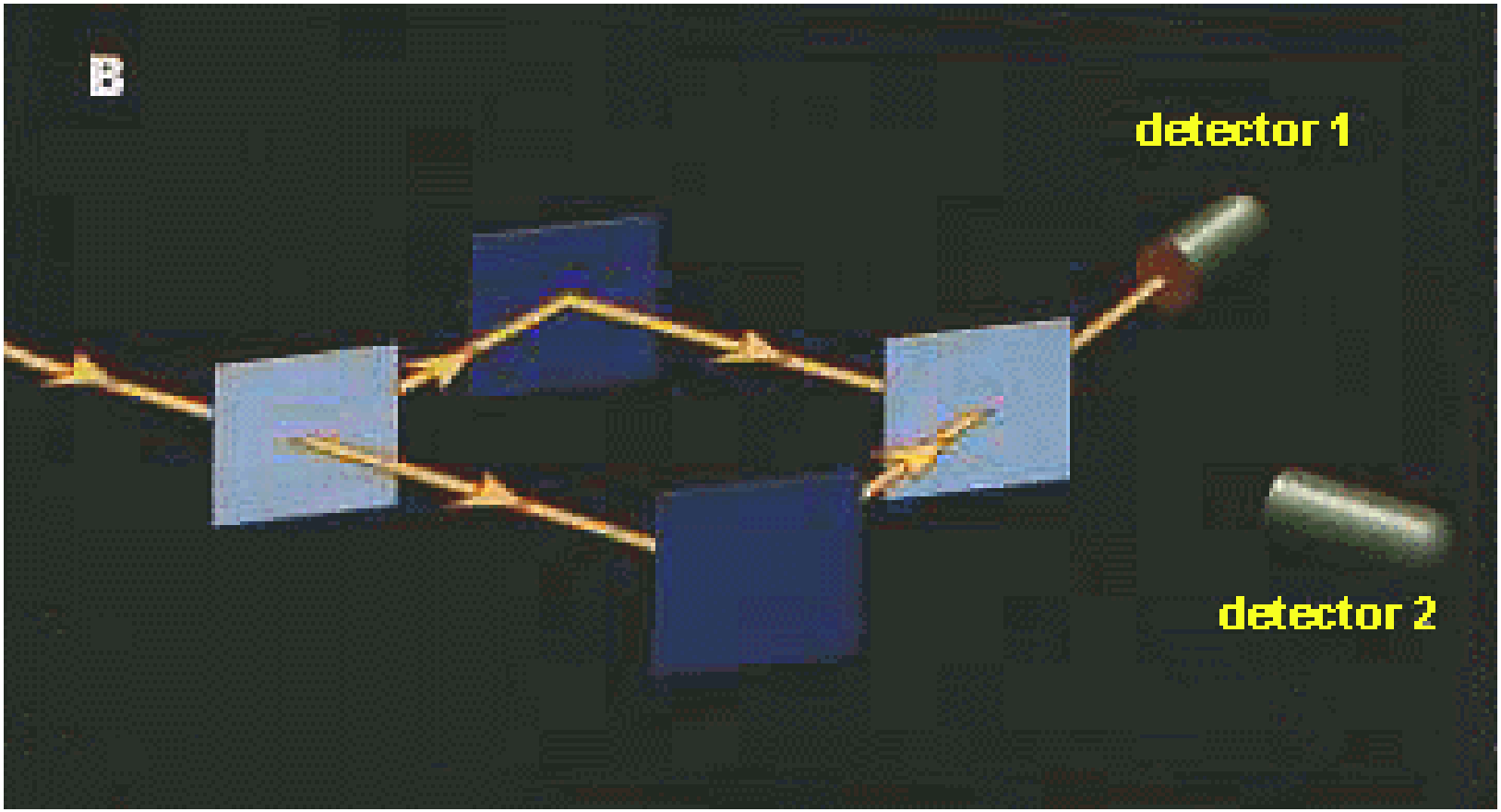
# PG aplicada a la computación cuántica

- En ordenadores clásicos, la información se almacena en binario (0 o 1)
- En mecánica cuántica, un sistema puede estar, además, en 0 **y** 1 **al mismo tiempo**
- Esto se denomina quantum bit o qubit

