# Classification à partir d'une collection de matrices

Clément Grimal et Gilles Bisson (LIG, Université de Grenoble)

Atelier REiSO - 25 Mai 2010

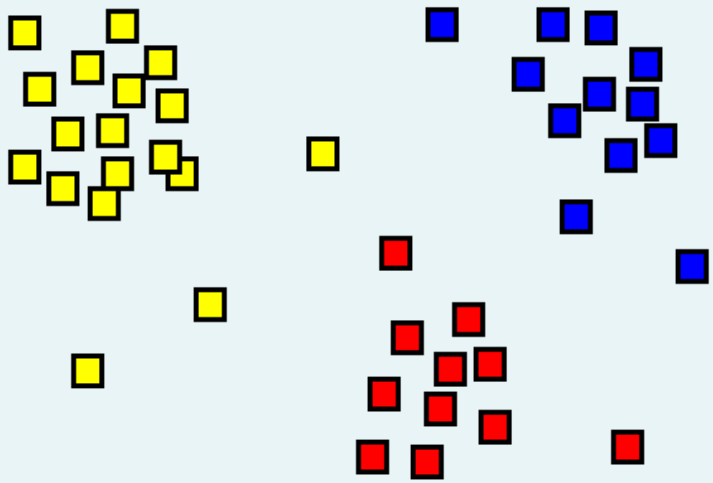

UNIVERSITÉ DE
GRENOBLE

# Classification...

La classification *(ou clustering)* : organiser des *individus* selon un critère de similarité, en ≪ paquets ≫ homogènes

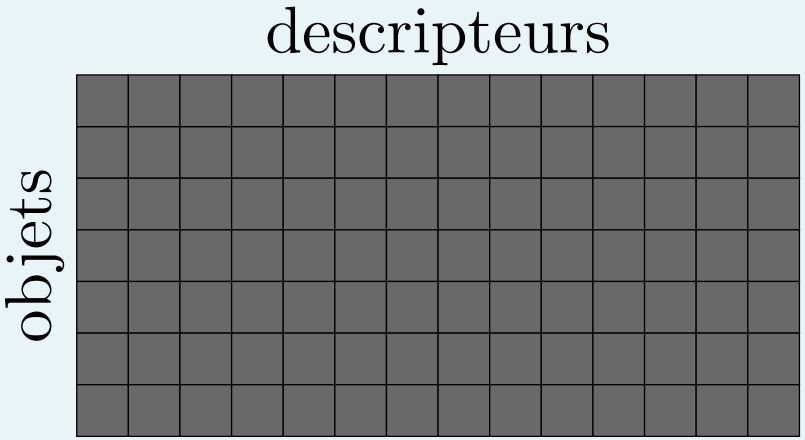
# Classification...

La classification *(ou clustering)* : organiser des *individus* selon un critère de similarité, en ≪ paquets ≫ homogènes



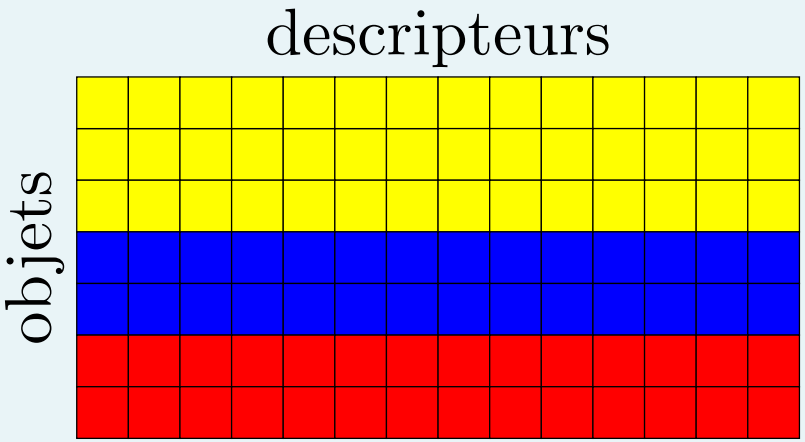
# ...et Bi-classification

La classification vue sous forme matricielle :



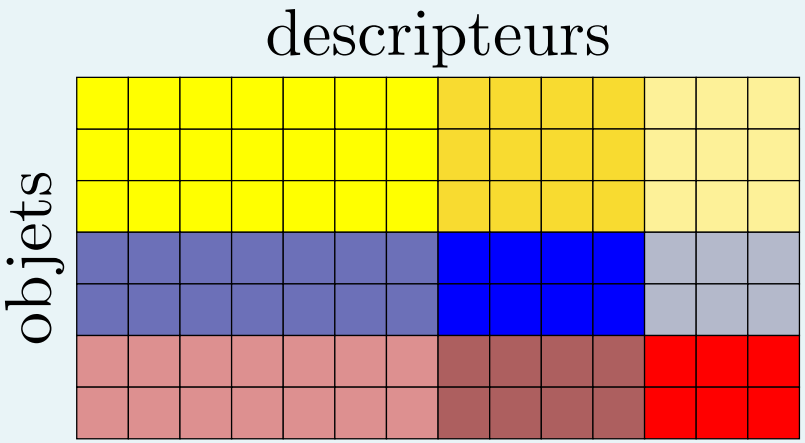
# ...et Bi-classification

La classification vue sous forme matricielle :



# ...et Bi-classification

Si les variables sont suffisamment homogènes...



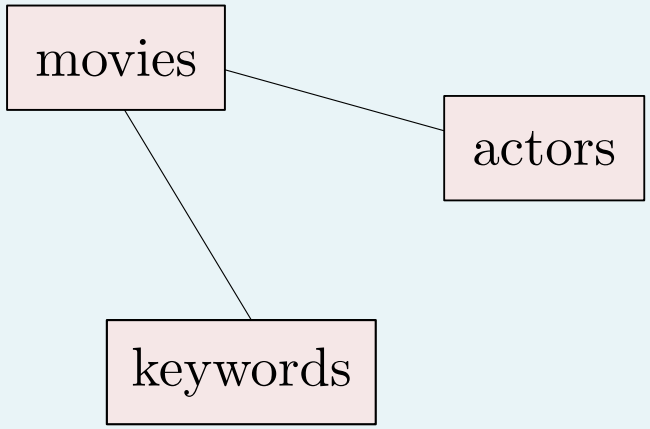

## La co-classification

Classifier **simultanément** deux types d'objets distincts

# Classification de données multi-relationnelles

## Données multi-relationnelles

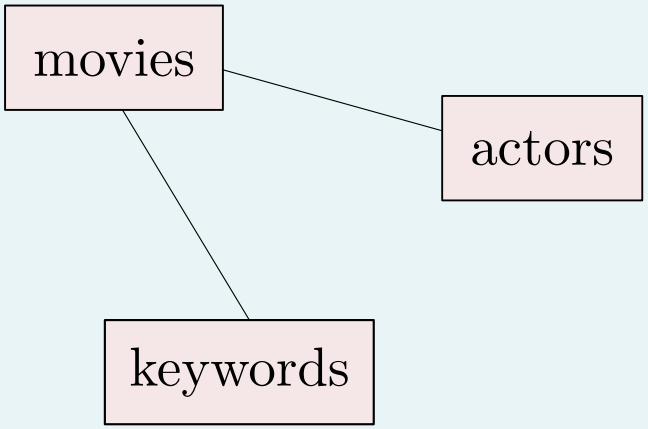
Plusieurs (plus de 2) types d'objets distincts, liés par des relations de type co-occurrence



