

# Inteligencia Artificial

**Unidad 1:** Representación Avanzada del Conocimiento

**TEMA 2:** Algoritmos de la IA Clásica

**Módulo 5:** Algoritmo de búsqueda Simulated Annealing

Unidad 1

# Representación Avanzada del Conocimiento

**TEMA 2:** Algoritmos de la IA Clásica

Sesión 8

## MÓDULO 5: Algoritmo de Búsqueda Simulated Annealing



### Contenido

1. Origen del Algoritmo Simulated Annealing
2. ¿Cómo funciona este Algoritmo?
3. Fases en el Algoritmo Simulated Annealing
4. Ejemplos de aplicación



### Preguntas

# 1. Origen del Algoritmo Simulated Annealing

Este algoritmo tiene su origen en la metalurgia.

- En metalurgia, el recocido se refiere al proceso de calentar el metal a una temperatura alta y luego enfriarlo lentamente en un ambiente controlado.
- A mayor temperatura, los átomos aumentan su energía del metal se mueven velozmente.
- Si el metal se enfriara rápidamente, los átomos se detendrían repentinamente donde quiera que estuvieran (y el metal no guardaría una forma uniforme). Se produciría una disposición aleatoria de los átomos y un resultado de mala calidad.
- En cambio, si el metal se enfría lentamente, los átomos tienen tiempo para encontrar gradualmente la mejor orientación posible y alinearse en una buena forma o estado.
- Entonces, es preferible el enfriamiento lento que el rápido, porque hace que el metal sea más dúctil, reduce los defectos y lo hace significativamente más fuerte (\*).



(\*) Fuente: Verhoeven, JD *Fundamentals of Physical Metalurgy*, Wiley, Nueva York, 1975, p. 326

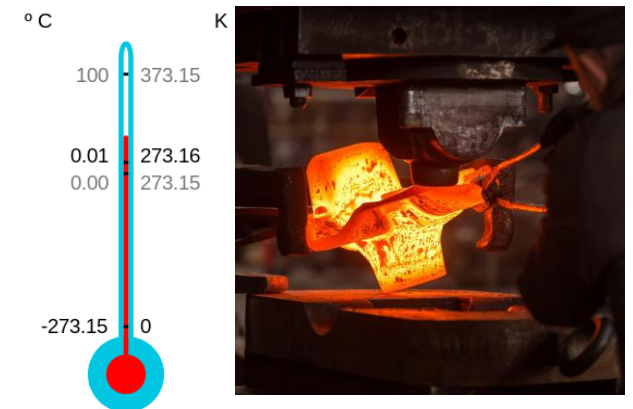
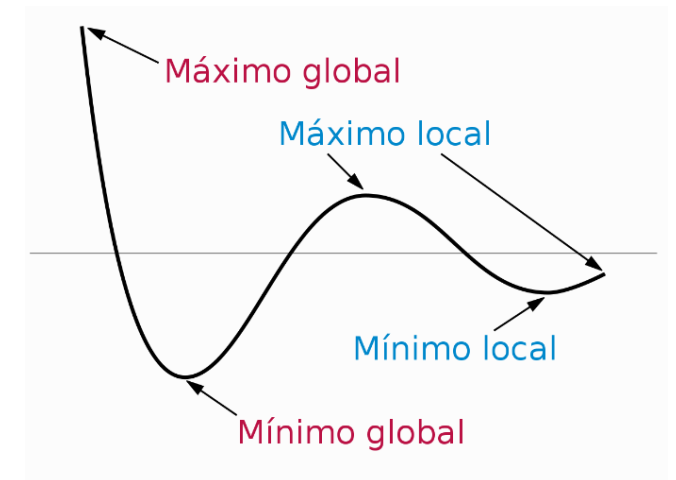
# 1. Origen del Algoritmo Simulated Annealing en I.A.

El **Algoritmo Simulated Annealing en I.A.** es un algoritmo que se inspira en la recocido de los metales para **resolver problemas de optimización global** (encontrar mínimos y máximos globales).

- Inspirado en el proceso físico de enfriamiento controlado de metales.
- Si el enfriamiento es adecuado, se obtiene la estructura de menor energía (mínimo global)

**Simulated Annealing** es un algoritmo Hill Climbing **estocástico** (casual).

**Objetivo:** Evitar el problema de los máximos (o mínimos) locales de la Escalada de la Colina.  
**Debe encontrar mínimos y máximos globales.**



# 1. Origen del Algoritmo Simulated Annealing en I.A.

Es una adaptación del algoritmo **Metropolis-Hastings** (1953), que simulaba el proceso de recocido.

**Metropolis-Hastings** logra verificar que un pequeño desplazamiento aleatorio de un átomo resulta en un cambio de energía. Así:

- Si el cambio de energía es negativo, el estado energético de la nueva configuración es menor y se acepta la nueva configuración.
- Si el cambio de energía es positivo, la nueva configuración tiene un estado energético superior.

La adaptación a **Simulated Annealing** (1983-85) incorpora el **factor de probabilidad de Boltzmann** para validar la función de aceptación (de un mejor estado):

$$P = e^{-\frac{\Delta E}{T}}$$

Donde:

$\Delta E$  = Es la diferencia de energía entre dos estados

$T$  = Temperatura

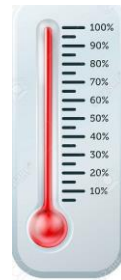
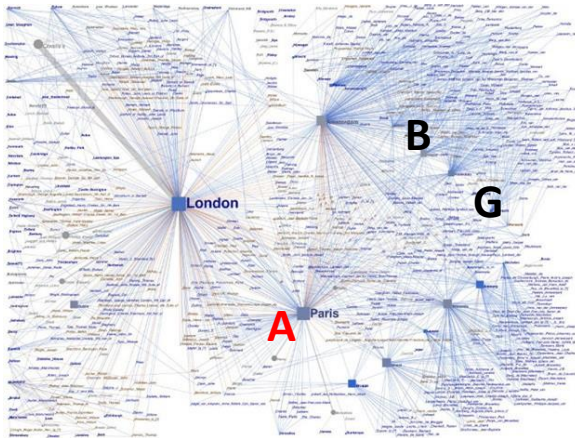
Al examinar esta ecuación, debemos notar dos cosas:

- La probabilidad es proporcional a la temperatura: a medida que el sólido (metal) se enfría, la probabilidad se reduce.
- La probabilidad es inversamente proporcional a la energía: a medida que el cambio de energía es mayor, la probabilidad de aceptar el cambio se reduce.

