Le problème du plus court chemin

Extrait du cours de MANIER H. (Logistique, UTBM)
Et cours CARLIER J. (Recherche Opérationnelle et Optimisation Combinatoire, UTSEUS)

Exemples d'applications

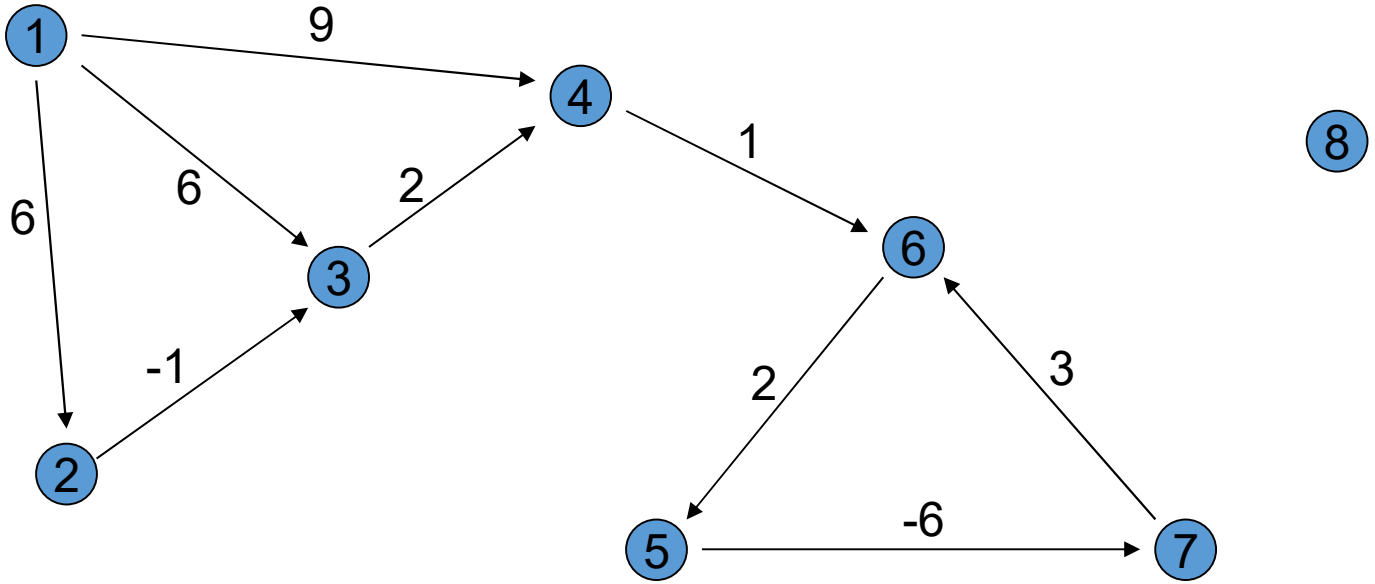
- Rechercher un meilleur itinéraire (le plus court, le moins long, le plus beau, le plus intéressant, le plus sûr) : GPS, sites internet, etc.
- Quand remplacer sa voiture en fonction du prix de revente (qui diminue d'année en année) et des frais d'entretien (qui augmentent d'année en année) ?
- Comment optimiser ses placements financiers en fonction des rendements des différentes possibilités ?
- Trouver la suite d'actions pour réussir un jeu solitaire déterministe de stratégie.

Applications

- Sous-problème de nombreux problèmes d'optimisation.
- Applications dans les transports:
  - Tournées de véhicules;
  - Détermination du trajet le plus rapide ou le plus court.
- Théorie des jeux.
- Applications dans les réseaux télécom.
- *etc.*

Source : Geoconcept

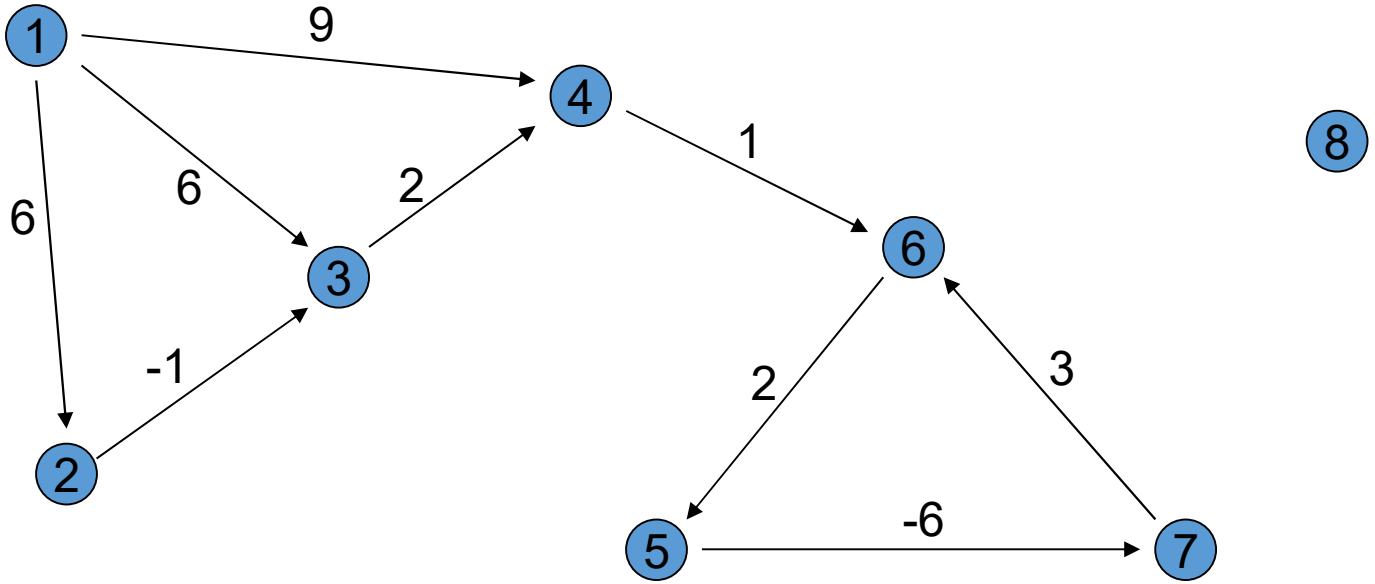
Existence d'un plus court chemin dans un graphe