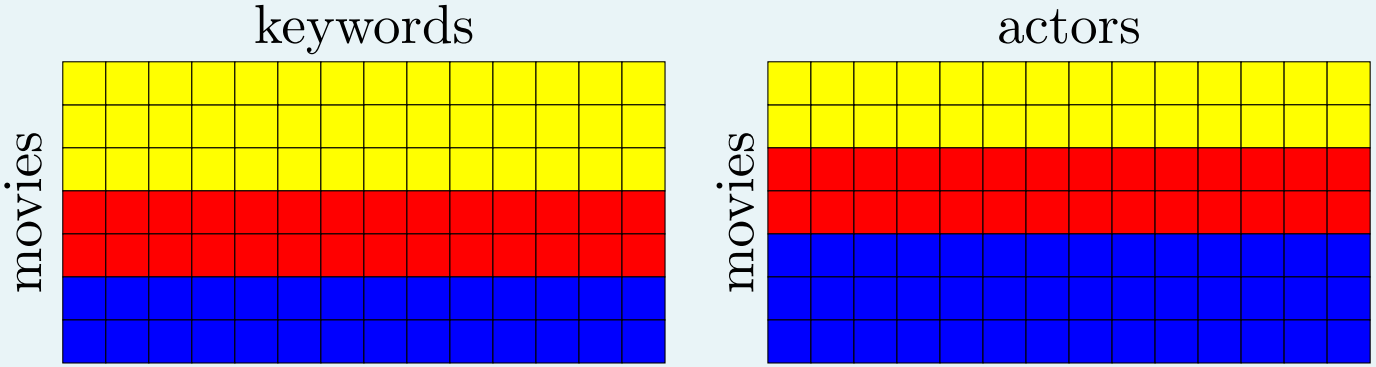
# Classification de données multi-relationnelles

## Données multi-relationnelles

Plusieurs (plus de 2) types d'objets distincts, liés par des relations de type co-occurrence




Autant de matrices de données que de relations, comment classifier au mieux les films ?



# Plan

# L'algorithme $\chi$-Sim (1)

On considère une matrice de données $\mathbf{M}$ *films* - *acteurs* et on cherche à calculer la matrice de similarité entre *films* $\mathbf{SR}$ et entre *acteurs* $\mathbf{SC}$.

# L'algorithme $\chi$-Sim (1)

On considère une matrice de données $\mathbf{M}$ *films* - *acteurs* et on cherche à calculer la matrice de similarité entre *films* $\mathbf{SR}$ et entre *acteurs* $\mathbf{SC}$.

## Fondements

- La similarité entre deux *films* est fonction de la similarité entre les *acteurs* qui y jouent
- La similarité entre deux *acteurs* est fonction de la similarité entre les *films* dans lesquels ils jouent

# L'algorithme $\chi$-SIM (1)

On considère une matrice de données $\mathbf{M}$ *films* - *acteurs* et on cherche à calculer la matrice de similarité entre *films* $\mathbf{SR}$ et entre *acteurs* $\mathbf{SC}$.

## Fondements

- La similarité entre deux *films* est fonction de la similarité entre les *acteurs* qui y jouent
- La similarité entre deux *acteurs* est fonction de la similarité entre les *films* dans lesquels ils jouent

Classiquement, la similarité entre deux films $\mathbf{m}_{i.}$ et $\mathbf{m}_{j.}$ est calculée grâce aux *éléments* communs :

$$\mathrm{Sim}(\mathbf{m}_{i.}, \mathbf{m}_{j.}) =$$
$$\mathrm{F_s}(m_{i1}, m_{j1}) + \mathrm{F_s}(m_{i2}, m_{j2}) + \cdots + \mathrm{F_s}(m_{ic}, m_{jc})$$

# L'algorithme $\chi$-SIM (1)

On considère une matrice de données $\mathbf{M}$ *films* - *acteurs* et on cherche à calculer la matrice de similarité entre *films* $\mathbf{SR}$ et entre *acteurs* $\mathbf{SC}$.

## Fondements

- La similarité entre deux *films* est fonction de la similarité entre les *acteurs* qui y jouent
- La similarité entre deux *acteurs* est fonction de la similarité entre les *films* dans lesquels ils jouent

Si l'on considère que $\forall i \in 1..c, sc_{ii} = 1$ :

$$\mathrm{Sim}(\mathbf{m}_{i\cdot}, \mathbf{m}_{j\cdot}) =$$
$$\mathrm{F_s}(m_{i1}, m_{j1}) \times sc_{11} + \mathrm{F_s}(m_{i2}, m_{j2}) \times sc_{22} + \cdots + \mathrm{F_s}(m_{ic}, m_{jc}) \times sc_{cc}$$

# L'algorithme $\chi$-Sim (1)

On considère une matrice de données $\mathbf{M}$ *films* - *acteurs* et on cherche à calculer la matrice de similarité entre *films* $\mathbf{SR}$ et entre *acteurs* $\mathbf{SC}$.

## Fondements

- La similarité entre deux *films* est fonction de la similarité entre les *acteurs* qui y jouent
- La similarité entre deux *acteurs* est fonction de la similarité entre les *films* dans lesquels ils jouent

Généralisation afin de comparer tous les *acteurs* entre eux :

$$\mathrm{Sim}(\mathbf{m}_{i\cdot}, \mathbf{m}_{j\cdot}) =$$

$$\mathrm{F_s}(m_{i1}, m_{j1}) \times sc_{11} + \mathrm{F_s}(m_{i1}, m_{j2}) \times sc_{12} + \ldots + \mathrm{F_s}(m_{i1}, m_{jc}) \times sc_{1c} +$$
$$\mathrm{F_s}(m_{i2}, m_{j1}) \times sc_{21} + \mathrm{F_s}(m_{i2}, m_{j2}) \times sc_{22} + \ldots + \mathrm{F_s}(m_{i2}, m_{jc}) \times sc_{2c} +$$
$$\ldots$$
$$\mathrm{F_s}(m_{ic}, m_{j1}) \times sc_{c1} + \mathrm{F_s}(m_{ic}, m_{j2}) \times sc_{c2} + \ldots + \mathrm{F_s}(m_{ic}, m_{jc}) \times sc_{cc}$$