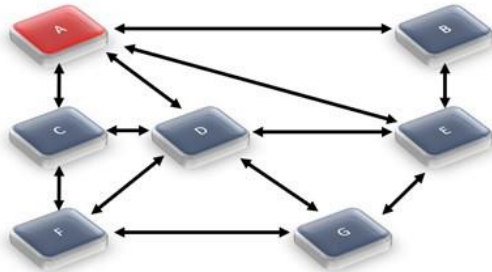
>>> **A menor temperatura, menor probabilidad de elegir sucesores peores** <<<

# 1. Origen del Algoritmo Simulated Annealing en I.A.

**Caso del Guía de Turismo:** Llevar a un grupo de pasajeros partiendo del punto **A** hacia **B** y **G** retornando finalmente al punto inicial **A**. Debe recorrer la menor distancia en el menor tiempo posible.



Imaginemos muchas conexiones/rutas como los átomos que se mueven a gran velocidad cuando la temperatura es alta



A medida que disminuimos la temperatura (energía) nos quedaremos con pocas conexiones/rutas, así como los átomos encuentran un estado final óptimo cuando menor temperatura reciban.

Simulamos el proceso de recocido en algunos sistemas de alta temperatura, pero al ir disminuyendo la temperatura logramos finalmente establecer una solución definitiva.

## 2. ¿Cómo funciona este Algoritmo?

### Metodología

- a) **Temperatura:** parámetro de control principal.
- b) **Energía:** función heurística sobre la calidad de la solución  $f'(n)$ .
- c) **Función de aceptación:** permite escoger el nodo sucesor (puede ser peor).

$$P = e^{-\frac{\Delta E}{T}}$$

Donde:

$\Delta E$  = Es la diferencia de energía entre dos estados

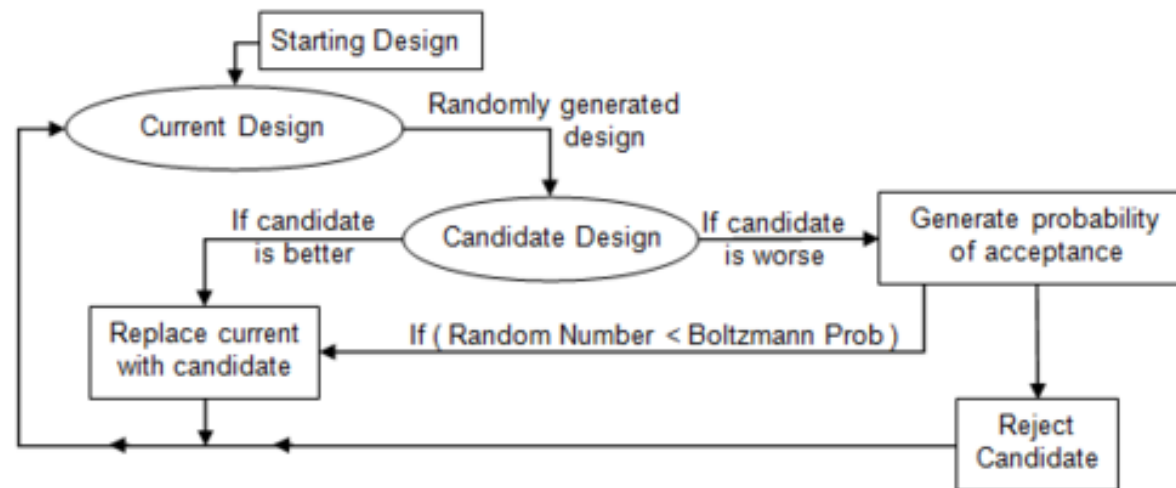
$T$  = Temperatura

### d) **Estrategia de enfriamiento:**

- El número de iteraciones de la búsqueda.
  - Cómo disminuir de la temperatura.
  - Cuántos sucesores explorar para cada paso de temperatura.
- e) Se hacen **pasos aleatorios** por el espacio de soluciones.

## 2. ¿Cómo funciona este Algoritmo?

### Simulated Annealing - Flujo





### 3. Fases en el Algoritmo Simulated Annealing

Se consideran nueve fases en este algoritmo:

1. Inicializar Temperatura
2. Establecer Tour Aleatorio
3. Inicializar Tours
4. Elegir dos nodos
5. Establecer Nuevo Tour
6. Energía
7. Función de Aceptación
8. El mejor
9. Actualizar temperatura

¿Cuáles son las **FASES** que componen el Algoritmo de Simulated Annealing?