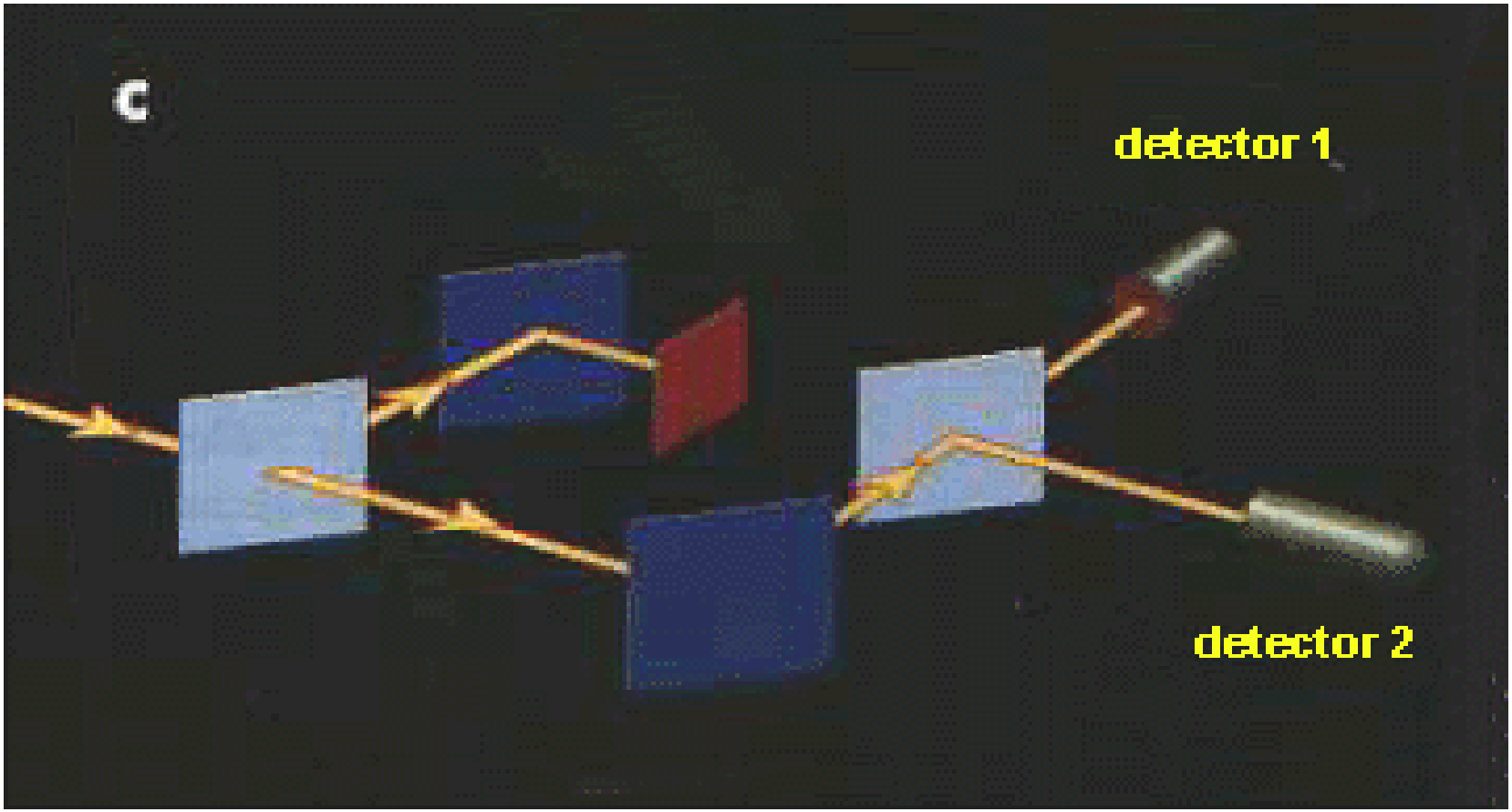
# Ejemplo del espejo translucido



# ¿Por qué camino va el fotón?

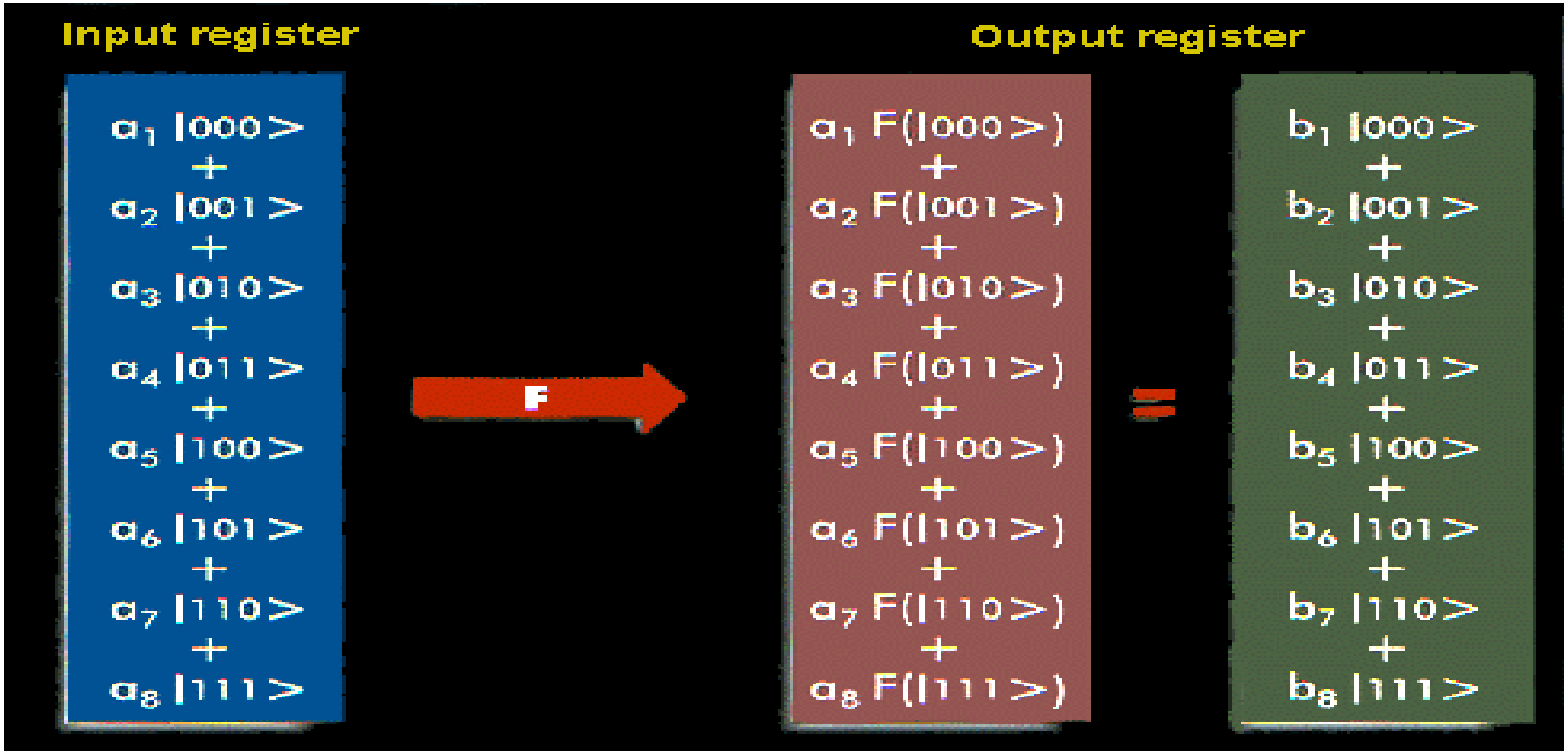


# El fotón va por los dos caminos (superposición de estados)

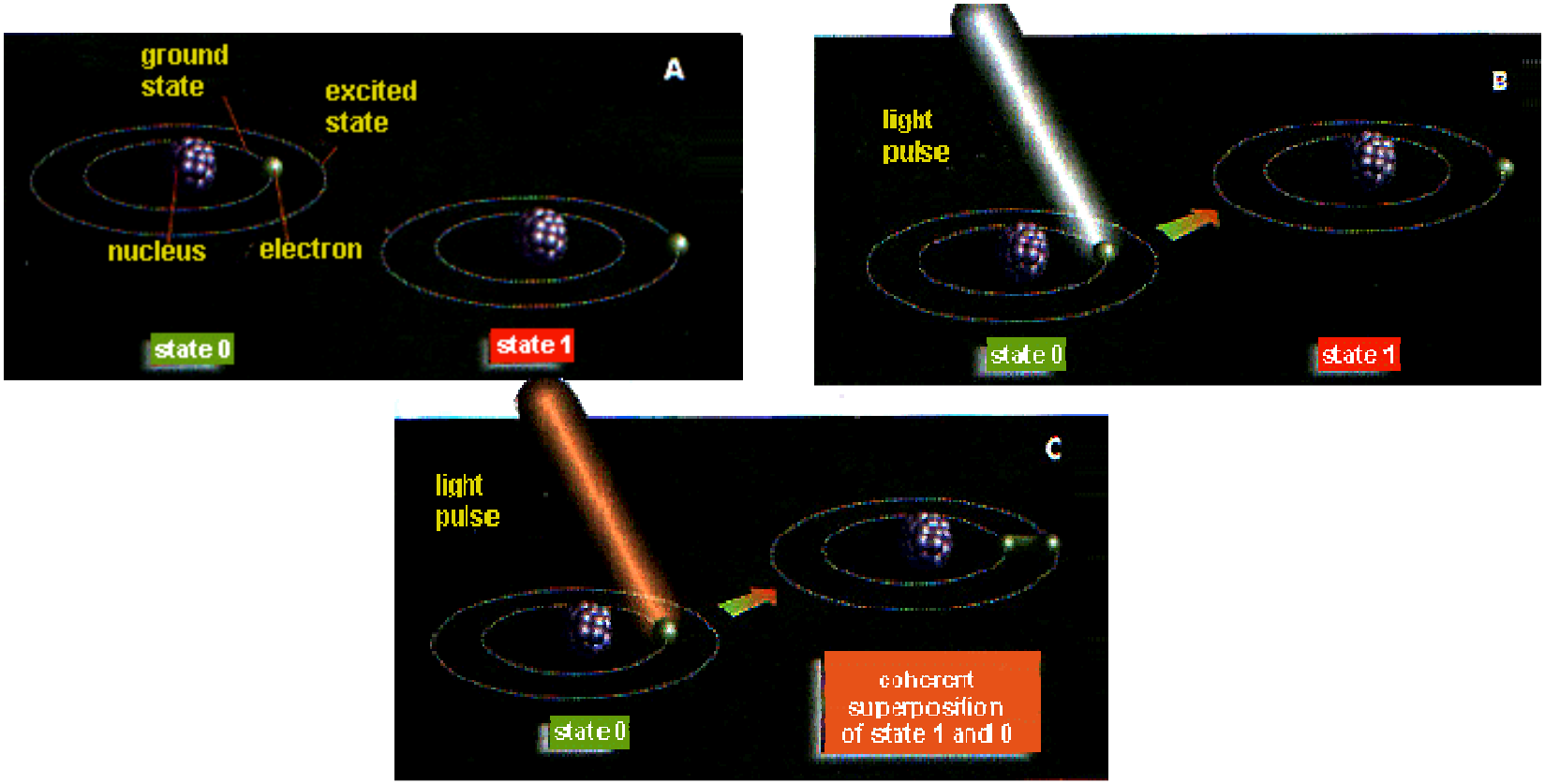


- http://www.dailymotion.com/video/x4nr1i_mecanica-cuantica-doble-ranura_tech

# Almacenamiento simultaneo de $2^L$ valores



# QBIT


ground state
excited state
A
nucleus
electron
state 0
state 1
light pulse
B
state 0
state 1
light pulse
C
state 0
coherent superposition of state 1 and 0