# L'algorithme $\chi$-Sim (2)

En pratique, on considère que la fonction de similarité $\mathrm{F_s}$ est le produit, et on normalise les valeurs pour qu'elle appartiennent à $[0, 1]$, par le produit du nombre d'acteurs du i$^{\text{ème}}$ film et du j$^{\text{ème}}$ film, soit $|\mathbf{m}_i.| \times |\mathbf{m}_j.|$

# L'algorithme $\chi$-Sim (2)

En pratique, on considère que la fonction de similarité $\mathrm{F_s}$ est le produit, et on normalise les valeurs pour qu'elle appartiennent à $[0, 1]$, par le produit du nombre d'acteurs du i$^{\text{ème}}$ film et du j$^{\text{ème}}$ film, soit $|\mathbf{m}_{i.}| \times |\mathbf{m}_{j.}|$

Ainsi, on peut calculer :

$$\mathrm{Sim}(\mathbf{m}_{i.}, \mathbf{m}_{j.}) = \frac{\mathbf{m}_{i.} \times \mathbf{SC} \times \mathbf{m}_{j.}^{\mathrm{T}}}{|\mathbf{m}_{i.}| \times |\mathbf{m}_{j.}|}$$

# L'algorithme $\chi$-SIM (2)

En pratique, on considère que la fonction de similarité $\mathrm{F_s}$ est le produit, et on normalise les valeurs pour qu'elle appartiennent à $[0, 1]$, par le produit du nombre d'acteurs du i$^{\text{ème}}$ film et du j$^{\text{ème}}$ film, soit $|\mathbf{m}_{i.}| \times |\mathbf{m}_{j.}|$

Ainsi, on peut calculer :

$$\mathrm{Sim}(\mathbf{m}_{i.}, \mathbf{m}_{j.}) = \frac{\mathbf{m}_{i.} \times \mathbf{SC} \times \mathbf{m}_{j.}^{\mathrm{T}}}{|\mathbf{m}_{i.}| \times |\mathbf{m}_{j.}|}$$

Le calcul pour *tous les films* peut d'exprimer sous forme matricielle :

$$\mathbf{SR} = (\mathbf{M} \times \mathbf{SC} \times \mathbf{M}^T) \circ \mathbf{NR} \quad \text{avec } nr_{ij} = \frac{1}{|\mathbf{m}_{i.}| \times |\mathbf{m}_{j.}|}$$

$$\mathbf{SC} = (\mathbf{M}^T \times \mathbf{SR} \times \mathbf{M}) \circ \mathbf{NC} \quad \text{avec } nc_{ij} = \frac{1}{|\mathbf{m}_{.i}| \times |\mathbf{m}_{.j}|}$$

# L'algorithme $\chi$-SIM (2)

En pratique, on considère que la fonction de similarité $\mathrm{F_s}$ est le produit, et on normalise les valeurs pour qu'elle appartiennent à $[0, 1]$, par le produit du nombre d'acteurs du i$^{\text{ème}}$ film et du j$^{\text{ème}}$ film, soit $|\mathbf{m}_{i.}| \times |\mathbf{m}_{j.}|$

Ainsi, on peut calculer :

$$\mathrm{Sim}(\mathbf{m}_{i.}, \mathbf{m}_{j.}) = \frac{\mathbf{m}_{i.} \times \mathbf{SC} \times \mathbf{m}_{j.}^{\mathrm{T}}}{|\mathbf{m}_{i.}| \times |\mathbf{m}_{j.}|}$$

On résout le système d'équations correspondant de manière itérative :

$$\mathbf{SR}^{(t)} = (\mathbf{M} \times \mathbf{SC}^{(t-1)} \times \mathbf{M}^{T}) \circ \mathbf{NR} \quad \text{avec } nr_{ij} = \frac{1}{|\mathbf{m}_i| \times |\mathbf{m}_j|}$$

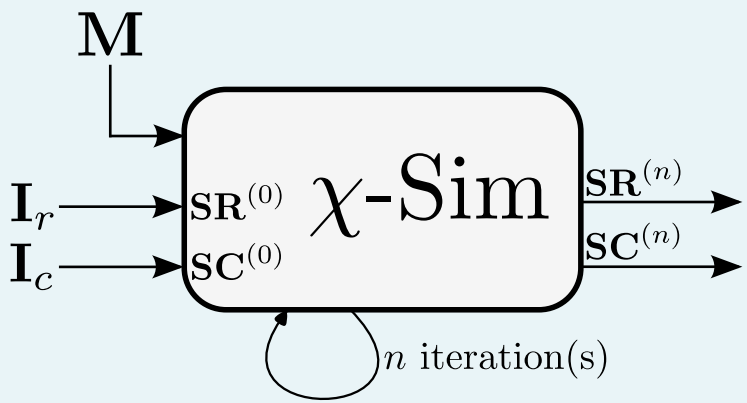
$$\mathbf{SC}^{(t)} = (\mathbf{M}^{T} \times \mathbf{SR}^{(t-1)} \times \mathbf{M}) \circ \mathbf{NC} \quad \text{avec } nc_{ij} = \frac{1}{|\mathbf{m}^i| \times |\mathbf{m}^j|}$$

avec $\mathbf{SR}^{(0)} = \mathbf{I}_r$ et $\mathbf{SC}^{(0)} = \mathbf{I}_c$.

# Principe des extensions

Nous allons maintenant proposer trois méthodes, utilisant l'algorithme $\chi$-Sim afin de classifier des données multi-relationnelles.

L'algorithme $\chi$-Sim sera représenté de façon schématique :

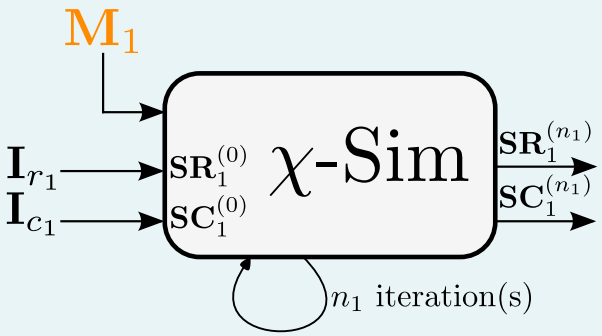



avec $\mathbf{I}_r$ la matrice identité de taille $r$, correspondant à une absence de connaissance *a priori* sur les similarités entre lignes.