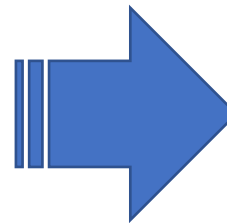
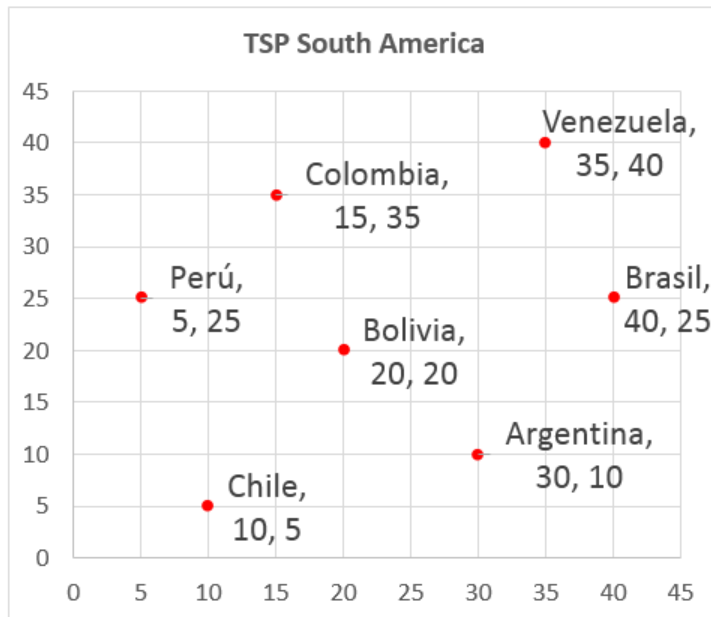


## 4. Ejemplos aplicando el Algoritmo Simulated Annealing

### EJEMPLO #1: el problema del Agente Viajero (TSP)

**Descripción del problema:** TSP por sus siglas en inglés (Travelman Salesman Problem), es un problema de optimización que consiste en hallar la ruta mínima de un recorrido de  $n$  ciudades, saliendo desde la ciudad de origen o nodo cero, recorriendo las  $n$  ciudades y retornando al punto o ciudad de origen

**Solución:** a través de Algoritmo de Recocido Simulado



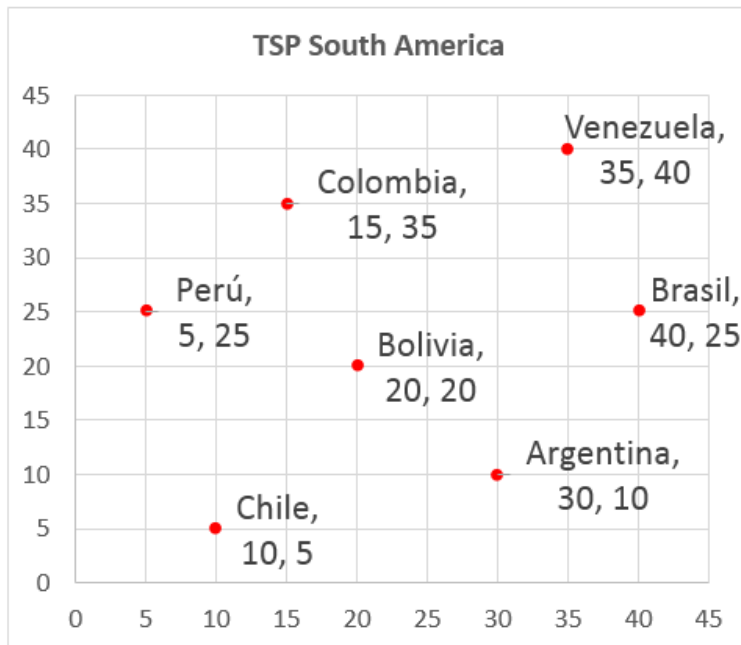
#### Pasos a seguir:

1. Inicializar Temperatura
2. Establecer Tour Aleatorio
3. Inicializar Tours
4. Elegir dos nodos
5. Establecer Nuevo Tour
6. Energía
7. Función de Aceptación
8. El mejor
9. Actualizar temperatura

## 4. Ejemplos aplicando el Algoritmo Simulated Annealing

**EJEMPLO #1:** el problema del Agente Viajero (TSP)

**Paso #1:** Inicializar temperatura



Temperatura inicial  
**Temp = 1000**

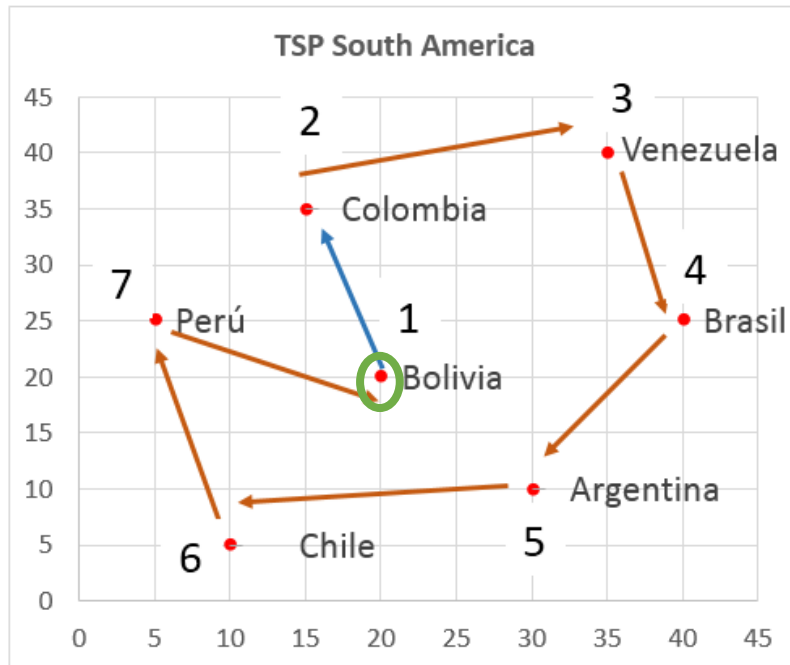
**Pasos a seguir:**

- 1. Inicializar Temperatura**
2. Establecer Tour Aleatorio
3. Inicializar Tours
4. Elegir dos nodos
5. Establecer Nuevo Tour
6. Energía
7. Función de Aceptación
8. El mejor
9. Actualizar temperatura