# Potencia de los computadores cuánticos

- En principio, un ordenador cuántico puede aplicar un programa P(x) simultáneamente a los $2^L$ valores en superposición
- Es posible pasar problemas de **NP** a **P**
- En la prácticas esto significa convertir problemas exponenciales en polinomiales
- Con un computador clásico, haría falta un número exponencial de procesadores

# Ejemplo: factorización

- Factorización: resolver el problema:
  - ? * ? = 29083 (127 x 129 = 29083 )
- Es un problema NP
- Factorizar un número de 30 dígitos es $10^{13}$ más costoso (en tiempo y memoria) que uno de 3 dígitos
- La criptografía depende de que este problema no sea resoluble en la práctica (RSA, bancos en Internet)

# Utilidad de la computación cuántica

- La computación cuántica no fue tomada en serio hasta que apareció el algoritmo cuántico de **Shor**, que hace factorización en un tiempo “razonable”:

$$O(2^{n^{\frac{1}{3}}\log(n)^{\frac{2}{3}}}) \rightarrow O(n^2 \log(n) \log\log(n))$$

- Para descibrar un mensaje de 1024 bits, se tardarían 100000 años con un ordenador clásico y 5 minutos con uno cuántico

# Algoritmo de Grover

- Grover encuentra elementos en listas desordenadas en $O(\sqrt{n})$ (lo normal es $O(n)$)

# Computación cuántica

- El estado inicial puede ser un estado no superpuesto:
  - Ej: el fotón acaba de ser emitido por el laser
- Este estado puede evolucionar hacia estados superpuestos:
  - Ej: el fotón yendo por un camino y por el otro: $E = \alpha_1 *|izquierda> + \alpha_2 *|derecha>$
- Se hace pasar al estado superpuesto por el "programa" (un conjunto de puertas cuánticas)
- Finalmente se realiza una medición y E "colapsa" a un estado no superpuesto, de manera probabilística, con probabilidad $| \alpha_i |^2$

# Computación cuántica

- El estado final que se mide es probabilístico
- Hay que conseguir que la solución sea el estado más probable
- Se puede conseguir por medio de interferencia, para cancelar aquellos estados que no representan la solución

# Computación cuántica

- Estado cuántico de n bits = superposición de $2^n$ estados, cada uno con su amplitud $\alpha_i$
- Cada amplitud es un número complejo (del tipo a + b*i)
- Con dos bits:
  - $E = \alpha_1 *|00> + \alpha_2 *|01> + \alpha_3 *|10> + \alpha_4 *|11>$
  - $E = (\alpha_1 , \alpha_2 , \alpha_3 , \alpha_4 )$
- El módulo $| \alpha_1 |^2$ es la probabilidad de encontrar al estado E en |00>, cuando realicemos una medición (se dice que “la función de onda colapsa”)

# Computación cuántica

- Utiliza q-gates, que toman 2 q-bits a la entrada y generan 2 q-bits a la salida
- Todas las puertas se pueden representar con matrices G
- Nuevo estado = antiguo estado * G

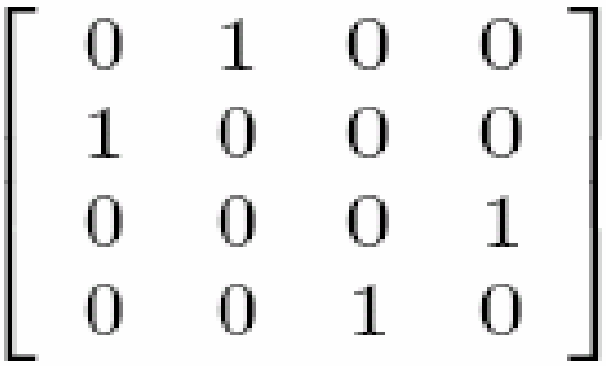
# Computación cuántica. Puerta NOT

- 1 BIT:
  - $\alpha_1 *|0> + \alpha_2 *|1> => \alpha_2 *|0> + \alpha_1 *|1>$

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix} = \begin{bmatrix} \alpha_1 \\ \alpha_0 \end{bmatrix}$$