- De 1 à 8 : pas de chemin

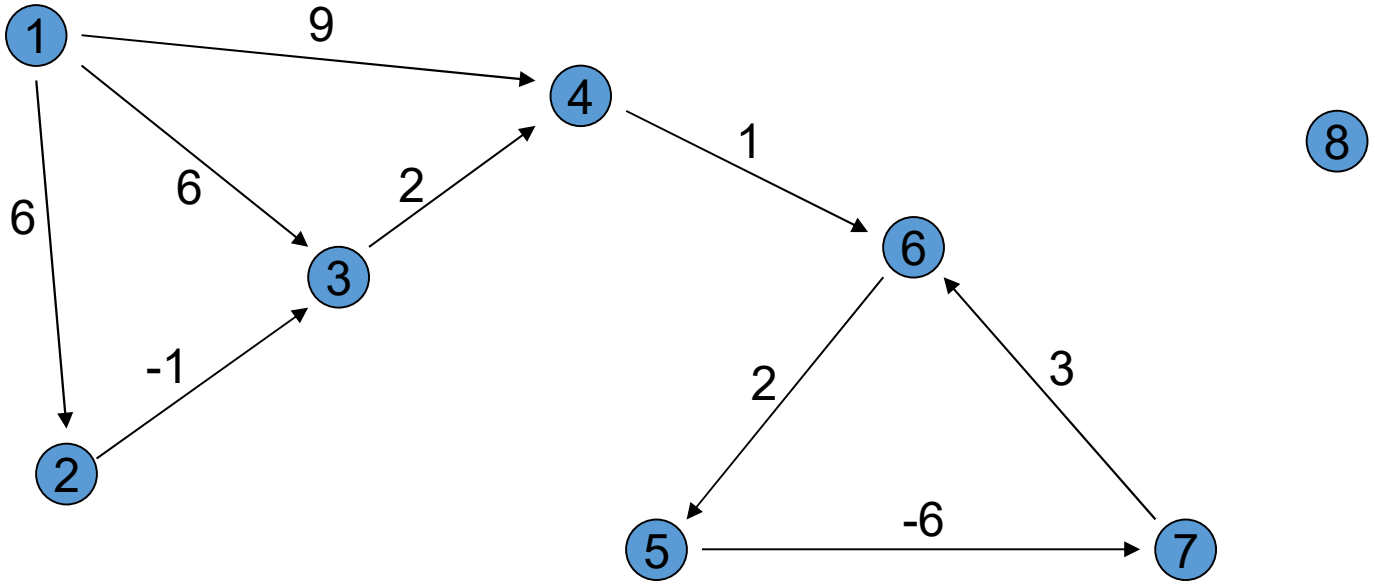
  De 7 à 1 : pas de plus court chemin

Existence d'un plus court chemin dans un graphe



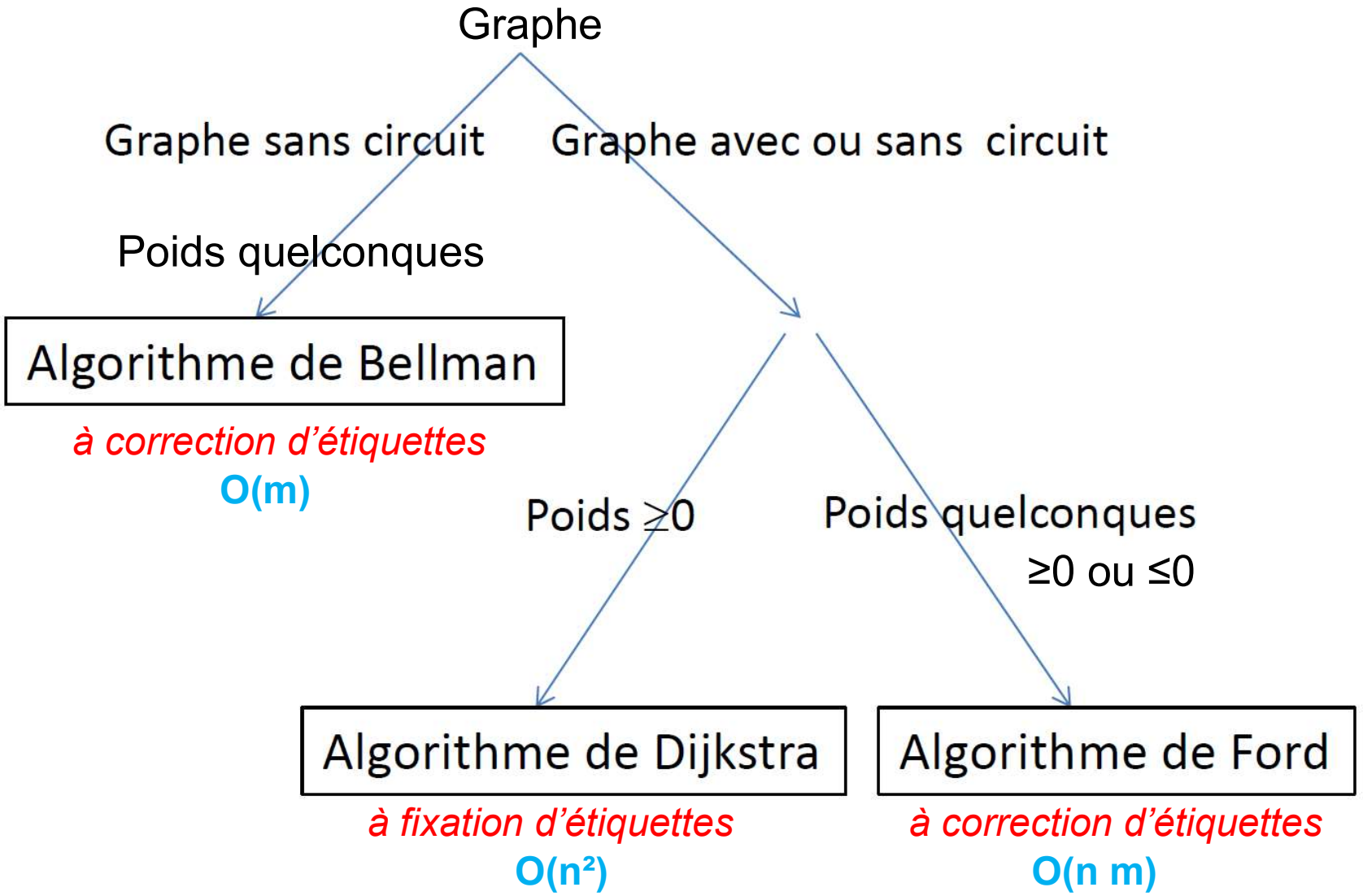
- De 1 à 4 : chemins : (1,4), (1,3,4), (1,2,3,4)
  plus court chemin : (1,2,3,4) *(longueur 7)*

Existence d'un plus court chemin dans un graphe



- De 3 à 5 : chemin (3,4,6,5), *longueur 5*
  chemin (3,4,6,5,7,6,5), *longueur 4*
  chemin (3,4,6,5,7,6,5,7,6,5), *longueur 3*
  …
  => Pas de plus court chemin

Algorithmes de cheminements



⇒ *Remarque: Une condition suffisante pour que, pour tout i, il existe un chemin de valeur minimale allant de x_0 à x_i est que le graphe G soit sans circuit de valeur strictement négative (ces circuits sont dit absorbants)*

Méthode de résolution du problème du plus court chemin dans un graphe

Introduction

- Plusieurs versions :
  - d'un nœud α vers un nœud β
  - de tous les nœuds vers un nœud β
  - d'un nœud α vers tous les nœuds
  - de tous les nœuds vers tous les nœuds
- Nous allons étudier le problème du plus court chemin d'un nœud α vers tous les autres.
- Souvent, on supposera que α=1.