# Structure en Cascade

## Principe

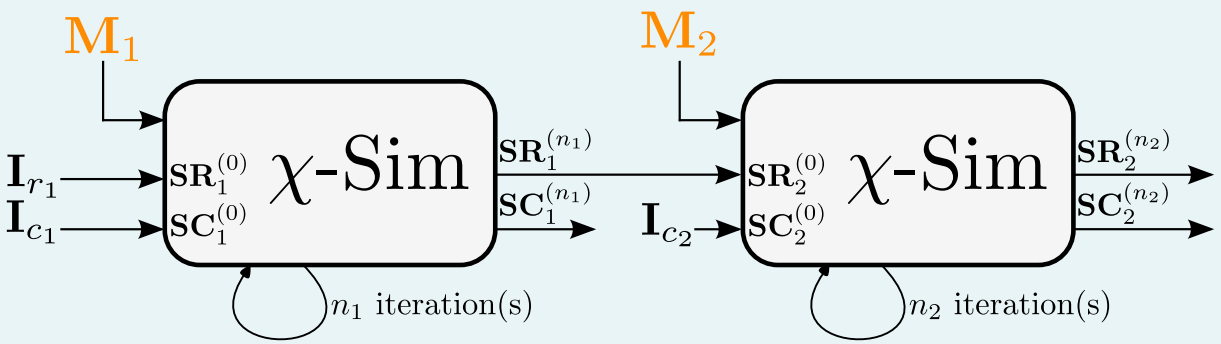
Utiliser la matrice de similarité calculée par une première instance de $\chi$-Sim grâce à une première matrice de données pour initialiser une seconde instance de $\chi$-Sim utilisant une seconde matrice de données



# Structure en Cascade

## Principe

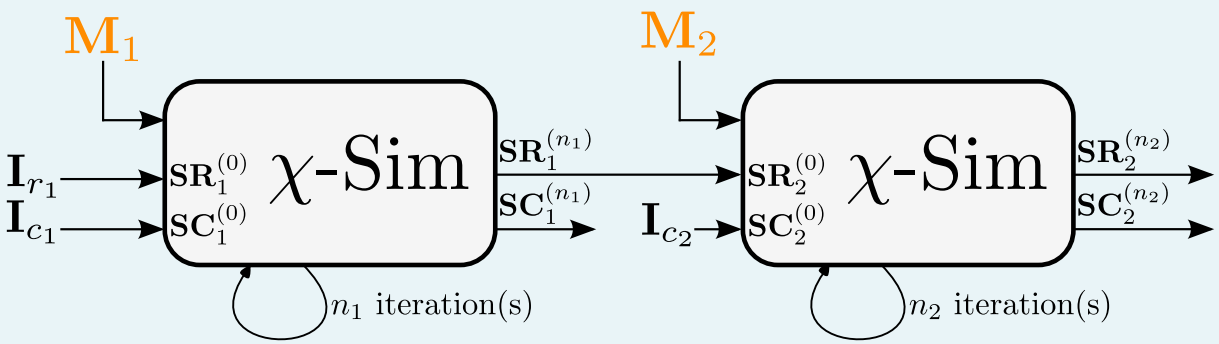
Utiliser la matrice de similarité calculée par une première instance de $\chi$-Sim grâce à une première matrice de données pour initialiser une seconde instance de $\chi$-Sim utilisant une seconde matrice de données



# Structure en Cascade

## Principe

Utiliser la matrice de similarité calculée par une première instance de $\chi$-Sim grâce à une première matrice de données pour initialiser une seconde instance de $\chi$-Sim utilisant une seconde matrice de données



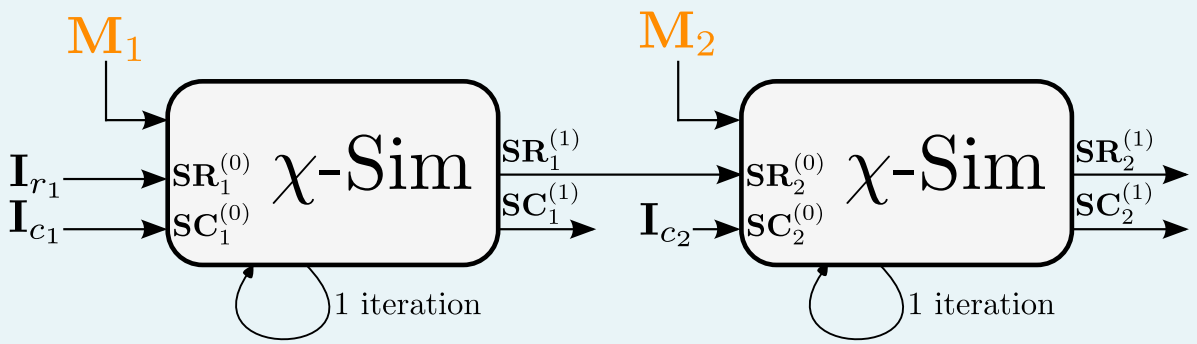

## Questions

- Ordre des matrices ?
- Nombre d'itérations pour chaque instance ?

# Structure en Anneau

## Principe

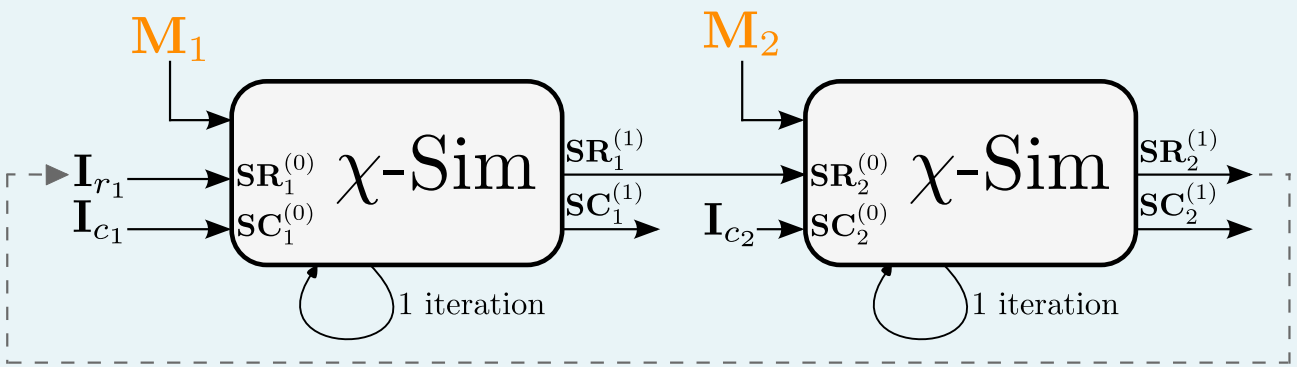
Similaire à la structure en Cascade, sauf que le nombre d'itérations par instance de $\chi$-SIM est fixé à 1, et que l'on peut alterner entre plusieurs instances