# Computación cuántica. Puerta NOT

- 2 BITS: (invierte sólo el primer bit)
  - $\alpha_1 *|00> + \alpha_2 *|01> + \alpha_3 *|10> + \alpha_4 *|11>$
  - =>
  - $\alpha_3 *|00> + \alpha_4 *|01> + \alpha_1 *|10> + \alpha_2 *|11>$



$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

# Otras Qpuertas

Controlled NOT

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

NAND 3-bits

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

HADAMARD

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

ROTACIÓN

$$U_\theta = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}$$

$$CPHASE = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & e^{i\alpha} \\ 0 & 0 & e^{-i\alpha} & 0 \end{bmatrix}$$

$$U_2 = \begin{bmatrix} e^{-i\phi} & 0 \\ 0 & e^{i\phi} \end{bmatrix} \times \begin{bmatrix} \cos(\theta) & \sin(-\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \times \begin{bmatrix} e^{-i\psi} & 0 \\ 0 & e^{i\psi} \end{bmatrix} \times \begin{bmatrix} e^{i\alpha} & 0 \\ 0 & e^{i\alpha} \end{bmatrix}$$

# Ejemplo de programa cuántico y su traza

COMIENZO = $1|00\rangle + 0|01\rangle + 0|10\rangle + 0|11\rangle$

Hadamard qubit:0 $\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|01\rangle + 0|10\rangle + 0|11\rangle$

Hadamard qubit:1 $\frac{1}{2}|00\rangle + \frac{1}{2}|01\rangle + \frac{1}{2}|10\rangle + \frac{1}{2}|11\rangle$

U-theta qubit:0 theta:pi/5 $0.698|00\rangle + 0.111|01\rangle + 0.698|10\rangle + 0.111|11\rangle$

Controlled-not control:1 target: $0.698|00\rangle + 0.111|01\rangle + 0.111|10\rangle + 0.698|11\rangle$

Hadamard qubit:1 $0.572|00\rangle + 0.572|01\rangle + 0.416|10\rangle - 0.416|11\rangle.$

# Estado final del programa cuántico

$$0.572|00\rangle + 0.572|01\rangle + 0.416|10\rangle - 0.416|11\rangle.$$

| state | probability |
|---|---|
| $\|00\rangle$ | 0.33 |
| $\|01\rangle$ | 0.33 |
| $\|10\rangle$ | 0.17 |
| $\|11\rangle$ | 0.17 |

Prob(0*) = 0,33+0,33 = 0,66

Prob(*1) = 0,5

Es decir, comenzamos con |00> y terminamos con, seguramente |00> o |01>

# Programación cuántica

- No resulta intuitiva para programadores humanos
- Los problemas opacos son apropiados para la PG
- Al fin y al cabo se trata de evolucionar un circuito con puertas que puede ser simulado
- Se utiliza la técnica “embrionaria”, incluyendo bucles
- El simulador de puertas QGAME usado para la fitness es exponencialmente lento (porque el tamaño de las matrices crece exponencialmente con n, $2^n$)
- Actualmente, ordenador de 12 Qbits. Problema: mantener el estado cuántico superpuesto sin interferencias del exterior
- Estudios teóricos muestran que la ganancia en problemas de búsqueda NP-completos serían, como mucho, cuadráticos (sqrt(n)), no exponenciales (log(n))

