

Plan

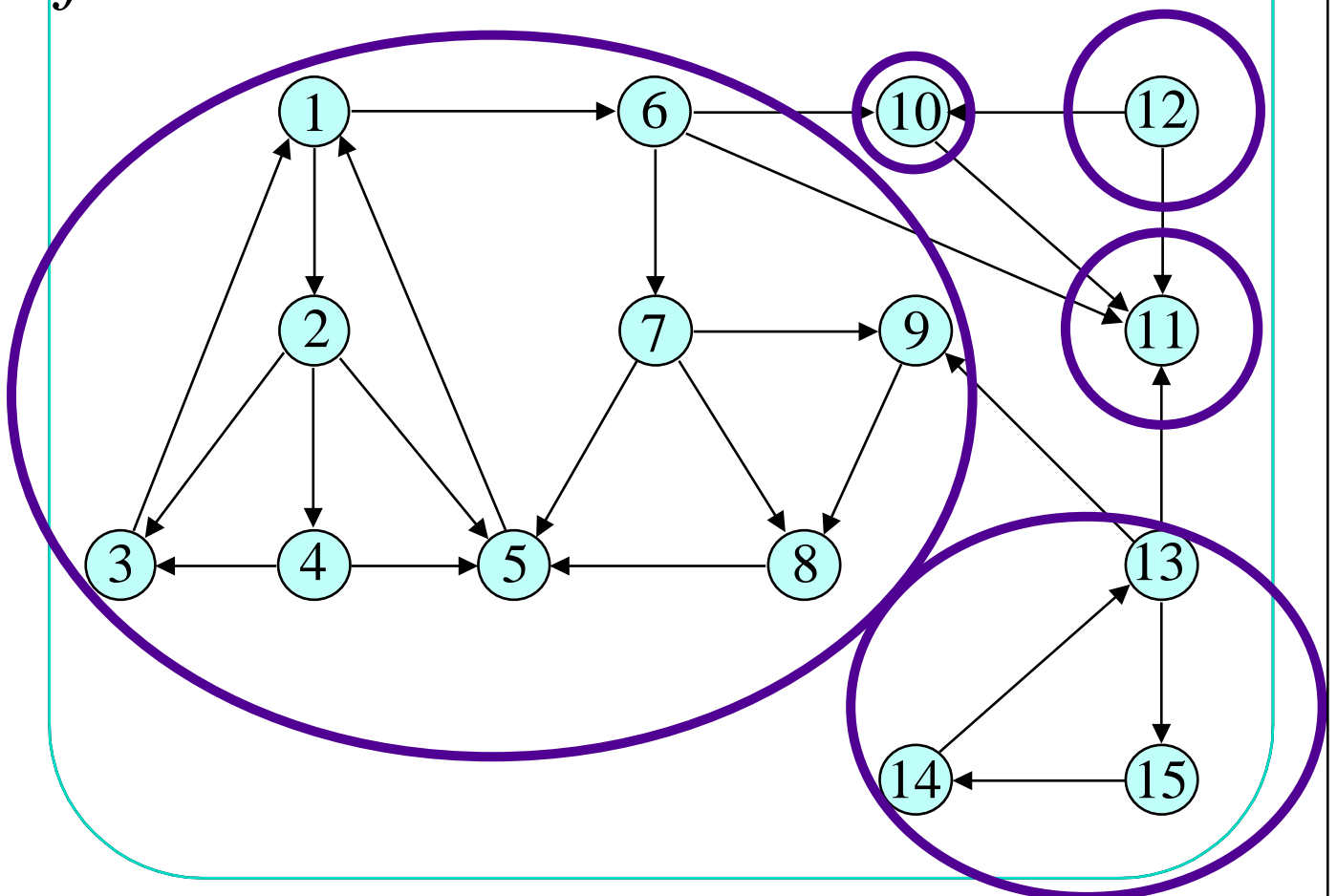
- Composantes fortement connexes d'un graphe
- Algorithme de Roy-Warshall
- Algorithme de Floyd