## 4. Ejemplos aplicando el Algoritmo Simulated Annealing

**EJEMPLO #1** el problema del Agente Viajero (TSP)

**Paso #2:** Establecer Tour aleatorio



Temperatura inicial  
**Temp = 1000**

**Tour aleatorio**

1(Bolivia)-2(Colombia)-3(Venezuela)-4(Brasil)-5(Argentina)-6(Chile)-7(Perú)

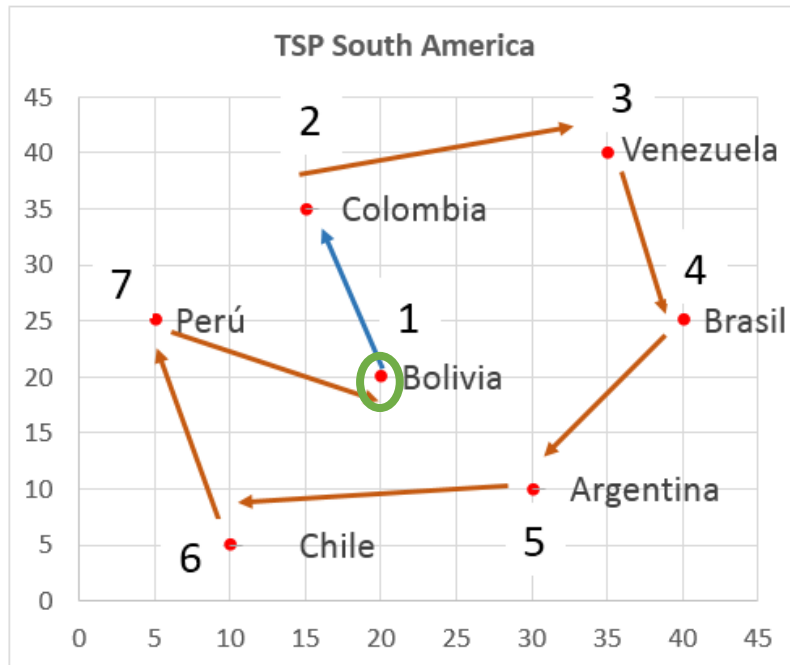
**Pasos a seguir:**

1. Inicializar Temperatura
2. **Establecer Tour Aleatorio**
3. Inicializar Tours
4. Elegir dos nodos
5. Establecer Nuevo Tour
6. Energía
7. Función de Aceptación
8. El mejor
9. Actualizar temperatura

## 4. Ejemplos aplicando el Algoritmo Simulated Annealing

**EJEMPLO #1** el problema del Agente Viajero (TSP)

**Paso #3:** Inicializar Tours -> actual, mejor y nuevo



Temperatura inicial  
**Temp = 1000**

**Tour aleatorio**

1(Bolivia)-2(Colombia)-3(Venezuela)-4(Brasil)-5(Argentina)-6(Chile)-7(Perú)

**TActual** = 1-2-3-4-5-6-7

**TMejor** = 1-2-3-4-5-6-7

**TNuevo** = 1-2-3-4-5-6-7

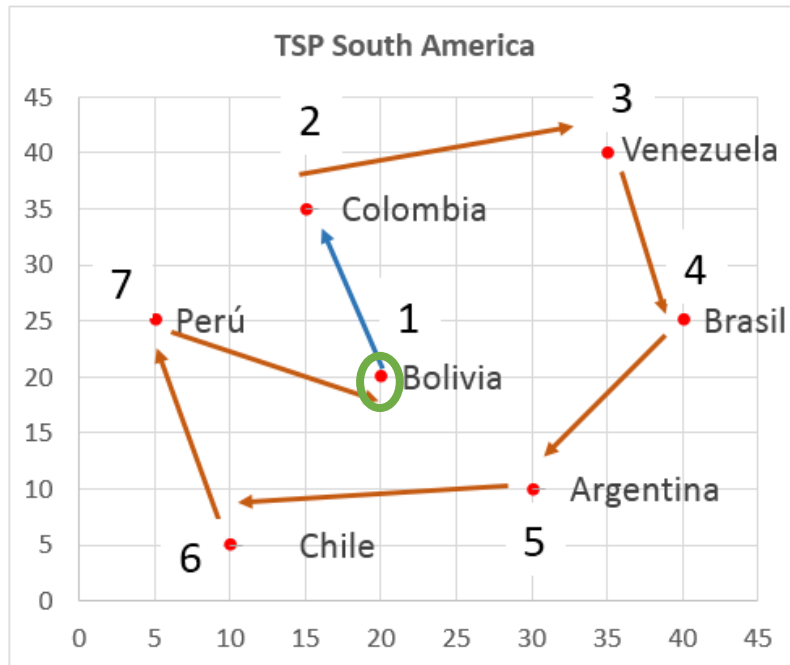
**Pasos a seguir:**

1. Inicializar Temperatura
2. Establecer Tour Aleatorio
- 3. Inicializar Tours**
4. Elegir dos nodos
5. Establecer Nuevo Tour
6. Energía
7. Función de Aceptación
8. El mejor
9. Actualizar temperatura

## 4. Ejemplos aplicando el Algoritmo Simulated Annealing

**EJEMPLO #1** el problema del Agente Viajero (TSP)

**Paso #4:** Elegir dos nodos (estados)



Temperatura inicial  
**Temp = 1000**

**Tour aleatorio**

1(Bolivia)-2(Colombia)-3(Venezuela)-4(Brasil)-5(Argentina)-6(Chile)-7(Perú)