# Ejemplo: buscar un elemento en una lista no ordenada

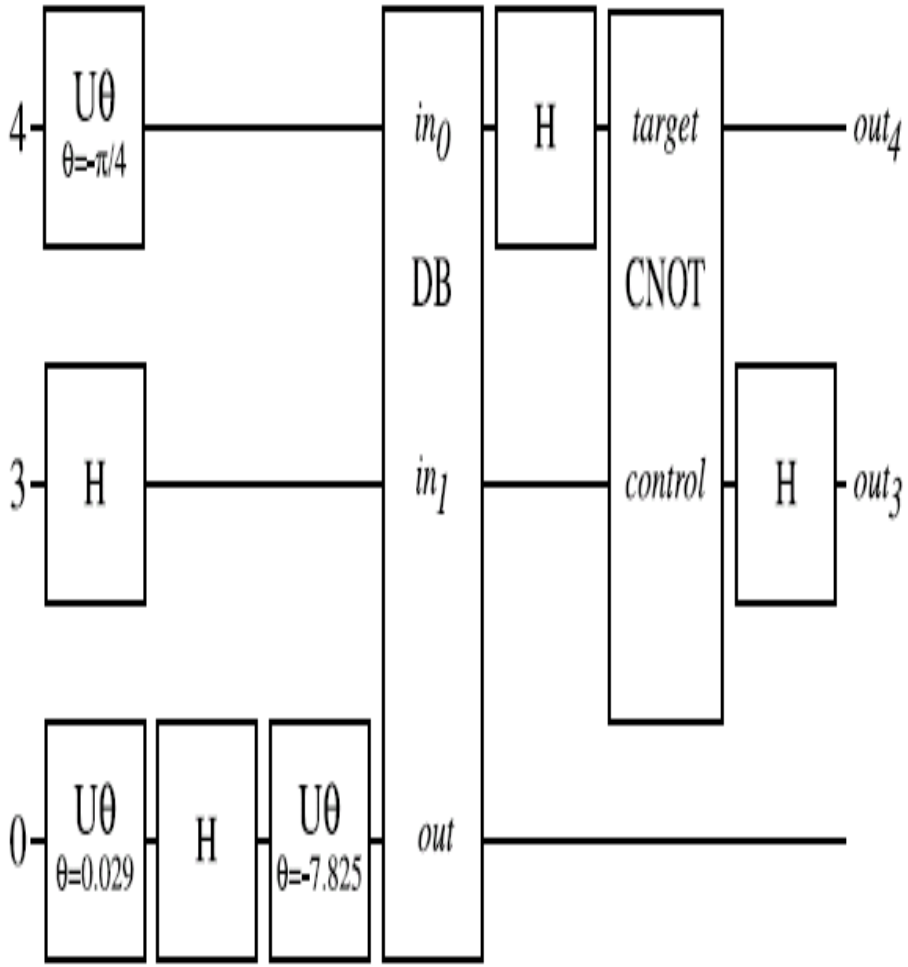
- Ejemplo de lista: [10,00,11,01]
- Si la lista tiene cuatro elementos, será necesario acceder al menos tres veces
- Se consiguió un algoritmo cuántico que necesita sólo **un acceso** y es prácticamente determinista

*MidGP* parameters for a run on the four-item database search problem.

| | |
|---|---|
| max number of generations | 1,001 |
| size of population | 1,000 |
| max program length | 256 |
| reproduction fraction | 0.5 |
| crossover fraction | 0.1 |
| mutation fraction | 0.4 |
| max mutation points | 127 |
| selection method | tournament (size=5) |
| function/terminal set | `noop`, `+`, `-`, `*`, `%p`, `DB-gate`, `H-gate`, `U-theta-gate`, `CNOT-gate`, `CPHASE-gate`, `U2-gate`, 0, 1, 2, 3, 4, $\pi$, `ephemeral-random-constant`, `pop` |

# Ejemplo de circuito cuántico evolucionado

```
U2 qubit:4 phi:0 theta:3 psi:3.14159 alpha:0.25908
Controlled-phase control-qubit:3 target-qubit:4, alpha:39.54646
Controlled-not control-qubit:0 target-qubit:3
U-theta qubit:0 theta:0.02934
Hadamard qubit:3
U-theta qubit:4 theta:3.14159
Hadamard qubit:0
Controlled-not control-qubit:1 target-qubit:3
U-theta qubit:4 theta:-4.06820
U-theta qubit:0 theta:-7.82538
Database-lookup input-qubits:4,3 output-qubit:0
Hadamard qubit:4
U-theta qubit:1 theta:4
U-theta qubit:3 theta:0
Controlled-phase control-qubit:3 target-qubit:4, alpha:0
Hadamard qubit:3
(read output from qubits 3 and 4)
```