# Structure en Anneau

## Principe

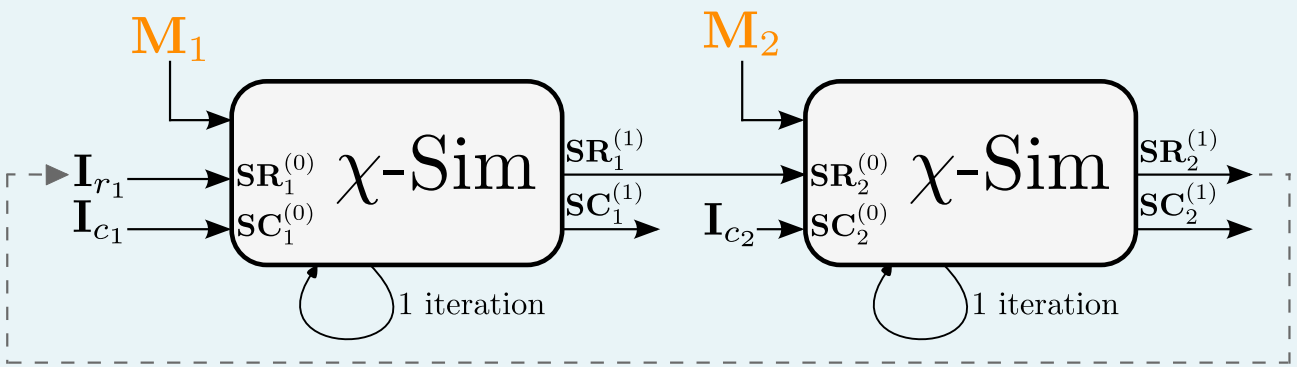
Similaire à la structure en Cascade, sauf que le nombre d'itérations par instance de $\chi$-SIM est fixé à 1, et que l'on peut alterner entre plusieurs instances



# Structure en Anneau

## Principe

Similaire à la structure en Cascade, sauf que le nombre d'itérations par instance de $\chi$-SIM est fixé à 1, et que l'on peut alterner entre plusieurs instances



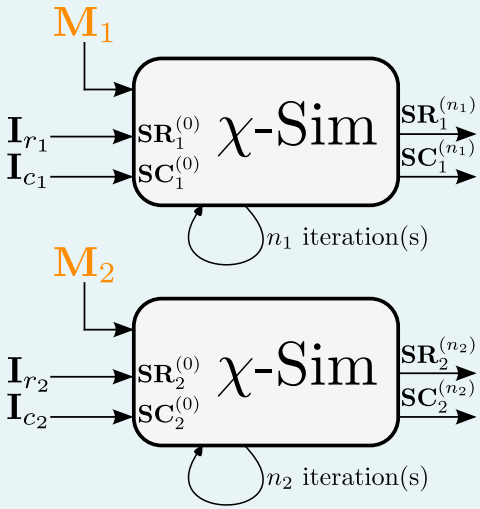

## Questions

- Ordre des matrices ?
- Nombre d'itérations totales ?

# Combinaison

## Principe

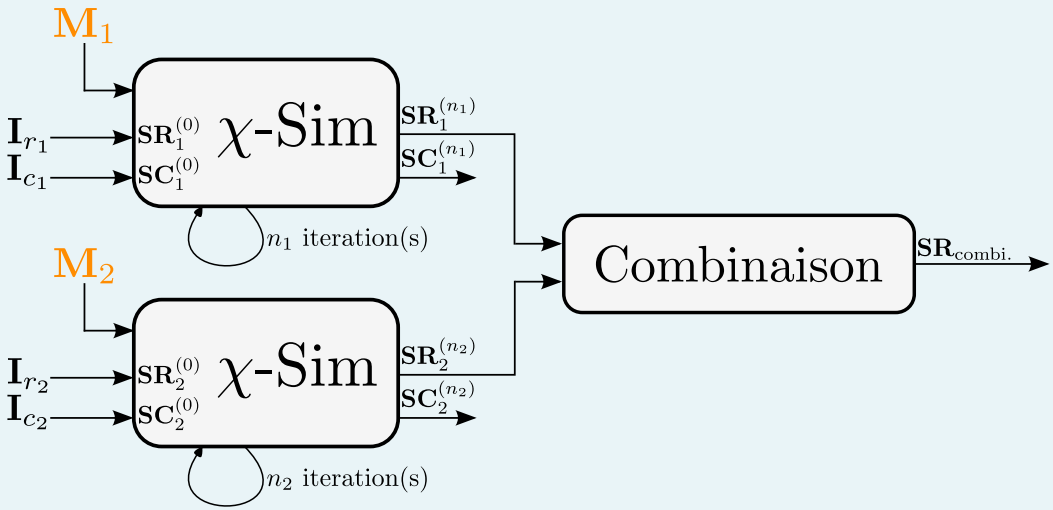
Calcul des matrices de similarité grâce à plusieurs instances de $\chi$-SIM utilisant des matrices de données différentes, puis combinaison de ces matrices



# Combinaison

## Principe

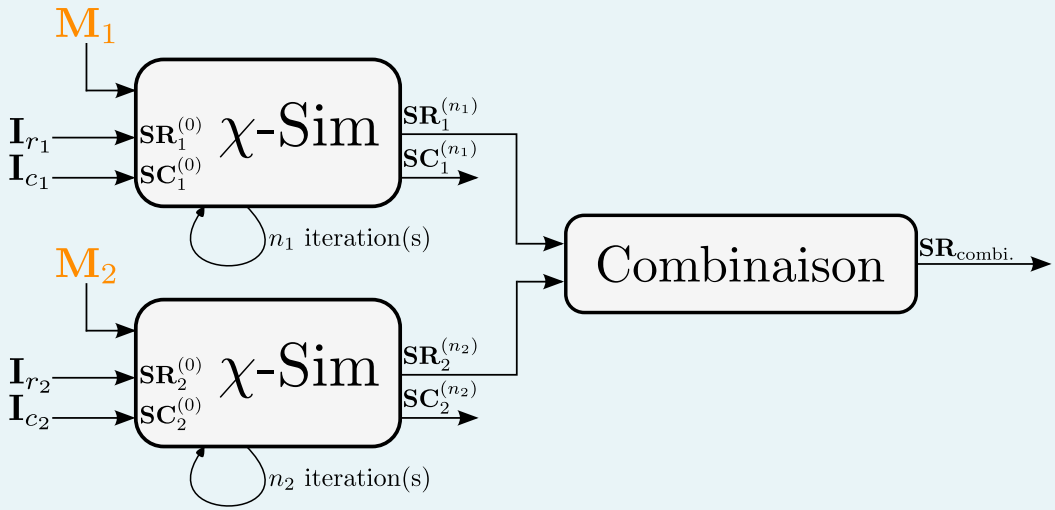
Calcul des matrices de similarité grâce à plusieurs instances de $\chi$-SIM utilisant des matrices de données différentes, puis combinaison de ces matrices



# Combinaison

## Principe

Calcul des matrices de similarité grâce à plusieurs instances de $\chi$-SIM utilisant des matrices de données différentes, puis combinaison de ces matrices




## Questions

- Nombre d'itérations pour chaque instance ?
- Stratégie de combinaison (min, max, moyenne...) ?