Composantes fortement connexes

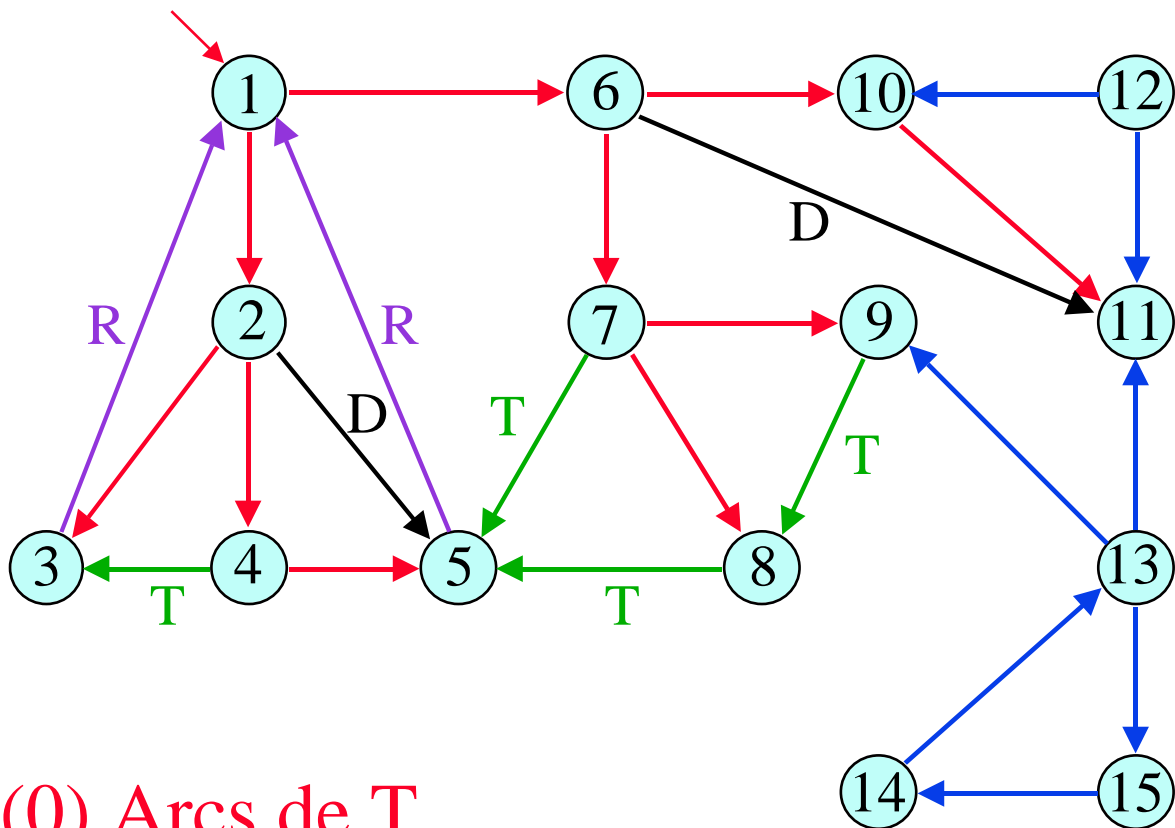
Soit G un graphe. On pose

$x \sim_G y$ $\left\{ \begin{array}{l} \text{il existe un chemin} \\ \text{de } x \text{ à } y \text{ et de } y \text{ à } x \end{array} \right.$

Les classes de cette relation d'équivalence sont les *composantes fortement connexes*.



Soit G un graphe et T une arborescence de Trémaux dans G .



(0) Arcs de T .

(1) Arcs inaccessibles.

(2) Arcs de descente : (x,y) avec y descendant de x .

(3) Arcs retour : x descendant de y .

(4) Arcs transverses : x non descendant de y et y non descendant de x .

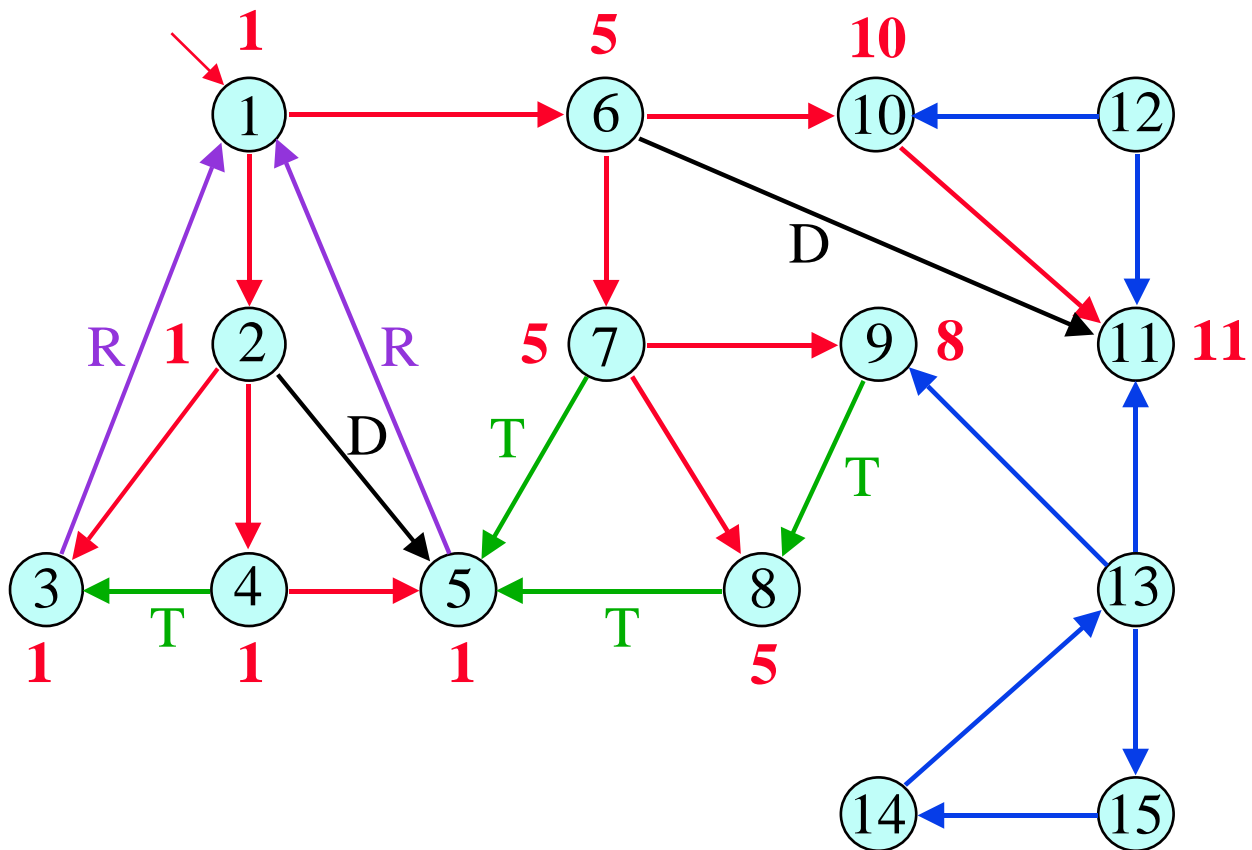
Points d'attache

Point d'attache d'un sommet x de T : plus petit y tel qu'il existe un chemin de x à y (éventuellement vide) contenant au plus un arc **retour** ou **transverse**.

Thm. Si un sommet est égal à son point d'attache $at(x)$, et si pour tout descendant y de x dans T , on a $at(y) < y$, alors l'ensemble des descendants de x forme une composante fortement connexe.