# PG en computación cuántica

- Creation of a better-than-classical quantum algorithm for the Deutsch-Jozsa “early promise” problem (B, F)
- Creation of a better-than-classical quantum algorithm for Grover’s database search problem (B, F)
- Creation of a quantum algorithm for the depth-two AND/OR query problem that is better than any previously published result (D)
- Creation of a quantum algorithm for the depth-one OR query problem that is better than any previously published result (D)
- Creation of a protocol for communicating information through a quantum gate that was previously thought not to permit such communication (D)
- Creation of a novel variant of quantum dense coding (D)

# Algunas referencias

- Tutorial de pg aplicado a computación cuántica
- http://portal.acm.org/citation.cfm?id=1274128
- *Automatic Quantum Computer Programming: A Genetic Programming Approach.* By Lee Spector. Kluwer Academic Publishers, 2004, and Springer Science+Business Media, 2007.
- *A review of procedures to evolve quantum algorithms*. Adrian Gepp & Phil Stocks. Genetic Programming Evolvable Machines (2009) 10:181–228

# Criterios para “human-competitive”

- **(A)** Patentado previamente o podría ser una patente
- **(B)** Igual o mejor que un resultado científico ya publicado
- **(C)** Igual o mejor que resultados almacenados en bases de datos expertas
- **(D)** El resultado es publicable
- **(E)** Igual o mejor que resultados recientes en un problema atacado por una serie de algoritmos previos
- **(F)** Igual o mejor que algo que fue considerado un éxito en su momento
- **(G)** El resultado soluciona un problema de dificultad probada en un campo
- **(H)** El resultado gana o empata en una competición con humanos o programas desarrollados por humanos

# Éxitos de la PG

- Creación de algoritmos cuánticos mejores que los existentes (B, F, D)
- Aplicación a la Robosoccer (H)
- Aplicaciones a bioinformática (B, E)
- Aplicaciones a la síntesis de circuitos y antenas (A, B, D, E, F, G)
- Paralelización de programas de ordenador

# Premios de $10000 del 2005 y 2006 para resultados “Human-Competitive”

- http://www.genetic-programming.org/
  - hc2005/cfe2005.html
  - hc2006/cfe2006.html

# Esfuerzo Computacional y Resultados en Programación Genética

| System | Fechas | Speed-up | Invenciones | Categoría problemas y resultados |
|---|---|---|---|---|
| LISP | 87–94 | 1 (base) | 0 | De juguete |
| 64 transputers | 94–97 | 9 | 2 | Competitivos, pero no patentados |
| 64 PowerPC’s | 95–00 | 204 | 12 | Patentes siglo XX |
| 70 Alpha’s | 99–01 | 1,481 | 2 | Patentes siglo XX |
| 1,000 Pentium II’s | 00–02 | 13,900 | 12 | Patentes siglo XXI |
| 4-week runs on 1,000 Pentium II’s | 02-03 | 130,000 | 2 | Nuevas patentes |

# Conclusiones PG I

- El objetivo de la PG es la creación de programas de ordenador de manera automática
- Permite usar variables y subrutinas. Bucles y recursividad, menos probado
- No tiene una fuerte base teórica. Seguramente hay mejores maneras de hacer programación automática
- ¿Pero cómo? (ADATE, ILP, ...)
- La PG funciona, resuelve problemas complejos e incluso innova en ciencia e ingeniería

# Conclusiones PG II

- La PG requiere de un **gran** esfuerzo computacional
- En problemas complejos, es conveniente guiar, utilizando funciones potentes o evolución dirigida
- Usarla en problema donde uno sabe qué quiere pero no como programarlo
- Ojo, ¡no suele funcionar a la primera!. Hay que probar distintas representaciones, parámetros, etc.
- Se puede aplicar PG a subproblemas del problema principal