# Plan

# Base expérimentale

Créée à partir d'IMDb pour recueillir des films, ainsi que leur genre, leur casting et les mots-clés affectés par des utilisateurs.

# Base expérimentale

Créée à partir d'IMDb pour recueillir des films, ainsi que leur genre, leur casting et les mots-clés affectés par des utilisateurs.

## Description

- 617 films de 17 genres différents (environ 36 films par genre)
- 1878 mots-clés $\rightarrow$ Matrice *films* - *mots-clés* : $\mathbf{M}_1$
- 1398 acteurs $\rightarrow$ Matrice *films* - *acteurs* : $\mathbf{M}_2$

# Base expérimentale

Créée à partir d'IMDb pour recueillir des films, ainsi que leur genre, leur casting et les mots-clés affectés par des utilisateurs.

## Description

- 617 films de 17 genres différents (environ 36 films par genre)
- 1878 mots-clés $\rightarrow$ Matrice *films* - *mots-clés* : $\mathbf{M}_1$
- 1398 acteurs $\rightarrow$ Matrice *films* - *acteurs* : $\mathbf{M}_2$

Objectif : classifier les films par genre, en utilisant $\mathbf{M}_1$ **et** $\mathbf{M}_2$

# Algorithmes utilisés et critère d'évaluation

## Algorithmes utilisés

Comparaison de nos 3 méthodes à :

- la similarité de cosinus (notre « ligne de base »)
- la mesure de similarité induite par LSA (Deerwester et al., 1990)
- l'algorithme de co-classification ITCC (Dhillon et al., 2003)
- la co-similarité $\chi$-SIM (Bisson and Hussain, 2008)

# Algorithmes utilisés et critère d'évaluation

## Algorithmes utilisés

Comparaison de nos 3 méthodes à :

- la similarité de cosinus (notre « ligne de base »)
- la mesure de similarité induite par LSA (Deerwester et al., 1990)
- l'algorithme de co-classification ITCC (Dhillon et al., 2003)
- la co-similarité $\chi$-SIM (Bisson and Hussain, 2008)

## Critère d'évaluation

La précision micro-moyennée introduite par Dhillon et al. (2003), pour laquelle une valeur de 1 correspond à une classification parfaite.

# Résultats

| Méthode | Cosinus | LSA | ITCC | $\chi$-SIM |
|---|---|---|---|---|
| $\mathbf{M}_1$ | 0,225 | 0,277 | 0,280 | **0,282** |
| $\mathbf{M}_2$ | 0,212 | 0,216 | 0,160 | **0,217** |
| $\mathbf{M}_1\mathbf{M}_2$ | 0,284 | **0,295** | 0,266 | 0,280 |

Nous avons également tester ces méthodes avec une matrice nommée $\mathbf{M}_1\mathbf{M}_2$, qui est la concaténation horizontale de $\mathbf{M}_1$ et de $\mathbf{M}_2$.

# Résultats

| Méthode | Cosinus | LSA | ITCC | $\chi$-SIM |
|---|---|---|---|---|
| $\mathbf{M}_1$ | 0,225 | 0,277 | 0,280 | **0,282** |
| $\mathbf{M}_2$ | 0,212 | 0,216 | 0,160 | **0,217** |
| $\mathbf{M}_1\mathbf{M}_2$ | 0,284 | **0,295** | 0,266 | 0,280 |

| Cascade | $\mathbf{M}_1 \rightarrow \mathbf{M}_2$ | $\mathbf{M}_2 \rightarrow \mathbf{M}_1$ |
|---|---|---|
| 1 itération $\rightarrow$ 1 itération | 0,207 | 0,254 |
| 1 itération $\rightarrow$ 2 itérations | **0,227** | 0,241 |
| 1 itérations $\rightarrow$ 3 itérations | 0,222 | 0,285 |
| 2 itérations $\rightarrow$ 1 itérations | 0,216 | **0,292** |
| 2 itérations $\rightarrow$ 2 itérations | 0,227 | 0,246 |
| 2 itérations $\rightarrow$ 3 itérations | 0,225 | 0,285 |

# Résultats

| Méthode | Cosinus | LSA | ITCC | $\chi$-SIM |
|---|---|---|---|---|
| $\mathbf{M}_1$ | 0,225 | 0,277 | 0,280 | **0,282** |
| $\mathbf{M}_2$ | 0,212 | 0,216 | 0,160 | **0,217** |
| $\mathbf{M}_1\mathbf{M}_2$ | 0,284 | **0,295** | 0,266 | 0,280 |

| Anneau | $\mathbf{M}_1 \leftrightarrow \mathbf{M}_2$ | $\mathbf{M}_2 \leftrightarrow \mathbf{M}_1$ |
|---|---|---|
| 3 itérations | **0,292** | 0,224 |
| 4 itérations | 0,219 | **0,266** |

# Résultats

| Méthode | Cosinus | LSA | ITCC | $\chi$-SIM |
|---|---|---|---|---|
| $\mathbf{M_1}$ | 0,225 | 0,277 | 0,280 | **0,282** |
| $\mathbf{M_2}$ | 0,212 | 0,216 | 0,160 | **0,217** |
| $\mathbf{M_1M_2}$ | 0,284 | **0,295** | 0,266 | 0,280 |

| Combinaison | Combinaison : $\mathbf{M_1}\|\mathbf{M_2}$ | | |
|---|---|---|---|
| | Moyenne | Minimum | Maximum |
| 2 itérations - 2 itérations | 0,232 | 0,220 | 0,241 |
| 3 itérations - 3 itérations | 0,222 | 0,225 | **0,266** |

# Conclusion et Perspectives

## Conclusion

- Nous avons testé différentes méthodes utilisant un algorithme de calcul de co-similarité pour traiter des données multi-relationnelles
- Les résultats obtenus sont moins bon que ceux de LSA mais on observe une légère amélioration par rapport à l'utilisation de la méthode $\chi$-SIM seule

# Conclusion et Perspectives

## Conclusion

- Nous avons testé différentes méthodes utilisant un algorithme de calcul de co-similarité pour traiter des données multi-relationnelles
- Les résultats obtenus sont moins bon que ceux de LSA mais on observe une légère amélioration par rapport à l'utilisation de la méthode $\chi$-Sim seule