Algorithme : Trouver une composante fortement connexe. La retirer et recommencer.

Calcul des points d'attache



On trouvera successivement les composantes $\{11\}$, $\{10\}$ et $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Définitions

Point d'articulation

Sommet d'un graphe, qui, si on le supprime, disconnecte le graphe.

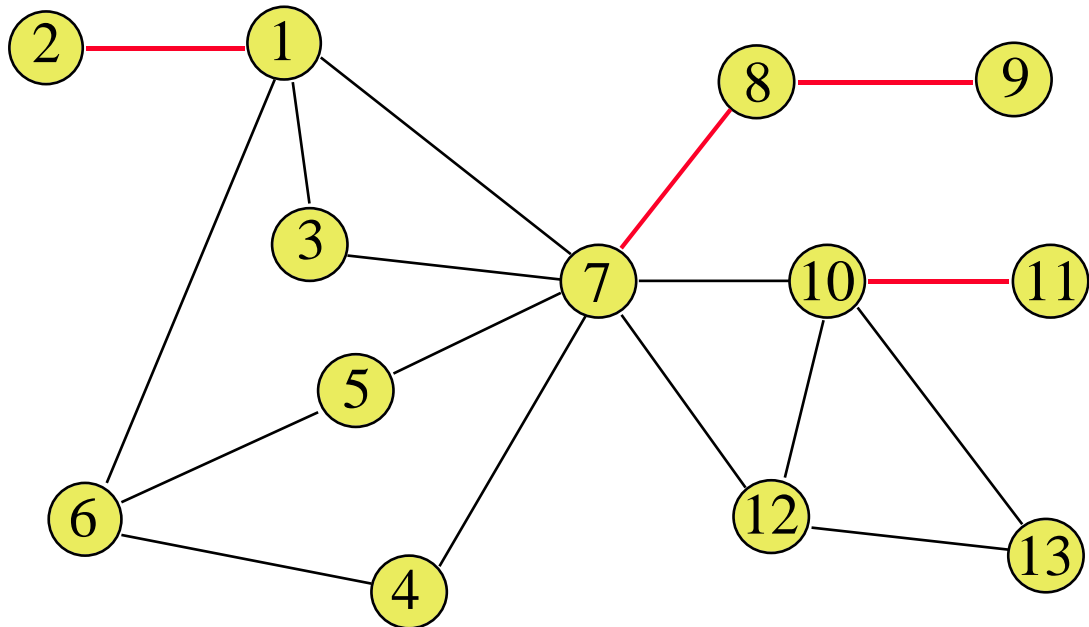
Pont

Arête d'un graphe, qui, si on la supprime, disconnecte le graphe.

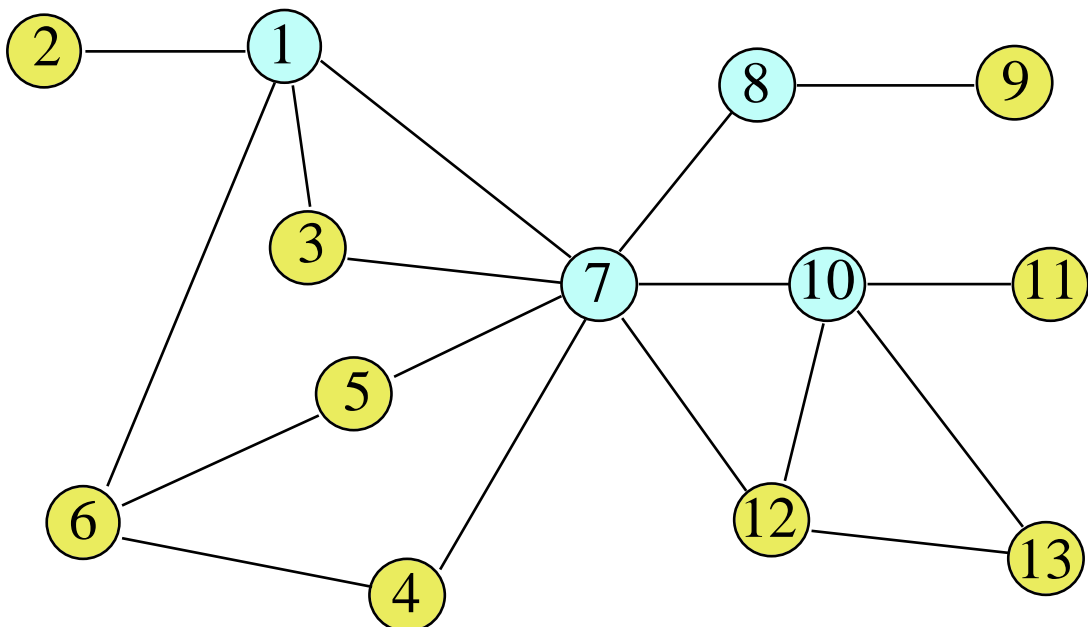
Composantes 2-connexes

Deux arêtes sont dans la même composante 2-connexe s'il existe un circuit (simple) les contenant.