Introduction

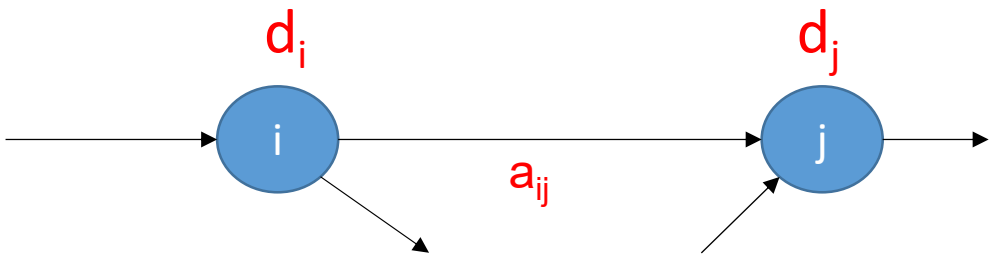
Idée générale :

- Parcourir le réseau à partir de l'origine
- Chaque étiquette est soit un scalaire soit $+\infty$.
- Appliquer et mettre à jour des **étiquettes** $(d_1,\ldots,d_N)$ à chaque nœud.

Conditions d'optimalité

- Soient $d_1, d_2, \ldots, d_N$ des scalaires tels que

$$d_j \leq d_i + a_{ij} \quad \forall (i,j) \in \mathcal{A}$$



- Soit P un chemin entre un nœud α et un nœud β.
- Si

$$d_j = d_i + a_{ij} \quad \forall (i,j) \in P$$

- Alors P est un plus court chemin entre α et β.

Principe (algorithme générique)

Idée :

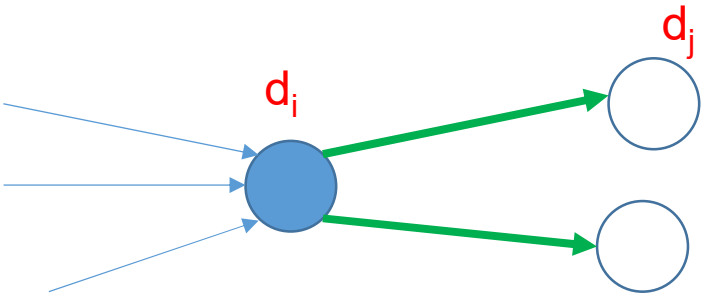
- On démarre avec un vecteur d'étiquettes $(d_1,\ldots,d_N)$
- On sélectionne un arc (i,j) qui viole les conditions d'optimalité, c-à-d tel que $d_j > d_i + a_{ij}$
- On met à jour l'étiquette de j

$$d_j \leftarrow d_i + a_{ij}$$

- Et ainsi de suite jusqu'à ce que tous les arcs vérifient la condition.

Principe (algorithme générique)

- A tout moment, l'étiquette d'un nœud i peut être interprétée comme la longueur d'un chemin reliant α à i.
- Si $d_j > d_i + a_{ij}$, cela signifie que le chemin arrivant en j à partir de i est plus court que le chemin actuel reliant α à j.
- Il est plus facile d'effectuer le traitement nœud par nœud.
- Pour chaque nœud considéré , on traitera tous ses arcs sortant
- V= liste des nœuds à traiter.



Question:
quel nœud considérer parmi les candidats pour l'itération suivante?

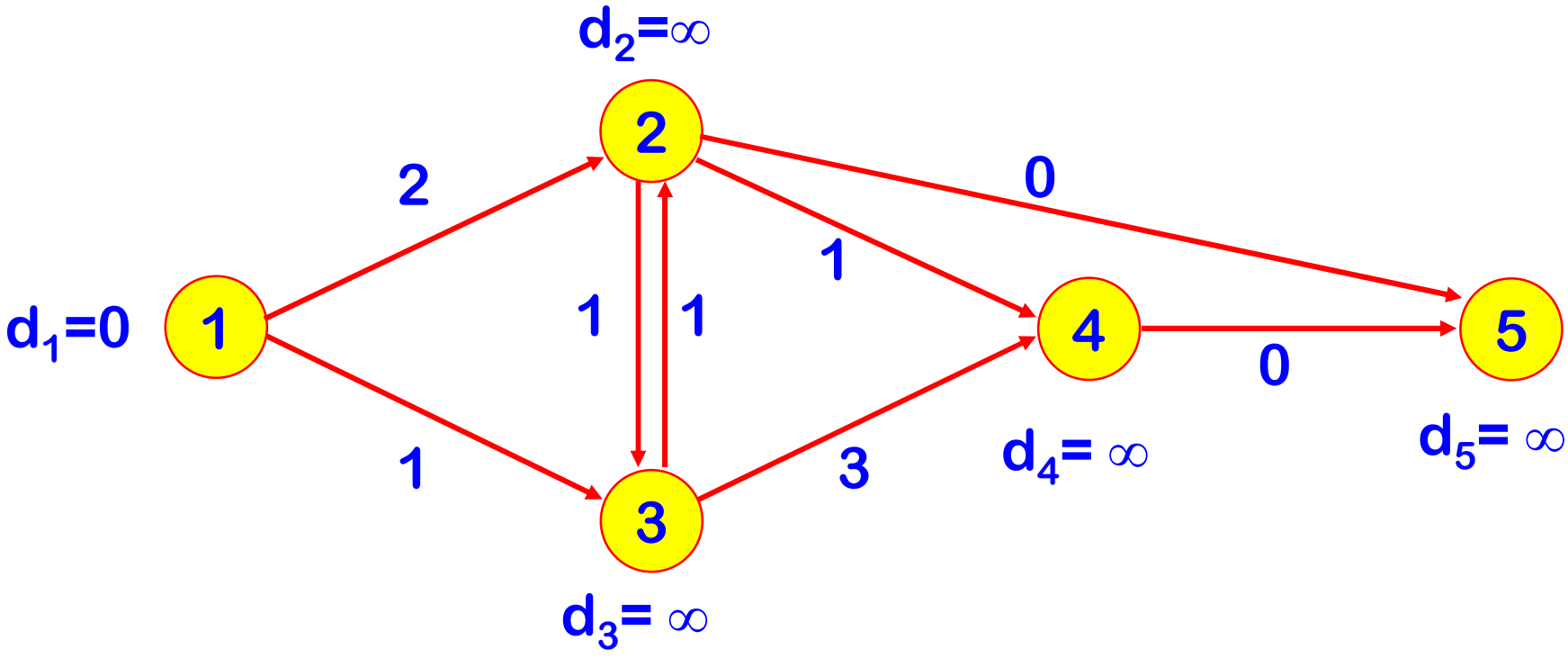
Algorithme de Dijkstra

Algorithme :