**TActual** = 1-2-3-4-5-6-7

**TMejor** = 1-2-3-4-5-6-7

**TNuevo** = 1-2-3-4-5-6-7

**4 = Brasil**

**6 = Chile**

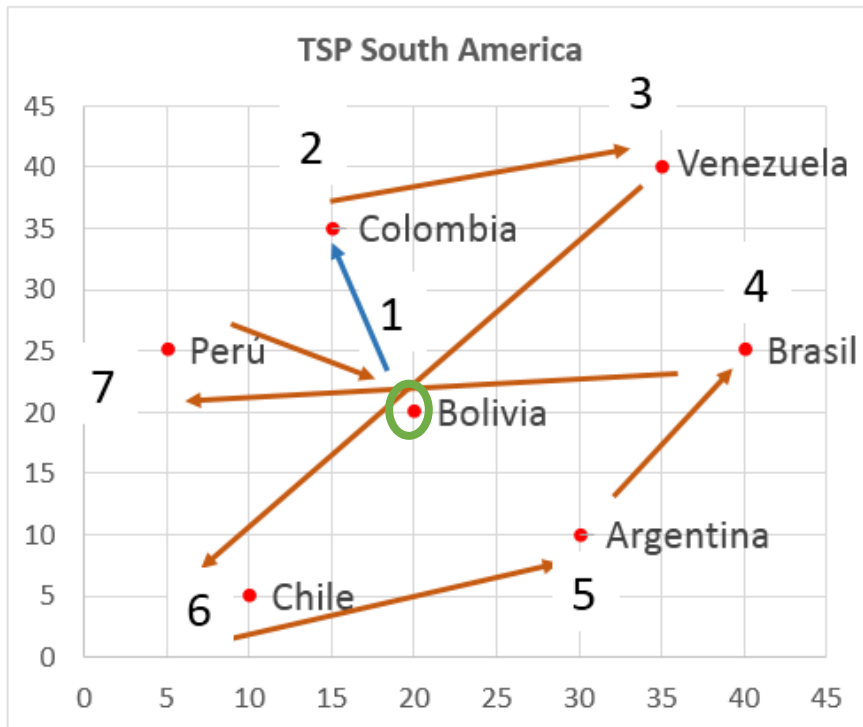
**Pasos a seguir:**

1. Inicializar Temperatura
2. Establecer Tour Aleatorio
3. Inicializar Tours
- 4. Elegir dos nodos**
5. Establecer Nuevo Tour
6. Energía
7. Función de Aceptación
8. El mejor
9. Actualizar temperatura

## 4. Ejemplos aplicando el Algoritmo Simulated Annealing

### EJEMPLO #1 el problema del Agente Viajero (TSP)

**Paso #5:** Alternamos el tour con el nuevo orden (intercambiamos el orden de visita de los nodos elegidos).



Temperatura inicial  
**Temp = 1000**

#### Tour aleatorio

1(Bolivia)-2(Colombia)-3(Venezuela)-4(Brasil)-5(Argentina)-6(Chile)-7(Perú)

TActual = 1-2-3-4-5-6-7

TMejor = 1-2-3-4-5-6-7

T~~Actual~~ivo = 1-2-3-4-5-6-7

TNuevo = 1-2-3-6-5-4-7

#### Pasos a seguir:

1. Inicializar Temperatura
2. Establecer Tour Aleatorio
3. Inicializar Tours
4. Elegir dos nodos
- 5. Establecer Nuevo Tour**
6. Energía
7. Función de Aceptación
8. El mejor
9. Actualizar temperatura

**4 = Brasil**

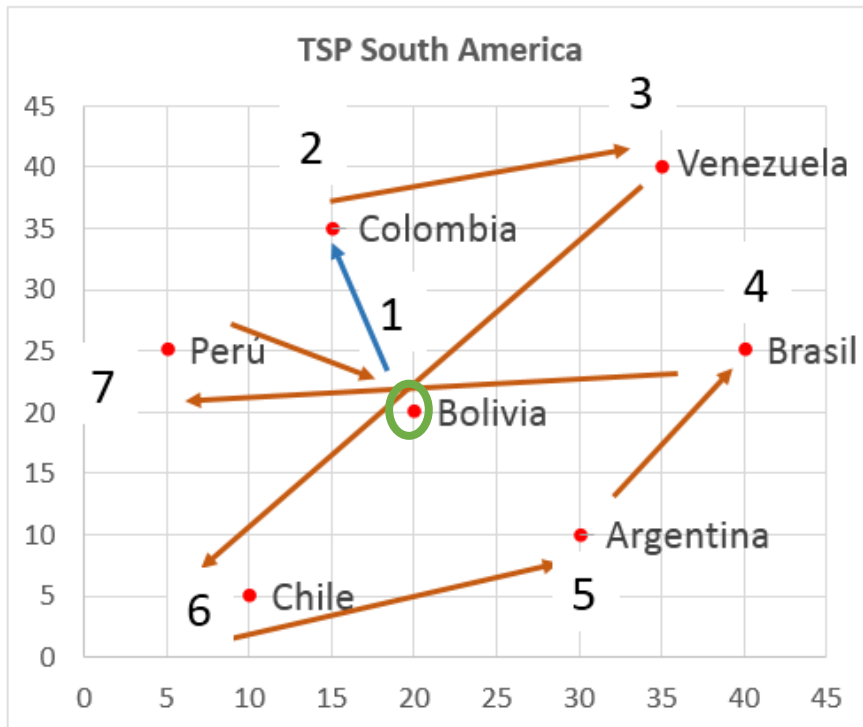
**6 = Chile**

Intercambiamos 4 por 6 y viceversa en el nuevo tour (TNuevo)

## 4. Ejemplos aplicando el Algoritmo Simulated Annealing

### EJEMPLO #1 el problema del Agente Viajero (TSP)

**Paso #6:** Establecemos un valor para la energía.



Temperatura inicial  
**Temp = 1000**

#### Tour aleatorio

1(Bolivia)-2(Colombia)-3(Venezuela)-4(Brasil)-5(Argentina)-6(Chile)-7(Perú)