Les ponts d'un graphe

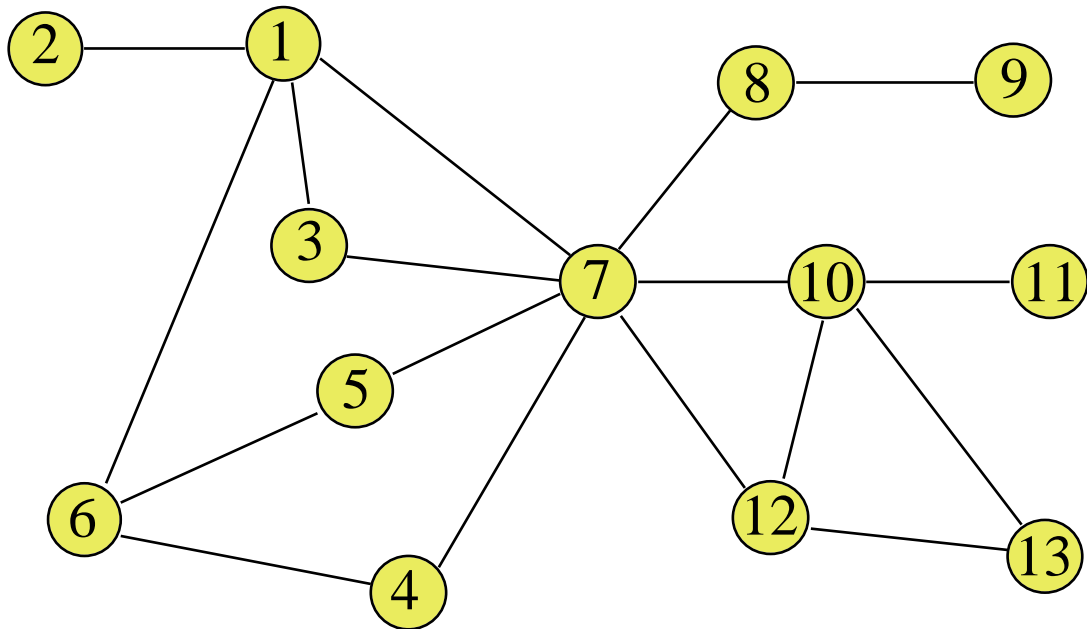


Les points d'articulation d'un graphe

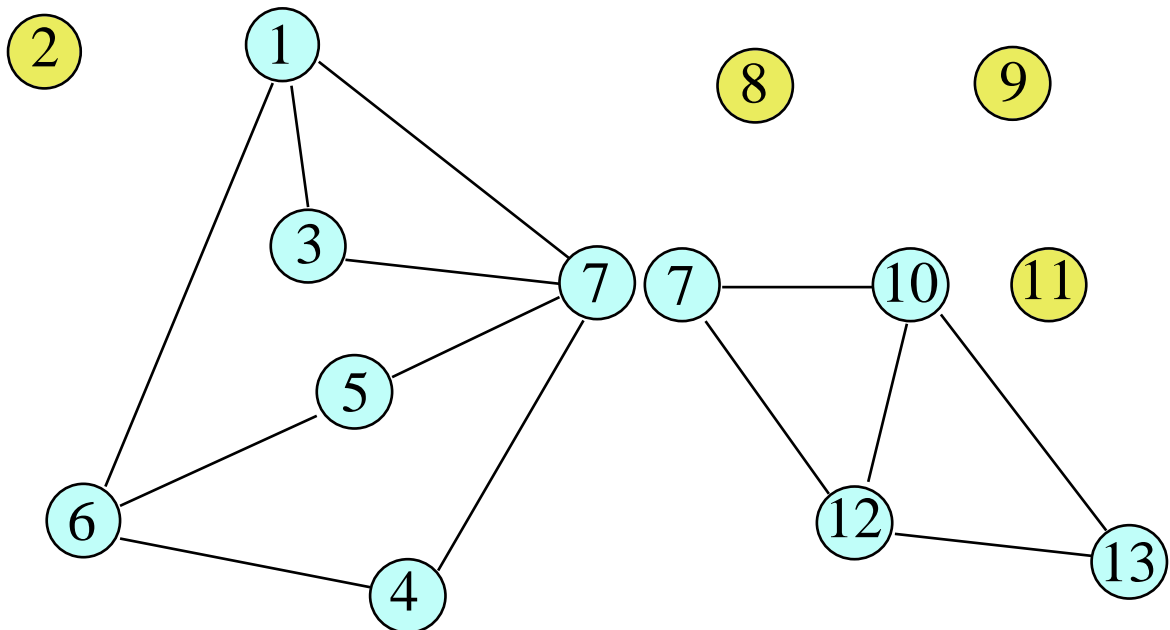


Si on supprime un point d'articulation, le graphe n'est plus connexe

2-connexité

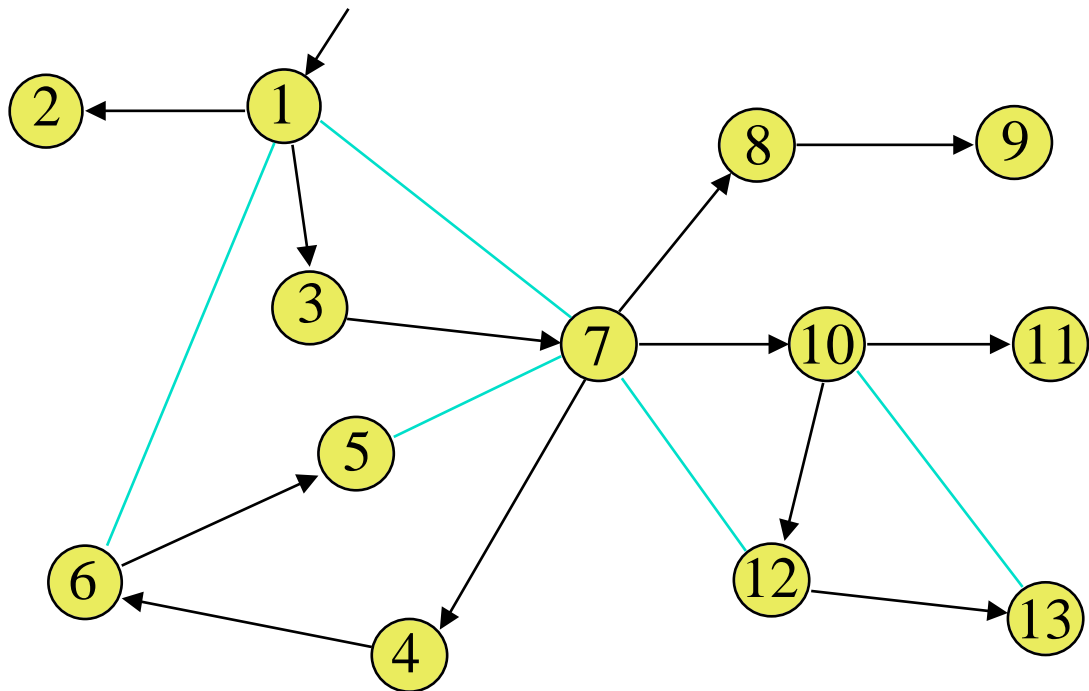


Les composantes 2-connexes



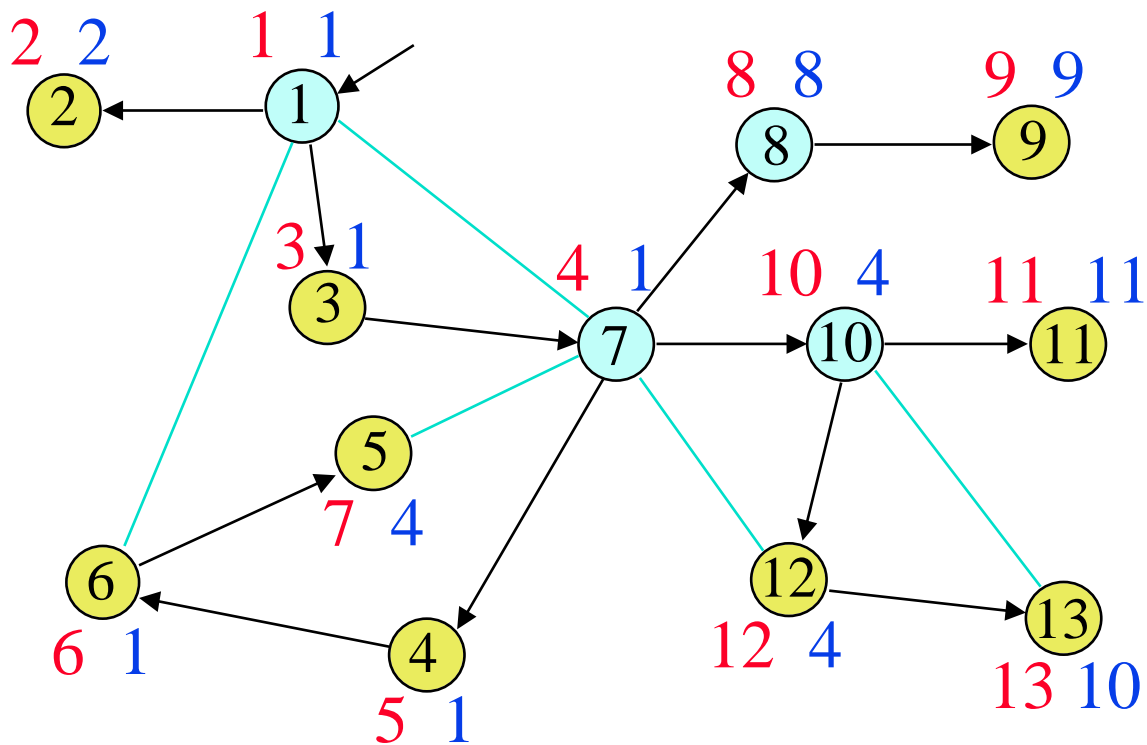
Dans une composante 2-connexes, il existe un circuit entre deux sommets quelconques.