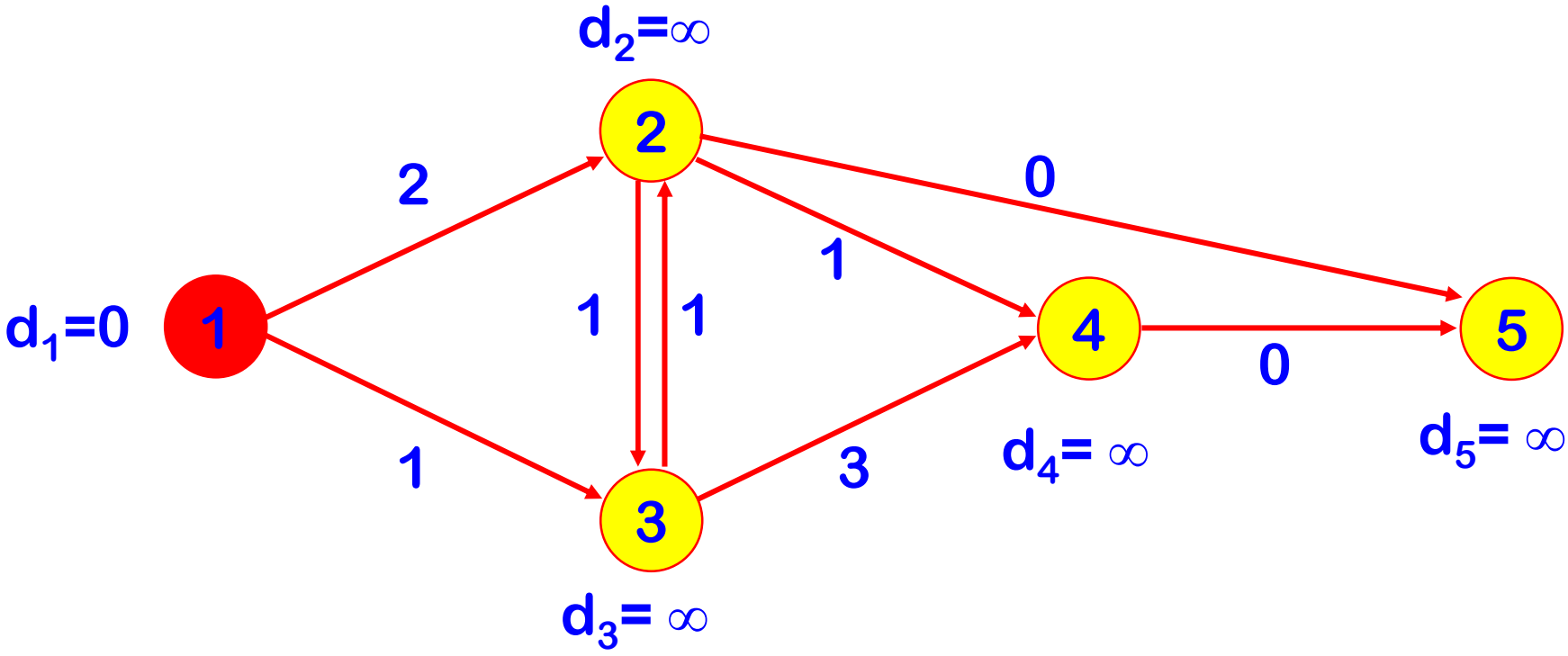
- Initialisation :
  - $V=\{\alpha\}$
  - $d_\alpha = 0$, $d_i = \infty \ \forall i \neq \alpha$
- Itérations. Tant que $V \neq \emptyset$
  - Sélectionner un nœud i dans V tel que

$$d_i = \min_{j \in V} d_j$$

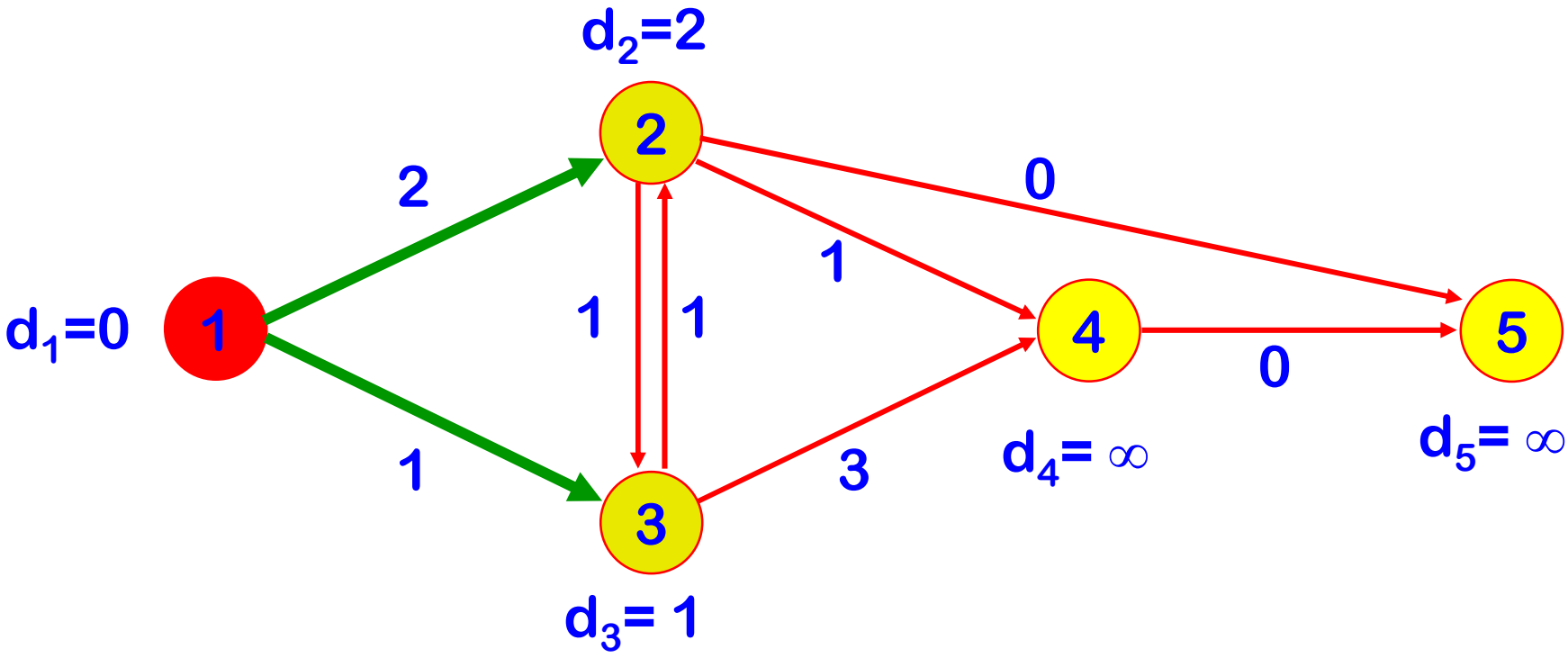
  - Retirer i de V
  - Pour chaque arc (i,j) sortant de i,
    - Si $d_j > d_i + a_{ij}$, alors
      - $d_j \leftarrow d_i + a_{ij}$
      - Ajouter j à V



