Fondements des couvertures de données : *Couvertures minimales*

M1 - Informatique

Lhouari Nourine, Karima Ennaoui et *Simon Vilmin*.

Institut d'informatique, ISIMA

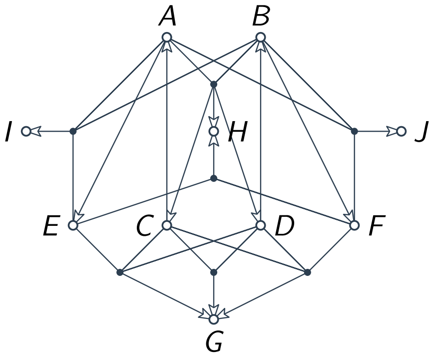
2020-2021

Manipuler des DF

- Dans les épisodes précédents :
  - Σ ensemble de DF ou *couverture*, déduction de nouvelles DF $X \rightarrow Y$ avec l'*implication logique*
  - utilisation des DF pour construire des *relations exemple*
- Deux ensembles *différents* de DF peuvent représenter la *même information* !
- Question : comment choisir entre plusieurs *couvertures* équivalentes ?
  - Privilégier l'expressivité : mettre en avant le *« sens »* des DF , *« si je connais X, je connais Y »*, ou *« si j'ai X, alors j'ai Y »*.
  - Privilégier la concision : trouver une couverture de petite taille, pour améliorer les algorithmes de base de données. C'est une mesure objective.
  - ...
- On va chercher la *concision*, mais (essayer) d'utiliser l'expressivité pour comprendre.

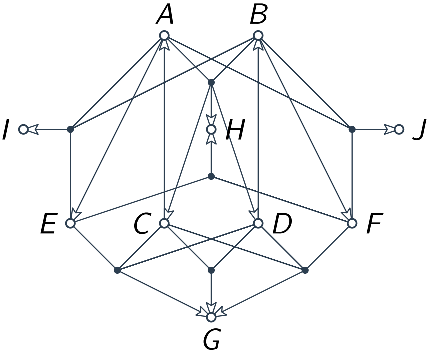
Exemple

- $\mathrm{R} = \{A, B, C, D, E, F, G, H, I, J\}$
- $\Sigma = \{C \to A, A \to E, D \to B, B \to F, AB \to CDH, CD \to G, EF \to H, ABE \to I, ABF \to J, ECD \to G, CDF \to G\}$
- C'est *chargé* ... On va faire le *ménage* !



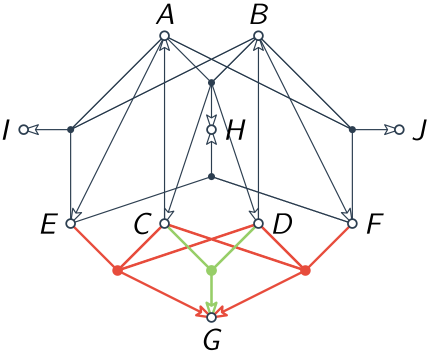
Exemple : ôter les DF inutiles

- Idée : si on peut déduire une DF à partir des autres, quel est son intérêt ?



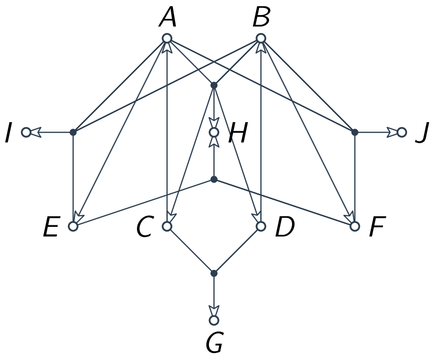
Exemple : ôter les DF inutiles

- on peut déduire $ECD \longrightarrow G$ et $CDF \longrightarrow G$ à partir de $CD \longrightarrow G$: elles sont *inutiles* !



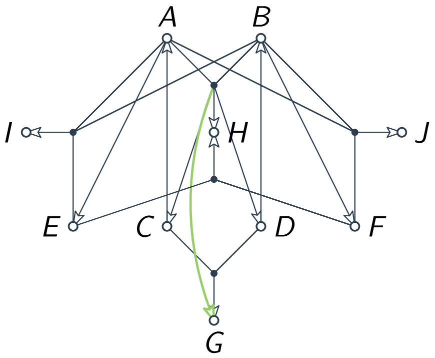
Exemple : ôter les DF inutiles

- À priori, plus rien est inutile. Par ex, si on enlève $CD \rightarrow G$, elle ne sera plus vraie.