Recherche des points d'articulation



Un parcours en profondeur et son arbre. Un sommet s (autre que la racine) n'est **pas point d'articulation** si chaque fils t de s possède un descendant connecté à un ancêtre de s (par un arc en **bleu**).

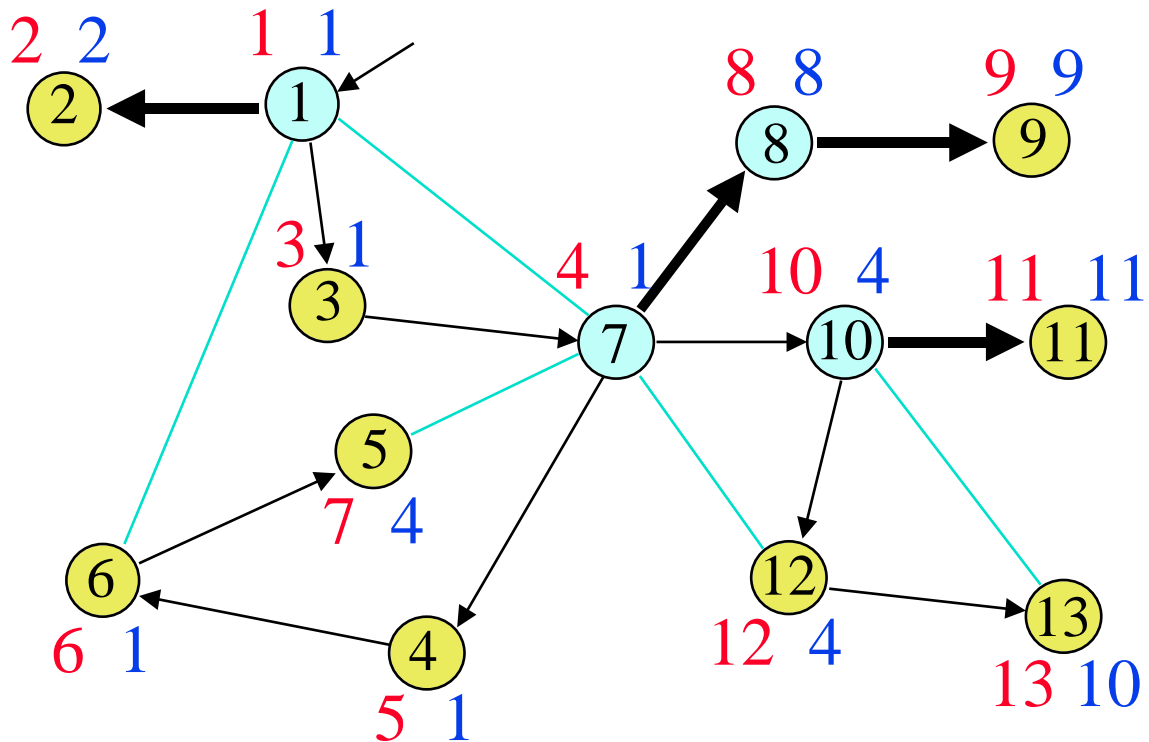
La **racine** est point d'articulation si elle possède **deux fils au moins**.

