

# Semillero de Programación

## Algoritmo de Bellman-Ford

Ana Echavarría    Juan Francisco Cardona

Universidad EAFIT

22 de marzo de 2013

# Contenido

- 1 Problemas semana anterior
  - Problema A - Non-Stop Travel
  - Problema B - Peter's Smokes
  - Problema C - Fire Station
- 2 Grafos con peso negativo
- 3 Algoritmo de Bellman-Ford
- 4 Tarea

# Contenido

- 1 Problemas semana anterior
  - Problema A - Non-Stop Travel
  - Problema B - Peter's Smokes
  - Problema C - Fire Station

# Problema A - Non-Stop Travel

# Implementación I

---

---

# Problema B - Peter's Smokes

# Implementación I

---

---

# Problema C - Fire Station



# Implementación I



# Contenido

## 2 Grafos con peso negativo

# Motivación

Sea el grafo  $G = (V, E)$  un grafo donde

- $V$  es un conjunto de moléculas químicas
- $(u, v) \in E$  si existe una reacción química que convierta la molécula  $u$  en la molécula  $v$ .
- $f(e)$  es la energía requerida para hacer la reacción  $e \in E$
- Si una reacción consume energía  $f(e) > 0$  y si produce energía  $f(e) < 0$ .

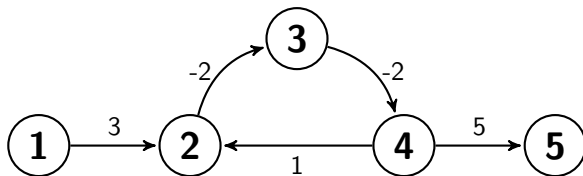
¿Qué hacer si se quiere hallar la mínima energía para convertir una molécula en otra?

# Grafos con pesos negativos

Algunas veces se necesita hallar el camino más corto desde una fuente a todos los demás nodos en un grafo que tiene pesos negativos.

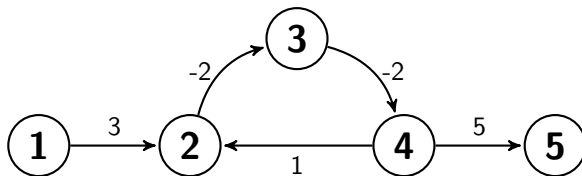
El algoritmo de Bellman-Ford encuentra esos caminos.

# Ciclos con peso negativo



¿Cuál es la distancia más corta del nodo 1 al nodo 5?

# Ciclos con peso negativo



¿Cuál es la distancia más corta del nodo 1 al nodo 5?

Se tiene el camino con peso  $-\infty$   
 $1 - (2 - 3 - 4) - (2 - 3 - 4) - \dots - (2 - 3 - 4) - 5.$

# Ciclos con peso negativo

- El problema del camino más corto está bien definido siempre y cuando no se tengan ciclos de peso negativo alcanzables desde la fuente.
- Si el grafo contiene ciclos de peso negativo alcanzables desde  $s$  el problema no está bien definido porque no hay ningún camino más corto saliendo desde la fuente.
- Si se tiene un camino “más corto” desde  $s$ , este se puede hacer aún más corto si se hace una vuelta más al ciclo de peso negativo.

# Contenido

## 3 Algoritmo de Bellman-Ford



# Single-Source Shortest-Paths con pesos negativos

## Entrada

- Un grafo  $G = (V, E)$
- Una función de pesos  $w : E \rightarrow \mathbb{R}$  (posiblemente  $w < 0$ )
- Un nodo  $s \in V$

## Objetivo

- Para todo  $v \in V$  hallar el camino más corto de  $s$  a  $v$

o

- Decir que  $G$  tiene un ciclo de peso negativo y que el problema no está bien definido.

# Pregunta

## Pregunta

Sea  $n$  el número de nodos y  $m$  el número de aristas de un grafo  $G$ . Si  $G$  no contiene ciclos de peso negativo, ¿cuál es el número máximo de aristas que puede contener un camino más corto entre dos nodos  $s$  y  $v$ ?

# Pregunta

## Pregunta

Sea  $n$  el número de nodos y  $m$  el número de aristas de un grafo  $G$ . Si  $G$  no contiene ciclos de peso negativo, ¿cuál es el número máximo de aristas que puede contener un camino más corto entre dos nodos  $s$  y  $v$ ?

## Respuesta

Un camino más corto en un grafo sin ciclos de peso negativo tiene máximo  $n-1$  aristas.

Si tuviera más es porque visita algún nodo dos veces lo que implica que hay un ciclo.

Como el ciclo no tiene peso negativo, se puede eliminar y reducir la longitud del camino.

# Idea principal

Hallar el camino más corto de  $s$  a  $v$  que usa máximo  $i$  aristas.

## ¿Cómo hacerlo?

Sea  $L_{i,v}$  la longitud del camino más corto de  $s$  a  $v$  con máximo  $i$  aristas (se permiten ciclos).

$$L_{0,v} = \begin{cases} 0 & \text{si } v = s \\ +\infty & \text{si } v \neq s \end{cases}$$

Ahora  $\forall v \in V$

$$L_{i,v} = \min \left\{ \begin{array}{l} L_{i-1,v} \\ \min_{(u,v) \in E} \{L_{i-1,u} + w_{u,v}\} \end{array} \right\}$$

# Algoritmo cuando no hay ciclos de peso negativo

- Asumamos que no hay ciclos de peso negativo.
- Ya vimos que el camino más corto tiene máximo  $n - 1$  aristas
- Para hallar la solución al problema hay que computar  $L_{i,v}$  para  $i = 0, 1, \dots, n - 1$  y todo  $v \in V$

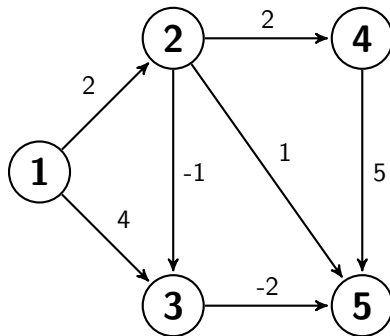
# Algoritmo cuando no hay ciclos de peso negativo

---

```
1  Crear matriz L[MAXN][MAXN]  // L[i][u]
2  Hacer L[0][s] = 0 y L[0][u] = INF para u != s
3  Para i = 1 hasta n-1
4      // El camino ms corto con i-1 aristas
5      Para u = 0 hasta n-1
6          L[i][u] = L[i-1][u]
7
8      // Mirar todas las aristas
9      Para u = 0 hasta n-1
10         Para k = 0 hasta g[u].size() - 1
11             v = g[u][k].first  // El nodo
12             w = g[u][k].second // El peso
13             // Mirando la arista (u, v) de peso w
14             L[i][v] = min(L[i][v], L[i-1][u] + w)
```

---

# Ejemplo



# Complejidad

## Complejidad

¿Cuál de las siguientes es la mejor aproximación a la complejidad del algoritmo cuando no hay ciclos de peso negativo?

- a  $O(n^2)$
- b  $O(n \cdot m)$
- c  $O(n^3)$
- d  $O(m^2)$



# Complejidad

## Complejidad

¿Cuál de las siguientes es la mejor aproximación a la complejidad del algoritmo cuando no hay ciclos de peso negativo?

- a  $O(n^2)$
- b  $O(n \cdot m)$
- c  $O(n^3)$
- d  $O(m^2)$



# Contenido

## 4 Tarea

# Tarea

## Tarea

Resolver los problemas de