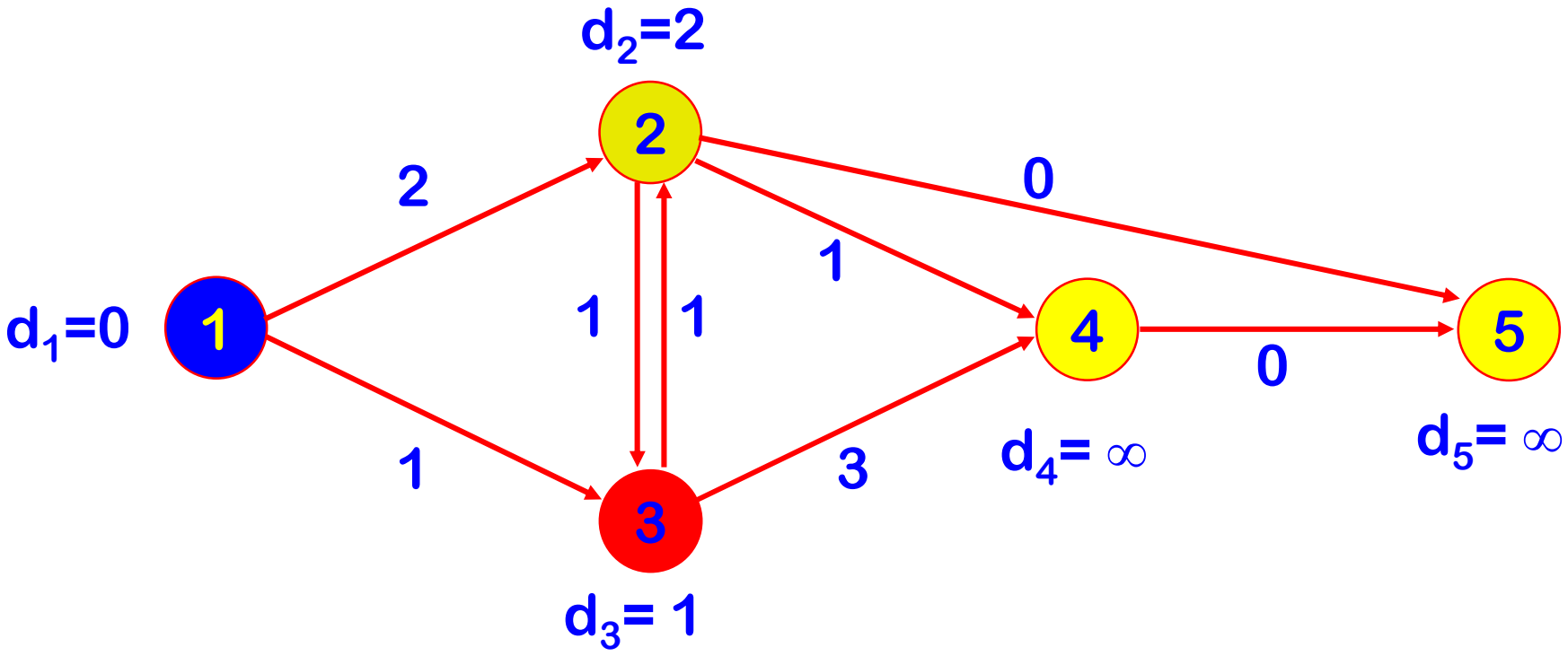
Iter	V	d_1	d_2	d_3	d_4	d_5	traiter
1	{1}	0	∞	∞	∞	∞	1



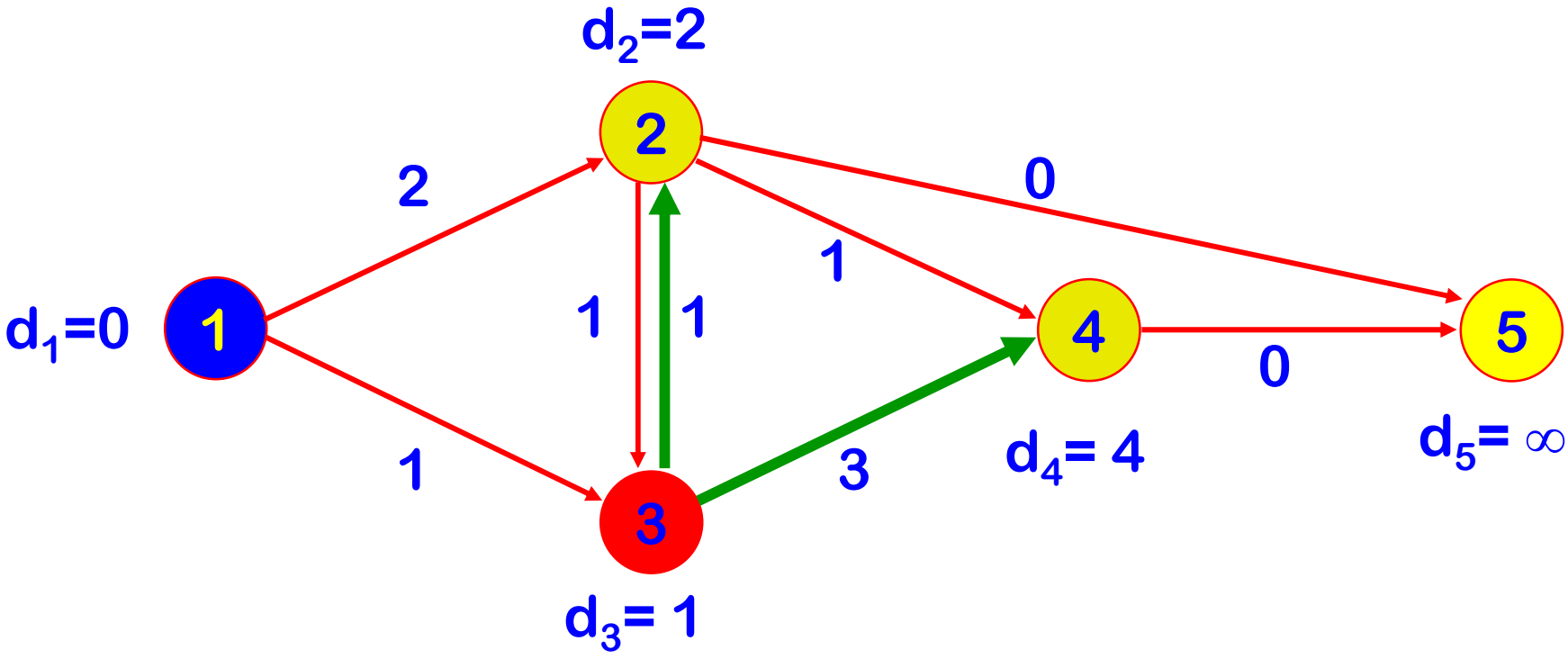
Iter	V	d_1	d_2	d_3	d_4	d_5	traiter
1	{1}	0	∞	∞	∞	∞	1