Algorithme en $O(n+m)$ 

On effectue un parcours récursif en profondeur (en **rouge**) en calculant le rang du sommet "retour" le moins élevé (nombre **bleu**).

Les points d'articulation (autre que la racine) sont ceux dont au moins un des fils a les deux nombres égaux.

Algorithme en $O(n+m)$



Les **ponts** (en gras) sont les arêtes dont le sommet d'arrivée a les deux nombres égaux :

(1, 2), (7, 8), (10, 11), (8, 9).

Relation d'accessibilité (fermeture transitive)

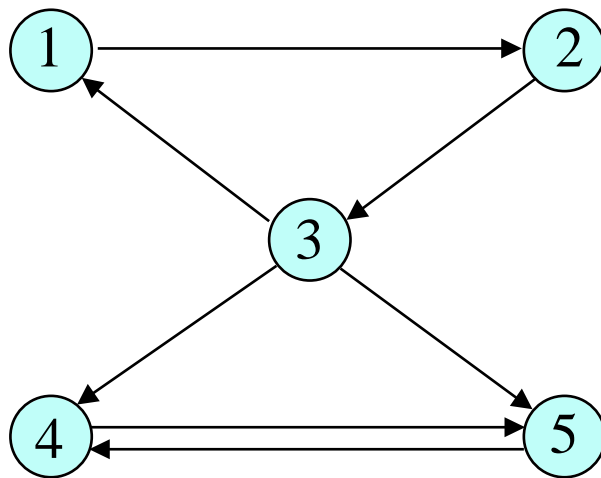
Soit $S = \{1, 2, \dots, n\}$ les sommets.
Soit G_0 le graphe initial, complété
d'une boucle en chaque sommet.

On note G_k le graphe (S, A_k) avec
 $(i, j) \in A_k$ ssi il existe dans G_0 un
chemin de i à j dont les sommets
internes sont dans $\{1, 2, \dots, k\}$

Proposition

- On a $(i, j) \in A_k$ ssi $(i, j) \in A_{k-1}$
ou $(i, k) \in A_{k-1}$ et $(k, j) \in A_{k-1}$
- Pour $j = k$, on trouve
 $(i, k) \in A_k$ ssi $(i, k) \in A_{k-1}$

X, Petite classe 9



Matrice d'adjacence

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$$A_0 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$$A_1 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$$A_2 = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$$A_3 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$$A_4 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Programme

```
void fermetureTransitive ()
{
    for (int k = 0; k < n; k++)
        for (int i = 0; i < n; i++)
            for (int j = 0; j < n; j++)
                if (m[i][k] && m[k][j])
                    m[i][j] = true;
}
```

```
void RoyWarshall ()
{
    for (int i = 0; i < n; i++)
        m[i][i] = true;
    for (int k = 0; k < n; k++)
        for (int i = 0; i < n; i++)
            if (m[i][k])
                for (int j = 0; j < n; j++)
                    m[i][j] = m[i][j] || (m[i][k]
                                            && m[k][j]);
}
```

Complexité : $O(n^3)$

Justification

On suppose qu'à l'étape k , i, j ,
 $M[i,j]$ contient $M_{i,j}^{(k-1)}$ avant la ligne

$$M[i,j] = M[i,j] \parallel (M[i,k] \ \&\& \ M[k,j]);$$

Alors $M[i, k]$ contient

$$M_{i,k}^{(k-1)} \text{ si } k = j \text{ et } M_{i,k}^{(k)} \text{ si } k < j$$

et $M[k, j]$ contient

$$M_{k,j}^{(k-1)} \text{ si } k = i \text{ et } M_{k,j}^{(k)} \text{ si } k < i$$

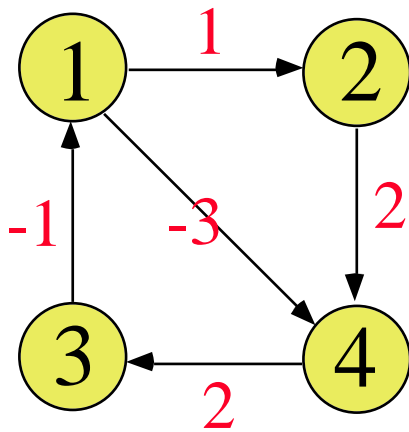
D'après la proposition, on aura
 dans tous les cas :

$$\begin{aligned} M[i,j] &= M_{i,j}^{(k-1)} \text{ or } (M_{i,k}^{(k-1)} \text{ and } M_{k,j}^{(k-1)}) \\ &= M_{i,j}^{(k)} \end{aligned}$$

Complexité

- Le **parcours en profondeur** conduit à un algorithme pour la fermeture transitive en
 $O(n(m + n))$ (**listes**)
 $O(n^3)$ (**matrices**)
- L'algorithme de **Roy-Warshall** est en $O(n^3)$.