## Perspectives

- Amélioration de $\chi$-Sim en cours permettant d'obtenir de meilleurs résultats que LSA
- Tester nos méthodes sur des jeux de données
  - plus complexes
  - concernant d'autres domaines
- Trouver des fondements théoriques à notre approche

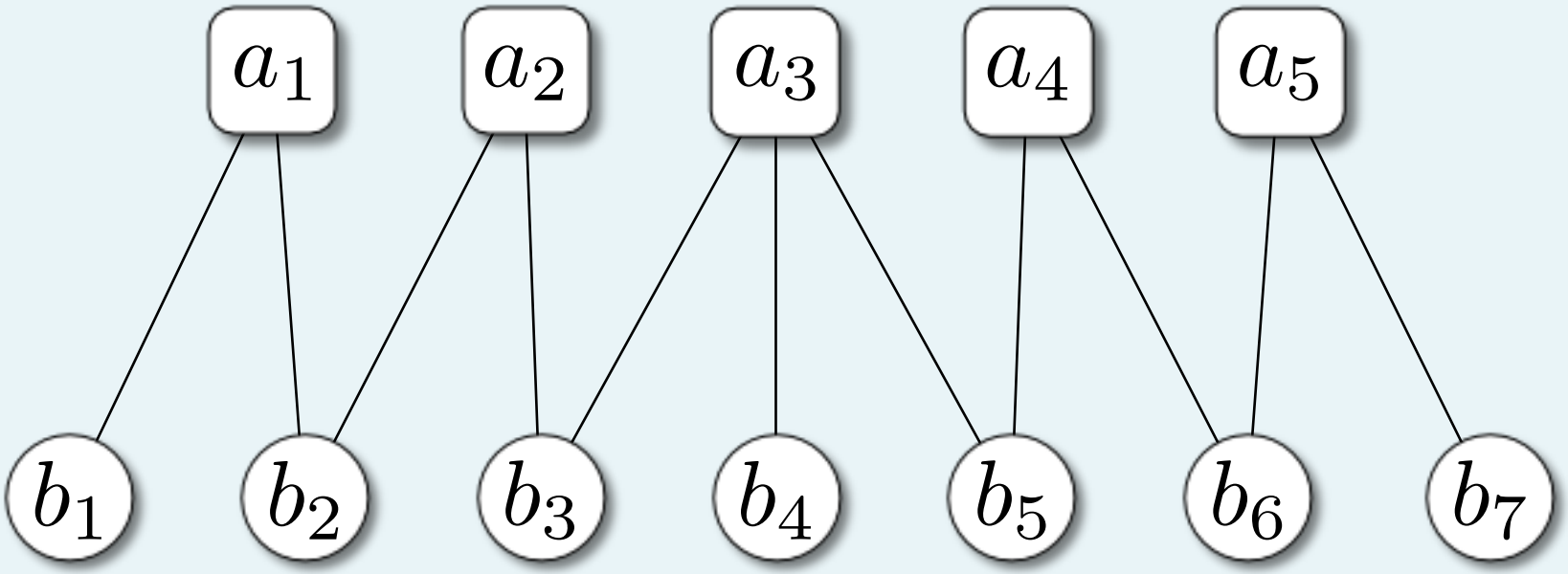
# Références

Bisson, G. and Hussain, F. (2008). Chi-sim : A new similarity measure for the co-clustering task. In *Seventh International Conference on Machine Learning and Applications (ICMLA)*, pages 211–217. IEEE Computer Society.

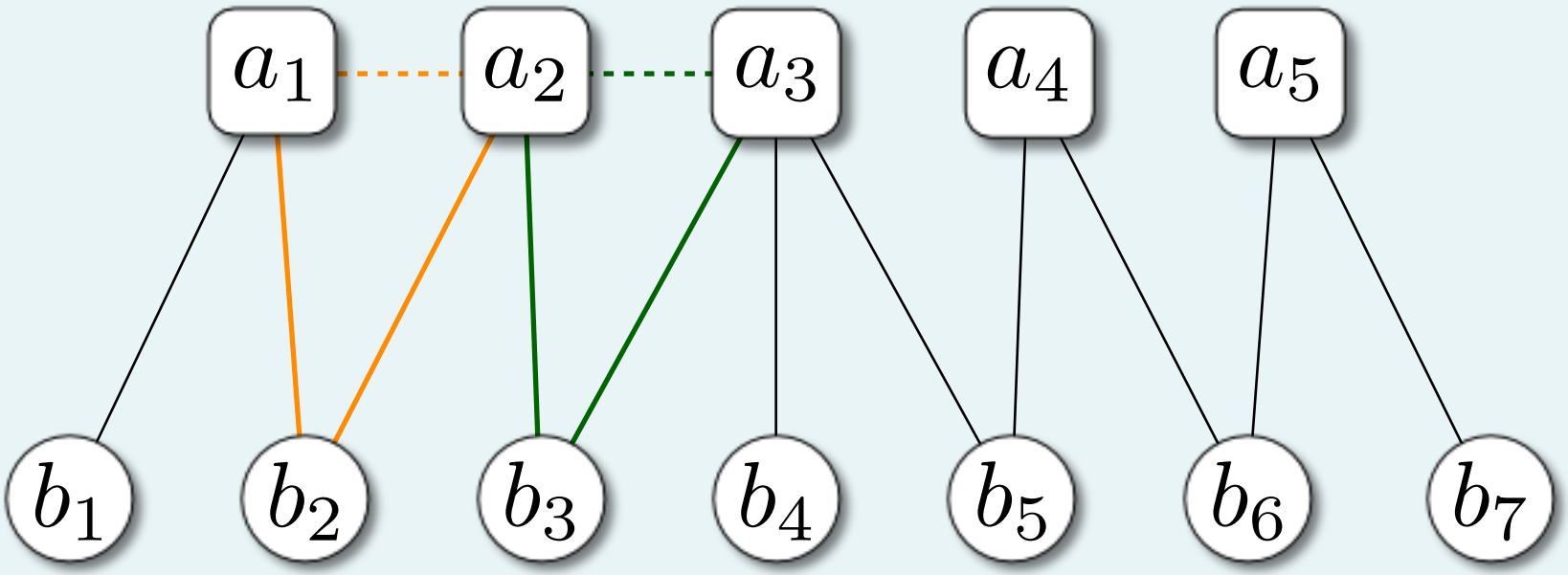
Deerwester, S., Dumais, S. T., Furnas, G. W., Thomas, and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41 :391–407.

Dhillon, I. S., Mallela, S., and Modha, D. S. (2003). Information-theoretic co-clustering. In *Proceedings of The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD-2003)*, pages 89–98.