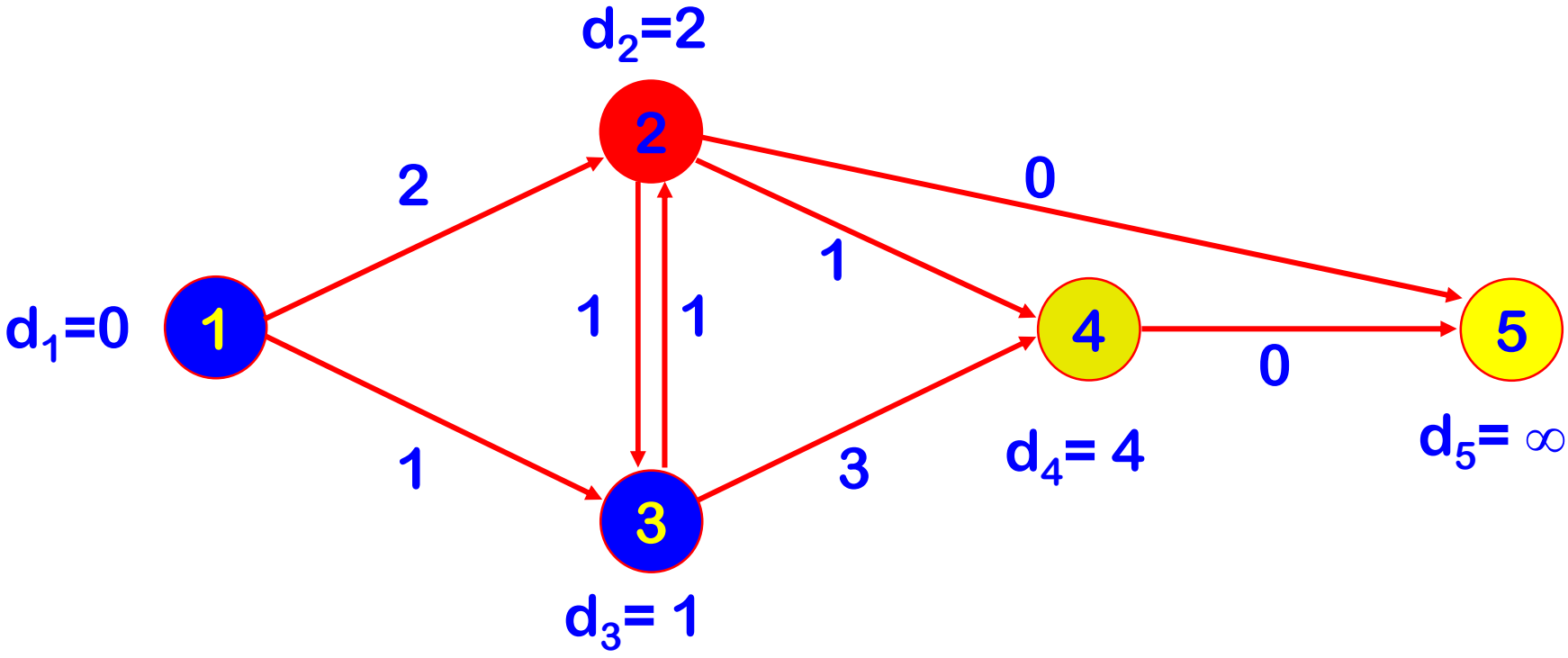
Iter	V	d_1	d_2	d_3	d_4	d_5	traiter
1	{1}	0	∞	∞	∞	∞	1
2	{2,3}	0	2	1	∞	∞	3



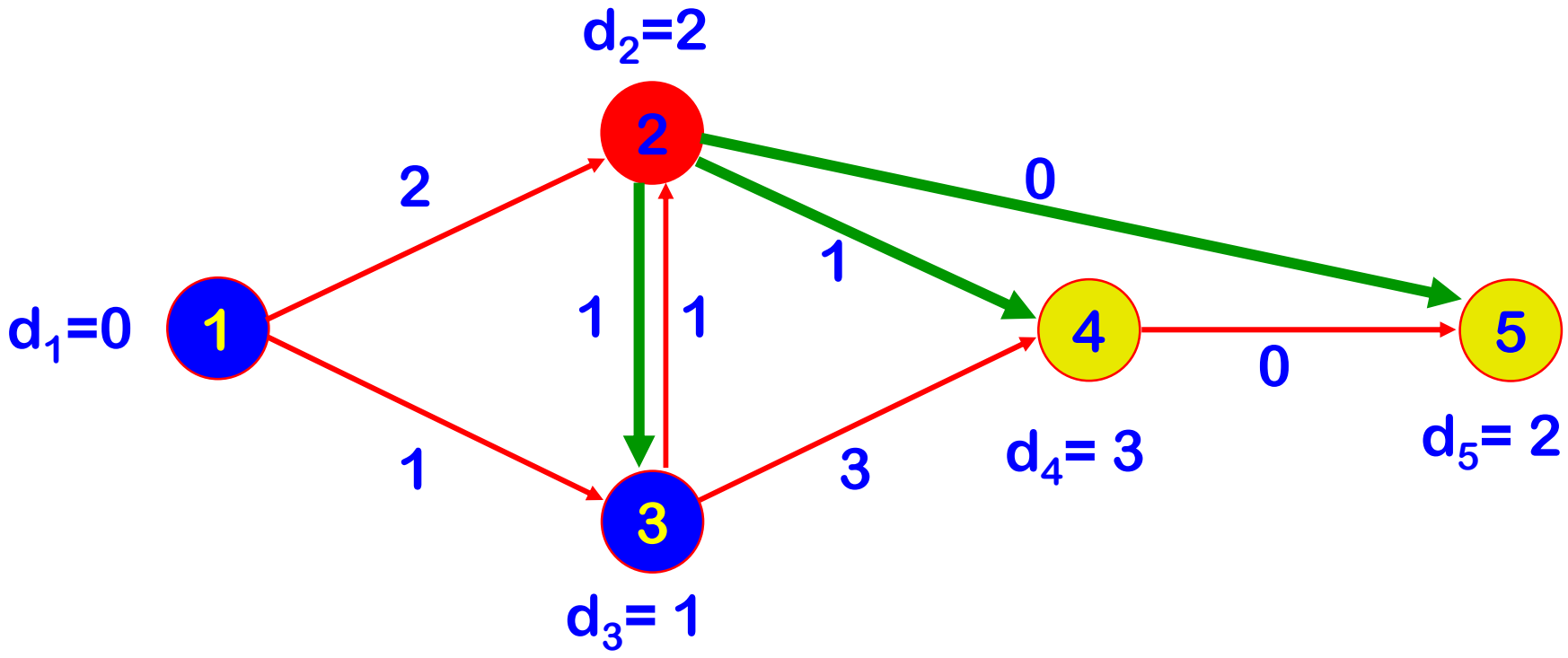
Iter	V	d_1	d_2	d_3	d_4	d_5	traiter
1	{1}	0	∞	∞	∞	∞	1
2	{2,3}	0	2	1	∞	∞	3