TActual = 1-2-3-4-5-6-7

TMejor = 1-2-3-4-5-6-7

TNuevo = 1-2-3-6-5-4-7

#### Pasos a seguir:

1. Inicializar Temperatura
2. Establecer Tour Aleatorio
3. Inicializar Tours
4. Elegir dos nodos
5. Establecer Nuevo Tour
- 6. Energía**
7. Función de Aceptación
8. El mejor
9. Actualizar temperatura

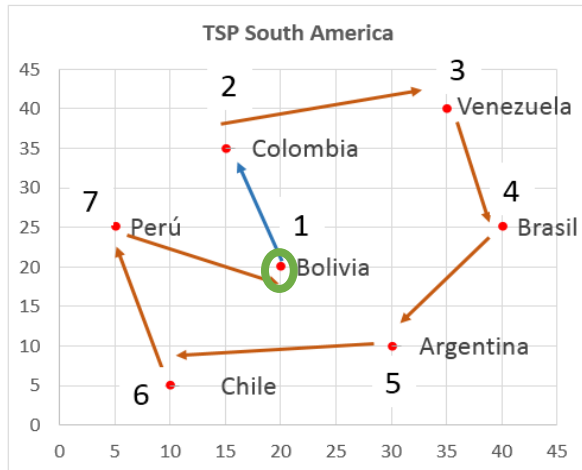
Energía= ????



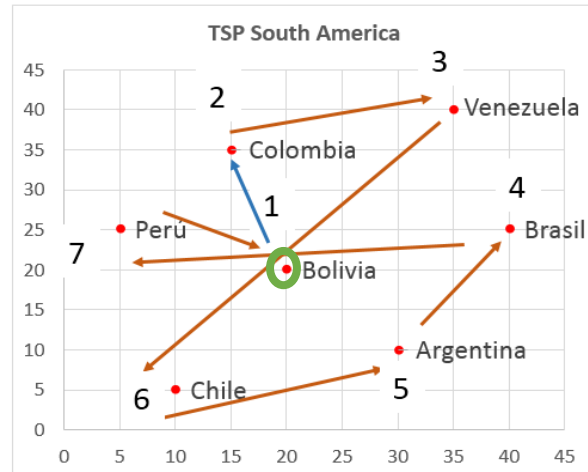
## 4. Ejemplos aplicando el Algoritmo Simulated Annealing

### EJEMPLO #1 el problema del Agente Viajero (TSP)

**Paso #6:** Establecemos un valor para la energía. La energía depende de la distancia entre dos nodos.



Energía Actual



Energía Nueva

Distancia entre dos nodos (países)

X\Y	Perú	Chile	Argentina	Brasil	Venezuela	Colom	Bolivia
Perú	0.00	20.62	29.15	35.00	33.54	14.14	15.81
Chile	20.62	0.00	20.62	36.06	43.01	30.41	18.03
Argentina	29.15	20.62	0.00	18.03	30.41	29.15	14.14
Brasil	35.00	36.06	18.03	0.00	15.81	26.93	20.62
Venezuela	33.54	43.01	30.41	15.81	0.00	20.62	25.00
Colombia	14.14	30.41	29.15	26.93	20.62	0.00	15.81
Bolivia	15.81	18.03	14.14	20.62	25.00	15.81	0.00

$$E_{\text{Actual}} = 15.81 + 20.62 + 15.81 + 18.03 + 20.62 + 20.62 + 15.81 = 127.32$$

$$E_{\text{Nueva}} = 15.81 + 20.62 + 43.01 + 20.62 + 18.03 + 35.00 + 15.81 = 168.90$$

## 4. Ejemplos aplicando el Algoritmo Simulated Annealing

### EJEMPLO #1 el problema del Agente Viajero (TSP)

**Paso #7:** Reemplazamos la energía actual (EActual) y nueva (ENueva) en la función de aceptación para obtener el Tour actual (TActual)

**Temp** = 1000  
**EActual** = 127.32  
**ENueva** = 168.90

#### Pasos a seguir:

1. Inicializar Temperatura
2. Establecer Tour Aleatorio
3. Inicializar Tours
4. Elegir dos nodos
5. Establecer Nuevo Tour
6. Energía
- 7. Función de Aceptación**
8. El mejor
9. Actualizar temperatura

#### FUNCIÓN DE ACEPTACIÓN

Si  $ENueva < EActual$

Prob = 1

sino

Prob =  $\exp((EActual - ENueva) / Temp)$

Si Prob > Random(0,1)

TActual = TNuevo



#### REEMPLAZANDO

Si  $168.90 < 127.32$

Prob = 1

sino

Prob =  $\exp((127.32 - 168.90) / 1000) = 0.96$

Si Prob > 0.45

TActual = TNuevo //En este caso TActual = 1-2-3-6-5-4-7

## 4. Ejemplos aplicando el Algoritmo Simulated Annealing

### EJEMPLO #1 el problema del Agente Viajero (TSP)

**Paso #8 y #9:** De ser el caso, actualizamos el mejor Tour (TMejor) y reducimos la temperatura (Temp).

**La mejor solución:**

Si Distancia TActual < Distancia TMejor  
TMejor = TActual



**REEMPLAZANDO**

Si  $168.90 < 127.32$   
TMejor = TActual //No cambia TMejor

TActual = 1-2-3-6-5-4-7  
TMejor = 1-2-3-4-5-6-7  
TNuevo = 1-2-3-6-5-4-7

Se actualiza la temperatura  
 $Temp = (1 - V_{\text{Enfriamiento}}) * Temp$   
 $Temp = (1 - 0.003) * 1000$   
 $Temp = 997$