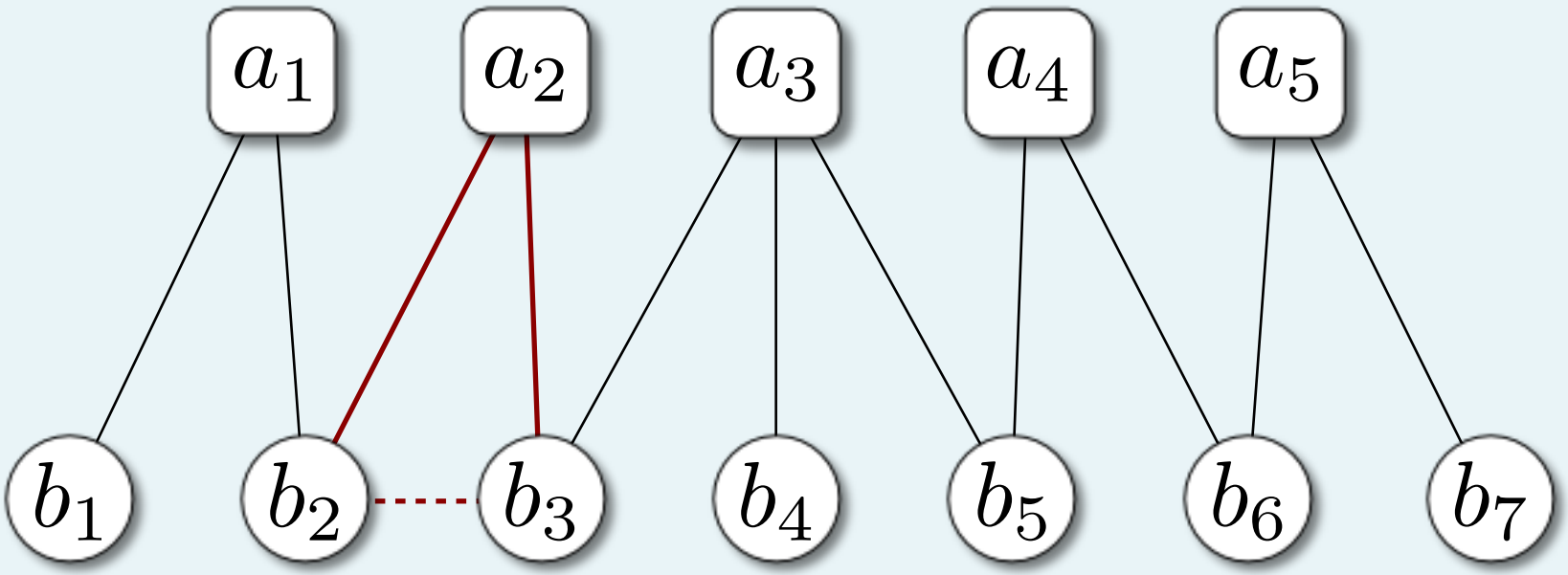
# Graphe bi-partite



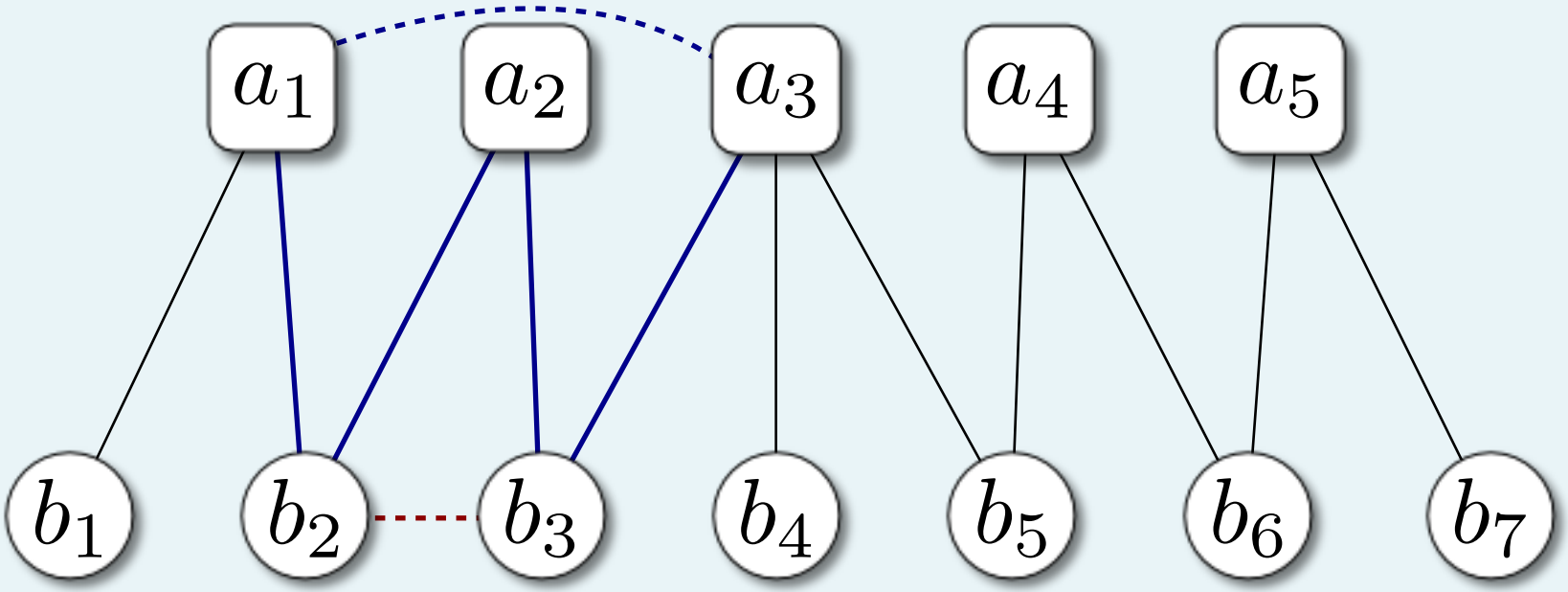
# Graphe bi-partite



# Graphe bi-partite



# Graphe bi-partite



# Méthodes

## Pour LSA

Nous avons fait varier le nombre $k$ de valeurs singulières conservées dans l'intervalle $[10 \ldots 200]$ avec un pas de 10.

## Pour ITCC

Nous avons fait varier pour chaque ensemble de tests le nombre de classes de mots dans la plage de valeur suggérée par Dhillon et al. (2003).
$+$ Trois répétitions pour garder le meilleur résultat.

## Pour Cosinus, LSA et $\chi$-SIM

Utilisation d'un algorithme de Classification Ascendante Hiérarchique (CAH) pour déterminer les groupes en fonction des similarités.