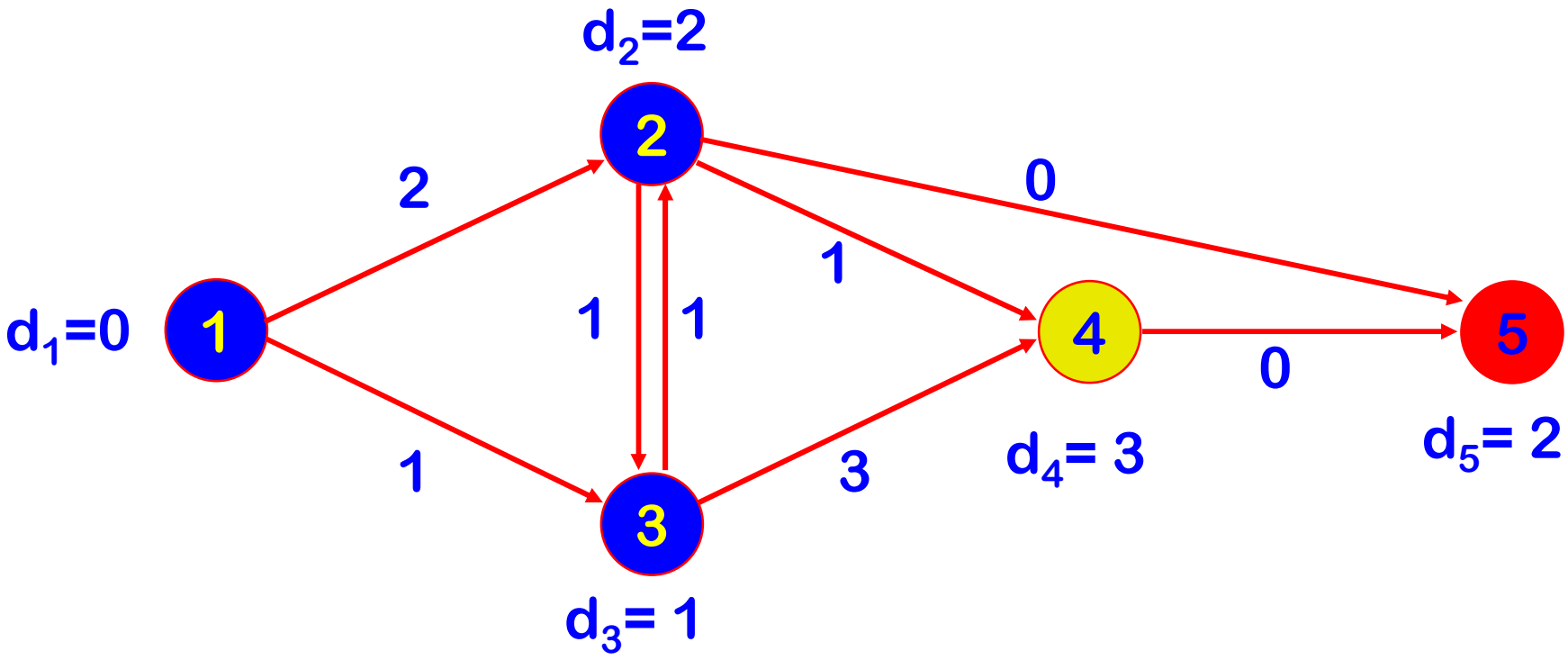
Iter	V	d_1	d_2	d_3	d_4	d_5	traiter
1	{1}	0	∞	∞	∞	∞	1
2	{2,3}	0	2	1	∞	∞	3
3	{2,4}	0	2	1	4	∞	2



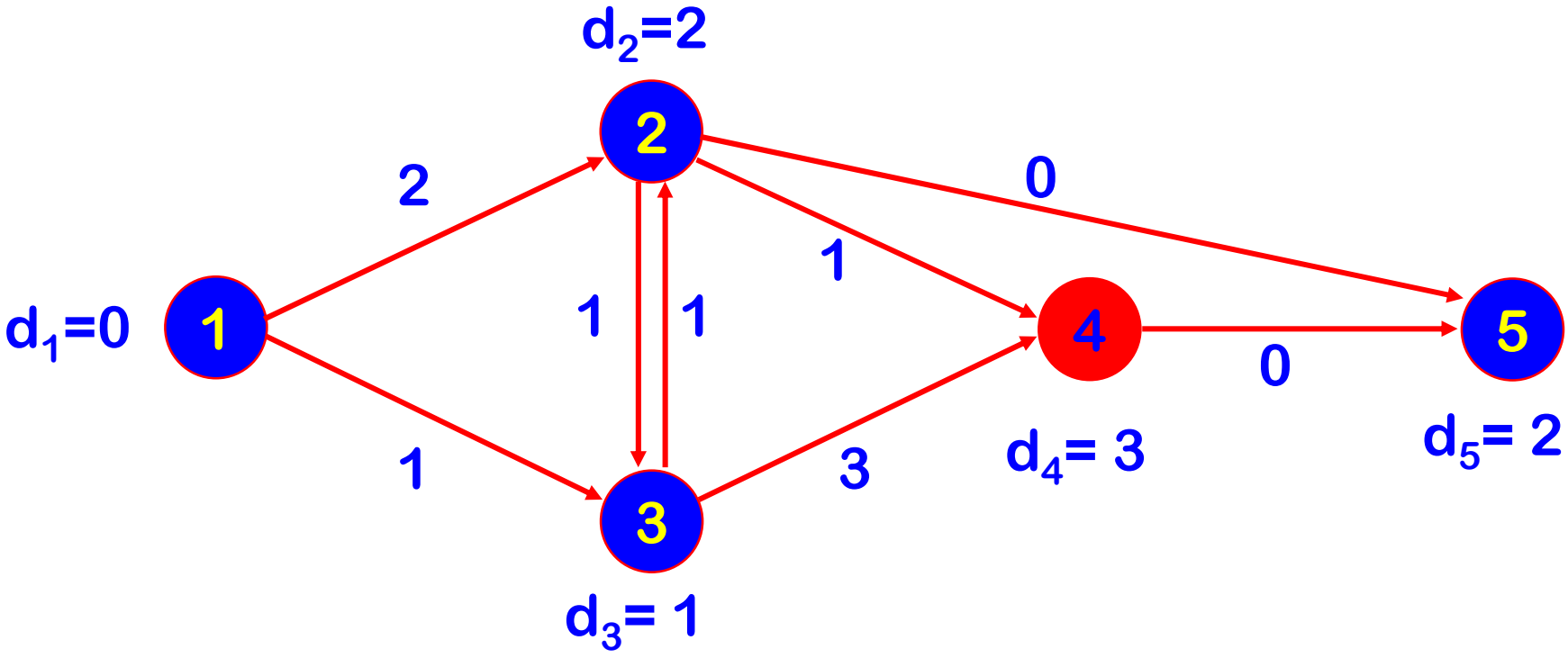
Iter	V	d_1	d_2	d_3	d_4	d_5	traiter
1	{1}	0	∞	∞	∞	∞	1
2	{2,3}	0	2	1	∞	∞	3
3	{2,4}	0	2	1	4	∞	2



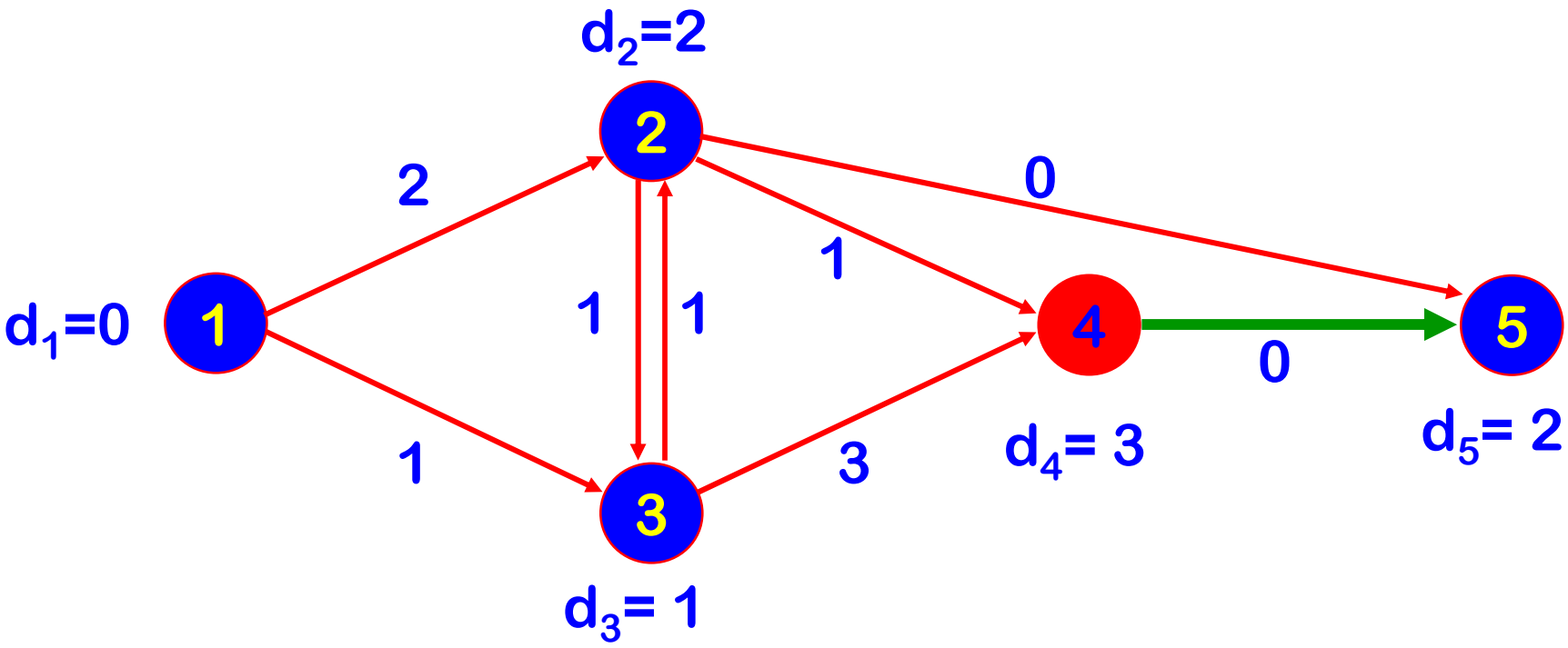
Iter	V	d_1	d_2	d_3	d_4	d_5	traiter
1	{1}	0	∞	∞	∞	∞	1
2	{2,3}	0	2	1	∞	∞	3
3	{2,4}	0	2	1	4	∞	2
4	{4,5}	0	2	1	3	2	5