Donde:  
 $V_{\text{Enfriamiento}} = 0.003$

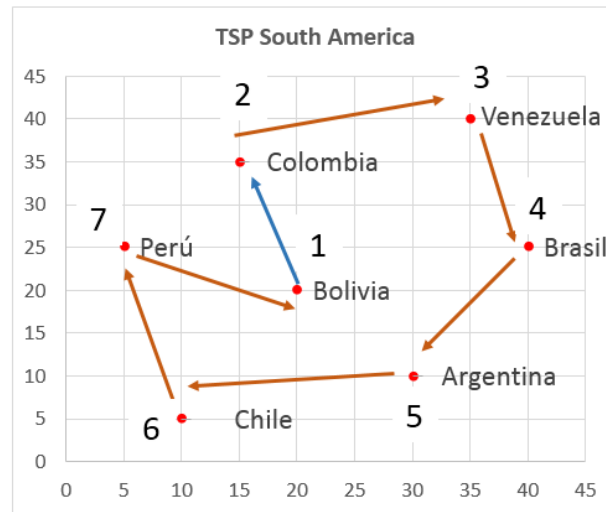
**Pasos a seguir:**

1. Inicializar Temperatura
2. Establecer Tour Aleatorio
3. Inicializar Tours
4. Elegir dos nodos
5. Establecer Nuevo Tour
6. Energía
7. Función de Aceptación
- 8. El mejor**
- 9. Actualizar temperatura**

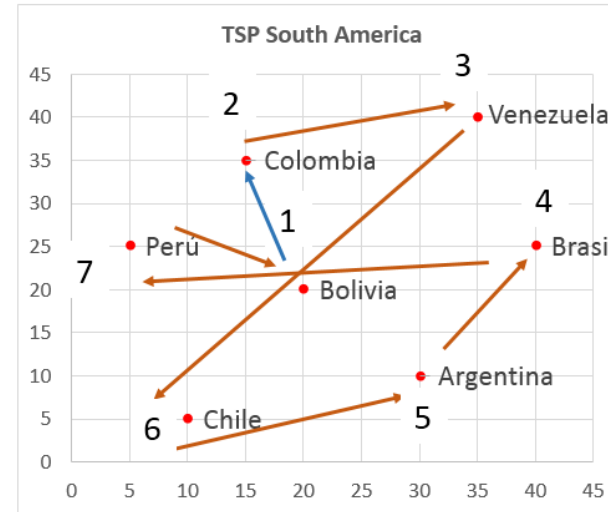
## 4. Ejemplos aplicando el Algoritmo Simulated Annealing

**EJEMPLO #1:** el problema del Agente Viajero (TSP)

**Resultados de esta primera iteración:**



TActual = 1-2-3-6-5-4-7  
TMejor = 1-2-3-4-5-6-7  
TNuevo = 1-2-3-6-5-4-7

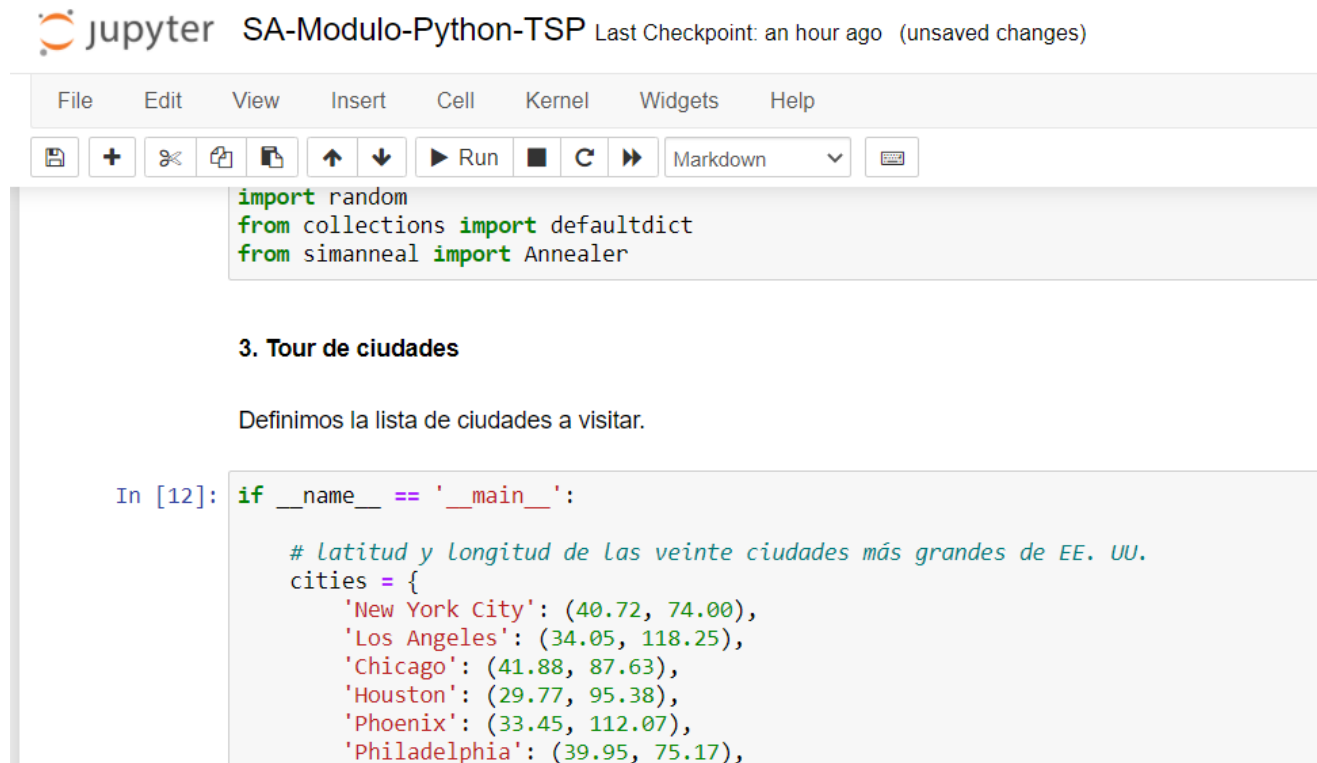


Distancia Actual = EActual = 168.90  
Distancia Mejor = 127.32  
Distancia Nueva = ENueva = 168.90  
Temperatura = 997

# 4. Ejemplos aplicando el Algoritmo Simulated Annealing

## EJEMPLO# 2:

Solución al problema del vendedor ambulante (TSP) utilizando el módulo **simanneal** – en Python



The screenshot shows a Jupyter Notebook titled "SA-Modulo-Python-TSP" with a status bar indicating "Last Checkpoint: an hour ago (unsaved changes)". The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for saving, adding cells, undo, redo, and running code. The notebook content is as follows:

```
import random
from collections import defaultdict
from simanneal import Annealer
```

### 3. Tour de ciudades

Definimos la lista de ciudades a visitar.