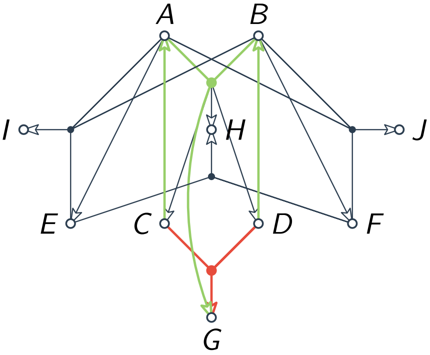
Exemple : ôter les DF inutiles

- Cela dit, on peut *renforcer* $AB \rightarrow CDH$ en lui ajoutant G : $AB \rightarrow CDHG$ puisque $AB \rightarrow CD \rightarrow G$



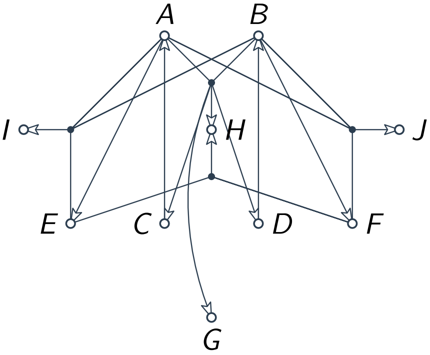
Exemple : ôter les DF inutiles

- Du coup, $CD \longrightarrow G$ devient inutile, on peut utiliser $C \longrightarrow A$, $D \longrightarrow B$ et $AB \longrightarrow G$ pour la déduire !



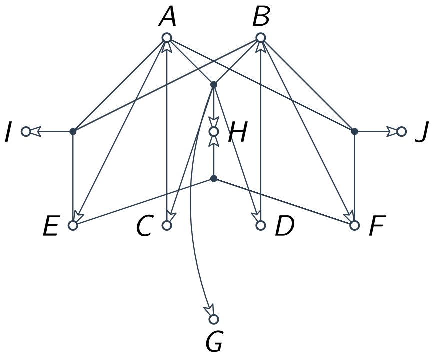
Exemple : ôter les DF inutiles

- Le Σ que l'on obtient n'a perdu aucune information.



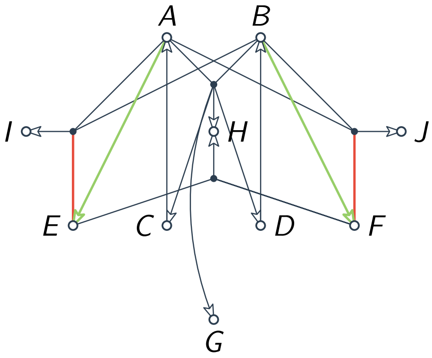
Exemple : nettoyer les attributs

- Idée 1 : dans une DF $X \to Y$, certains attributs ne sont pas essentiels à X pour impliquer Y



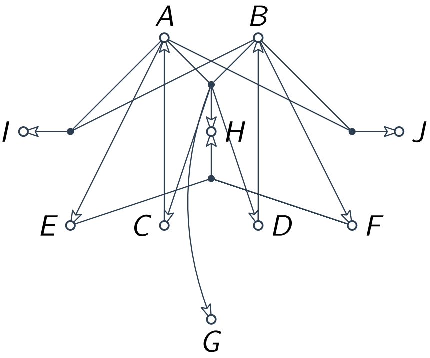
Exemple : nettoyer les attributs

- Par exemple, E est inutile dans $ABE \longrightarrow I$ puisque $A \longrightarrow E$. De même pour F dans $ABF \longrightarrow J$ avec $B \longrightarrow F$.



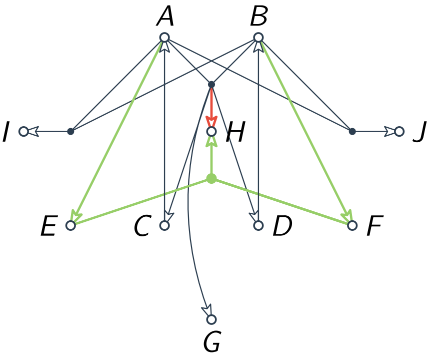
Exemple : nettoyer les attributs

- Idée 2 : on peut réduire une conclusion Y en jouant avec la transitivité.



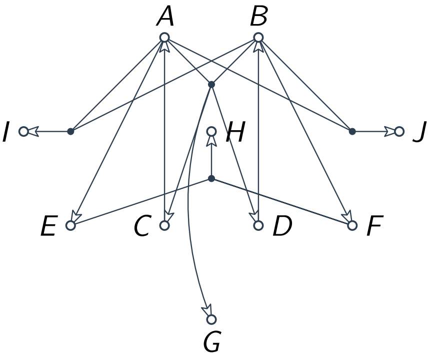
Exemple : nettoyer les attributs

- Par exemple, puisque $A \longrightarrow E$, $B \longrightarrow F$ et $EF \longrightarrow H$, avoir H dans la DF $AB \longrightarrow CDH$ n'apporte rien !