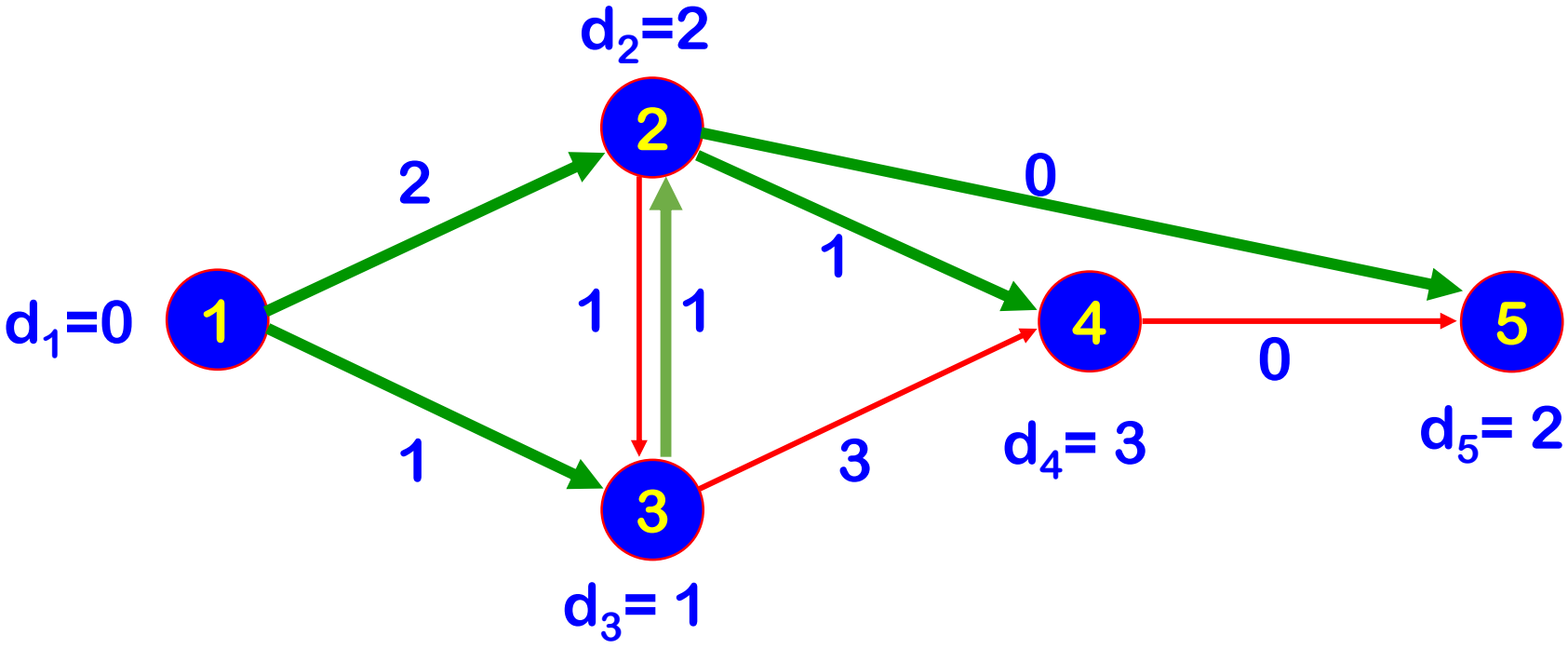
Iter	V	d_1	d_2	d_3	d_4	d_5	traiter
1	{1}	0	∞	∞	∞	∞	1
2	{2,3}	0	2	1	∞	∞	3
3	{2,4}	0	2	1	4	∞	2
4	{4,5}	0	2	1	3	2	5
5	{4}	0	2	1	3	2	4



Iter	V	d_1	d_2	d_3	d_4	d_5	traiter
1	{1}	0	∞	∞	∞	∞	1
2	{2,3}	0	2	1	∞	∞	3
3	{2,4}	0	2	1	4	∞	2
4	{4,5}	0	2	1	3	2	5
5	{4}	0	2	1	3	2	4



Iter	V	d_1	d_2	d_3	d_4	d_5	traiter
1	{1}	0	∞	∞	∞	∞	1
2	{2,3}	0	2	1	∞	∞	3
3	{2,4}	0	2	1	4	∞	2
4	{4,5}	0	2	1	3	2	5
5	{4}	0	2	1	3	2	4
	{}	0	2	1	3	2	



Iter	V	d_1	d_2	d_3	d_4	d_5	traiter
1	{1}	0	∞	∞	∞	∞	1
2	{2,3}	0	2	1	∞	∞	3
3	{2,4}	0	2	1	4	∞	2
4	{4,5}	0	2	1	3	2	5
5	{4}	0	2	1	3	2	4
	{}	0	2	1	3	2	

Notes

Propriétés si l'algorithme se termine

- $\forall j$ t.q. $d_j < \infty$,
  - d_j est la longueur du plus court chemin entre α et j.
  - $d_j = \min_{(i,j) \in \mathcal{A}} d_i + a_{ij}$ si $j \neq \alpha$ [Eq. de Bellman]
  - $d_\alpha = 0$
- $d_j = +\infty$ ssi il n'y a pas de chemin reliant α et j.

L'algorithme se termine ssi il n'y a aucun chemin commençant en α et contenant un cycle à coût négatif

Notes

- Les équations de Bellman permettent de reconstituer les plus courts chemins à partir des étiquettes.

- Le prédécesseur du nœud j dans le plus court chemin est celui qui réalise le minimum de l'équation de Bellman.