Chemins de valeur maximale



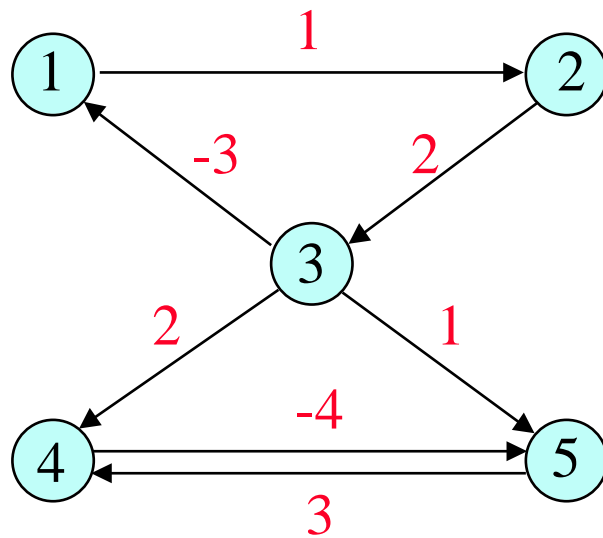
0	1	-	-3
-	0	-	2
-1	-	0	-
-	-	2	0

Il s'agit de trouver les chemins **élémentaires** de valeur maximum. Soit $M_{i,j}^{(k)}$ la valeur maximale des chemins élémentaires de i à j dont les sommets intérieurs sont dans $\{1, 2, \dots, k\}$.

$$M_{i,j}^{(0)} = \begin{cases} M_{i,j} & \text{si } (i, j) \in A \text{ et } i \neq j, \\ 0 & \text{si } i = j, \\ - & \text{sinon.} \end{cases}$$

$$M_{i,j}^{(k)} = \max \{ M_{i,j}^{(k-1)}, M_{i,k}^{(k-1)} + M_{k,j}^{(k-1)} \}$$

X, Petite classe 9



Résultat

$$A_0 = \begin{matrix} 0 & 1 & * & * & * \\ * & 0 & 2 & * & * \\ -3 & * & 0 & 2 & 1 \\ * & * & * & 0 & -4 \\ * & * & * & 3 & 0 \end{matrix}$$

$$A_1 = \begin{matrix} 0 & 1 & * & * & * \\ * & 0 & 2 & * & * \\ -3 & -2 & 0 & 2 & 1 \\ * & * & * & 0 & -4 \\ * & * & * & 3 & 0 \end{matrix}$$

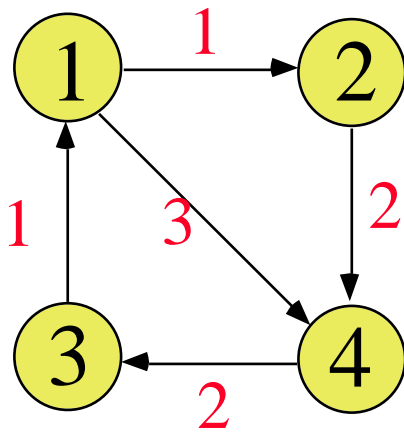
$$A_2 = \begin{matrix} 0 & 1 & 3 & * & * \\ * & 0 & 2 & * & * \\ -3 & -2 & 0 & 2 & 1 \\ * & * & * & 0 & -4 \\ * & * & * & 3 & 0 \end{matrix}$$

$$A_3 = \begin{matrix} 0 & 1 & 3 & 5 & 4 \\ -1 & 0 & 2 & 4 & 3 \\ -3 & -2 & 0 & 2 & 1 \\ * & * & * & 0 & -4 \\ * & * & * & 3 & 0 \end{matrix}$$

$$A_4 = \begin{matrix} 0 & 1 & 3 & 5 & 4 \\ -1 & 0 & 2 & 4 & 3 \\ -3 & -2 & 0 & 2 & 1 \\ * & * & * & 0 & -4 \\ * & * & * & 3 & 0 \end{matrix}$$

$$A_5 = \begin{matrix} 0 & 1 & 3 & 7 & 4 \\ -1 & 0 & 2 & 6 & 3 \\ -3 & -2 & 0 & 4 & 1 \\ * & * & * & 0 & -4 \\ * & * & * & 3 & 0 \end{matrix}$$

Plus court chemin



0	1	+	3
+	0	+	2
1	+	0	+
-	+	2	0

Il s'agit de trouver les chemins les plus courts.

Soit $M_{i,j}^{(k)}$ la valeur maximale des chemins élémentaires de i à j dont les sommets intérieurs sont dans $\{1, 2, \dots, k\}$.

$$M_{i,j}^{(0)} = \begin{cases} M_{i,j} & \text{si } (i, j) \in A \text{ et } i \neq j, \\ 0 & \text{si } i = j, \\ + & \text{sinon.} \end{cases}$$

$$M_{i,j}^{(k)} = \min \{ M_{i,j}^{(k-1)}, M_{i,k}^{(k-1)} + M_{k,j}^{(k-1)} \}$$

Plus court chemin

```
final static int PLUSINFINI = 10000000000;
```

```
void PlusCourtChemin()
```

```
{
```

```
    for (int i = 0; i < n; i++)
```

```
        m[i][i] = 0;
```

```
    for (int k = 0; k < n; k++)
```

```
        for (int i = 0; i < n; i++)
```

```
            for (int j = 0; j < n; j++)
```

```
                m[i][j] = Math.min(m[i][j],
```

```
                    Somme(m[i][k], m[k][j]));
```

```
}
```

```
int Somme(int x, int y)
```

```
{
```

```
    if (x == PLUSINFINI || y == PLUSINFINI)
```