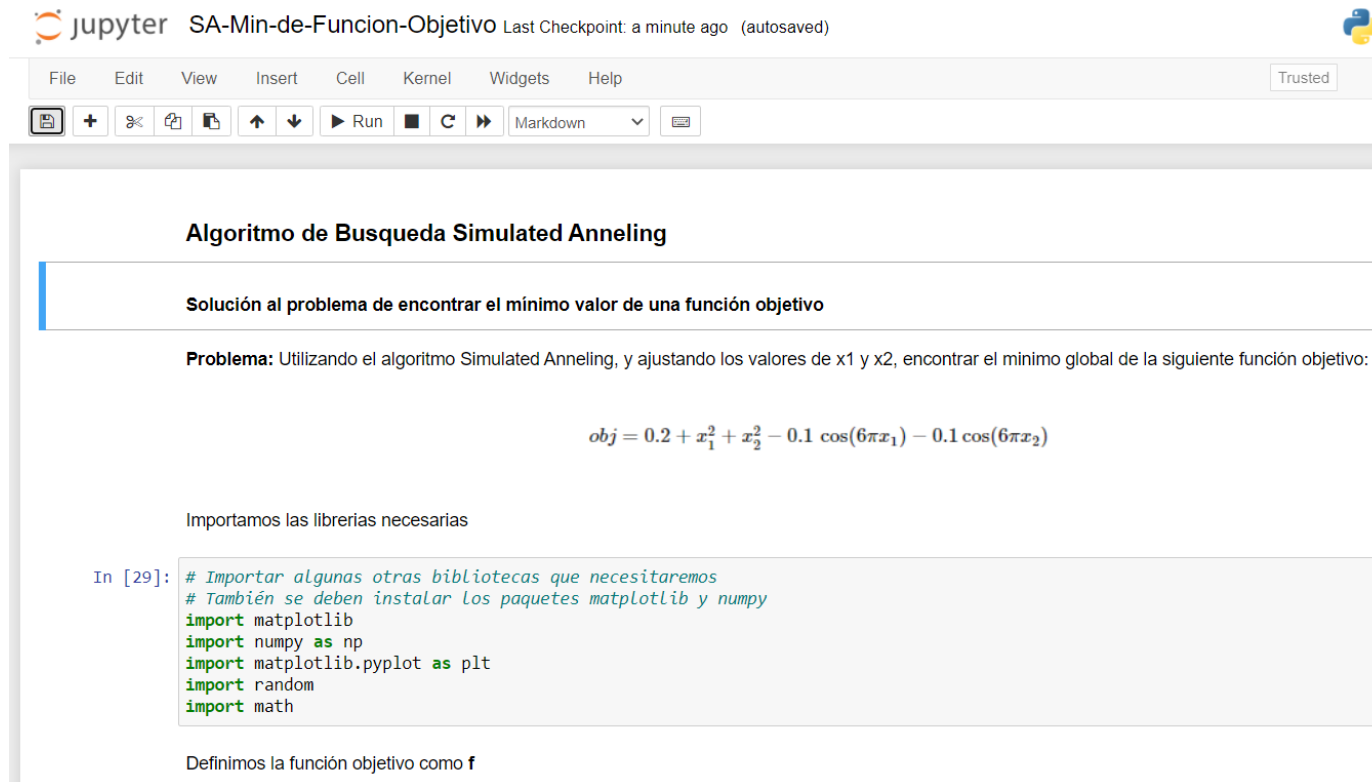
```
In [12]: if __name__ == '__main__':

# Latitud y Longitud de las veinte ciudades más grandes de EE. UU.
cities = {
    'New York City': (40.72, 74.00),
    'Los Angeles': (34.05, 118.25),
    'Chicago': (41.88, 87.63),
    'Houston': (29.77, 95.38),
    'Phoenix': (33.45, 112.07),
    'Philadelphia': (39.95, 75.17),
```

# 4. Ejemplos aplicando el Algoritmo Simulated Annealing

## EJEMPLO# 3:

Solución al problema de encontrar el mínimo valor de una función objetivo - en Python



The screenshot shows a Jupyter Notebook interface with the title "SA-Min-de-Funcion-Objetivo" and a status bar indicating "Last Checkpoint: a minute ago (autosaved)". The notebook has a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for saving, adding cells, undo, redo, and running code. The main content area is divided into sections:

- Algoritmo de Búsqueda Simulated Annealing**
- Solución al problema de encontrar el mínimo valor de una función objetivo**
- Problema:** Utilizando el algoritmo Simulated Annealing, y ajustando los valores de  $x_1$  y  $x_2$ , encontrar el mínimo global de la siguiente función objetivo:
$$obj = 0.2 + x_1^2 + x_2^2 - 0.1 \cos(6\pi x_1) - 0.1 \cos(6\pi x_2)$$
- Importamos las librerías necesarias
- In [29]:**

```
# Importar algunas otras bibliotecas que necesitaremos
# También se deben instalar los paquetes matplotlib y numpy
import matplotlib
import numpy as np
import matplotlib.pyplot as plt
import random
import math
```
- Definimos la función objetivo como **f**

# PREGUNTAS

Dudas y opiniones