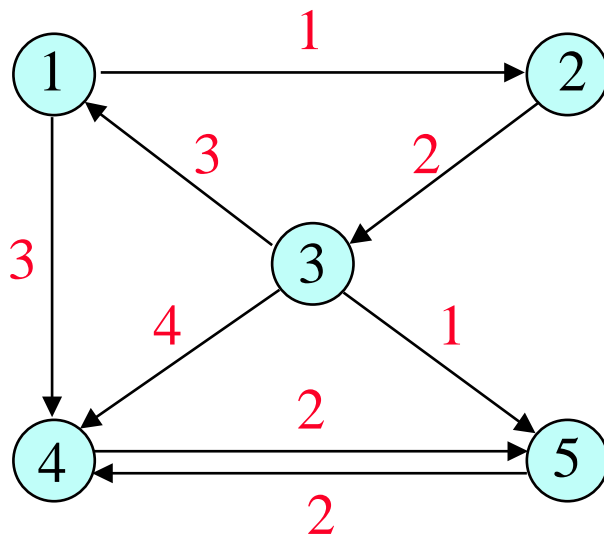
```
        return PLUSINFINI;
```

```
    return(x + y);
```

```
}
```

L'algorithme est en $O(n^3)$.

X, Petite classe 9



Résultat

$$A_0 = \begin{matrix} 0 & 1 & * & 3 & * \\ * & 0 & 2 & * & * \\ 3 & * & 0 & 4 & 1 \\ * & * & * & 0 & 2 \\ * & * & * & 2 & 0 \end{matrix}$$

$$A_1 = \begin{matrix} 0 & 1 & * & 3 & * \\ * & 0 & 2 & * & * \\ 3 & 4 & 0 & 4 & 1 \\ * & * & * & 0 & 2 \\ * & * & * & 2 & 0 \end{matrix}$$

$$A_2 = \begin{matrix} 0 & 1 & 3 & 3 & * \\ * & 0 & 2 & * & * \\ 3 & 4 & 0 & 4 & 1 \\ * & * & * & 0 & 2 \\ * & * & * & 2 & 0 \end{matrix}$$

$$A_3 = \begin{matrix} 0 & 1 & 3 & 3 & 4 \\ 5 & 0 & 2 & 6 & 3 \\ 3 & 4 & 0 & 4 & 1 \\ * & * & * & 0 & 2 \\ * & * & * & 2 & 0 \end{matrix}$$

$$A_4 = \begin{matrix} 0 & 1 & 3 & 3 & 4 \\ 5 & 0 & 2 & 6 & 3 \\ 3 & 4 & 0 & 4 & 1 \\ * & * & * & 0 & 2 \\ * & * & * & 2 & 0 \end{matrix}$$

$$A_5 = \begin{matrix} 0 & 1 & 3 & 3 & 4 \\ 5 & 0 & 2 & 5 & 3 \\ 3 & 4 & 0 & 3 & 1 \\ * & * & * & 0 & 2 \\ * & * & * & 2 & 0 \end{matrix}$$

Semi-anneau

Ensemble k muni de deux opérations associatives, $+$ et \times , et de deux éléments distingués 0 et 1 , tels que

- (1) $(k, +, 0)$ est un monoïde commutatif : l'addition est commutative et admet 0 comme élément neutre.
- (2) $(k, \times, 1)$ est un monoïde : la multiplication admet 1 comme élément neutre
- (3) distributivité : $a, b, c \in k$
 $(a + b) \times c = a \times c + b \times c$ et
 $c \times (a + b) = c \times a + c \times b$
- (4) 0 est absorbant. Pour tout $a \in k$,
 $0 \times a = a \times 0 = 0$

Exemples de semi-anneaux

(1) Tous les anneaux.

(2) $(\mathbb{N}, \{+, \cdot\}, \min, +, \cdot, 0)$

(3) $(\mathbb{N}, \{-, +\}, \max, +, -, 0)$
avec $(- \) + (+ \) = (+ \)$

(4) $(\mathbb{N}, \{-, +\}, \min, \max,$
 $\{+\}, \{-\})$

(5) $(\{\text{Vrai}, \text{Faux}\}, \text{OU}, \text{ET})$

(6) $(\mathcal{P}(E), \cup, \cap)$

(7) Les matrices carrées de
dimension n sur un semi-anneau

Semi-anneau complet

Semi-anneau dans lequel on peut définir des **sommes infinies** que l'on peut manipuler comme des sommes finies (associativité et distributivité comprises).

(On peut donner une définition plus rigoureuse!)

Soit a un élément d'un semi-anneau complet. On pose

$$a^* = 1 + a + a^2 + \dots + a^k + \dots$$

Exemples. Dans $(\mathbb{N}, \min, +)$

$$a^* = \min(0, a, 2a, \dots) = 0$$

Dans $(\mathbb{N} \setminus \{0\}, \{ -, + \}, \max, +)$

$$a^* = \max(0, a, 2a, \dots) = + \quad \text{si } a$$

> 0

Soit M une matrice $n \times n$ à coefficients dans un semi-anneau idempotent complet. On pose

$$M^* = I + M + M^2 + \dots + M^k + \dots$$

Algorithme (Jordan). Notons $M^{(0)}$, ..., $M^{(n)}$ les matrices définies par

$$M^{(0)} = I + A \quad \text{et, pour } 1 \leq k \leq n$$

$$M_{i,j}^{(k)} = M_{i,j}^{(k-1)} + M_{i,k}^{(k-1)} \times (M_{k,k}^{(k-1)})^* \times M_{k,j}^{(k-1)}$$

$$\text{Alors } M^* = M^{(k)}$$

Exemple : dans le semi-anneau $(\mathbb{N}, \min, +)$, la formule devient

$$M_{i,j}^{(k)} = \min(M_{i,j}^{(k-1)}, M_{i,k}^{(k-1)} + M_{k,j}^{(k-1)})$$