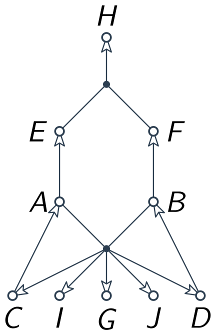
Exemple : nettoyer les attributs

- On finit avec $\Sigma = \{AB \to I, AB \to J, EF \to H, AB \to CDG, C \to A, A \to E, D \to B, B \to F\}$



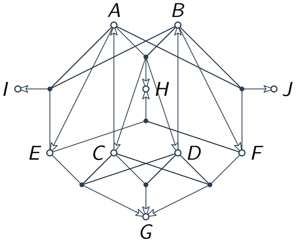
Exemple : nettoyer les attributs

- En *agrégeant* les DF on a
  $\Sigma = \{AB \rightarrow CDGIJ, EF \rightarrow H, C \rightarrow A, A \rightarrow E, D \rightarrow B, B \rightarrow F\}$



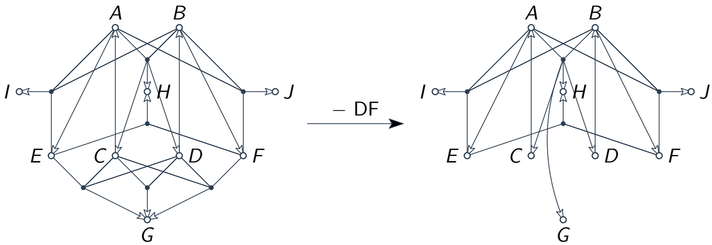
Exemple : résumé

- On part de Σ



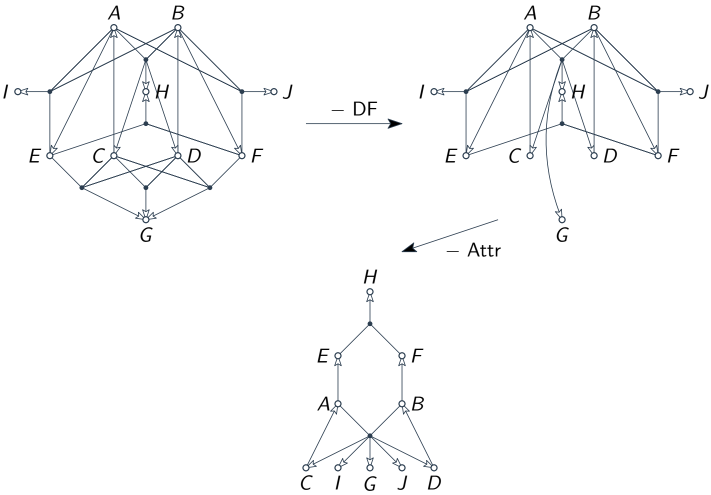
Exemple : résumé

- On enlève les DF inutiles



Exemple : résumé

- On épure les DF restantes



Équivalence

- Soient Σ_1, Σ_2 deux couvertures ($=$ deux ensembles de DF),

Équivalence

- Soient Σ_1, Σ_2 deux couvertures ($=$ deux ensembles de DF),
- Si on peut déduire toutes les DF de Σ_2 depuis Σ_1 on note $\Sigma_1 \models \Sigma_2$: l'information contenue dans Σ_2 est *aussi* contenue dans Σ_1.

Équivalence

- Soient Σ_1, Σ_2 deux couvertures (= deux ensembles de DF),
- Si on peut déduire toutes les DF de Σ_2 depuis Σ_1 on note $\Sigma_1 \models \Sigma_2$: l'information contenue dans Σ_2 est *aussi* contenue dans Σ_1.
- Si de plus $\Sigma_2 \models \Sigma_1$, alors l'information contenue dans Σ_1 *est la même* que celle contenue dans Σ_2 : les deux couvertures sont *équivalentes*.

Équivalence

- Soient Σ_1, Σ_2 deux couvertures (= deux ensembles de DF),
- Si on peut déduire toutes les DF de Σ_2 depuis Σ_1 on note $\Sigma_1 \models \Sigma_2$: l'information contenue dans Σ_2 est *aussi* contenue dans Σ_1.
- Si de plus $\Sigma_2 \models \Sigma_1$, alors l'information contenue dans Σ_1 *est la même* que celle contenue dans Σ_2 : les deux couvertures sont *équivalentes*.

Définition - Équivalence

Deux ensembles de DF Σ_1 et Σ_2 sont *équivalents*, noté $\Sigma_1 \equiv \Sigma_2$, si $\Sigma_1 \models \Sigma_2$ et $\Sigma_2 \models \Sigma_1$.

- Parmi ces trois couvertures, lesquelles sont équivalentes ?
  - $\Sigma_1 = \{AB \rightarrow C, BC \rightarrow A, ABD \rightarrow E\}$
  - $\Sigma_2 = \{AB \rightarrow C, BC \rightarrow A, ACD \rightarrow E\}$
  - $\Sigma_3 = \{AB \rightarrow C, BC \rightarrow A, BCD \rightarrow E\}$

Équivalence

- Soient Σ_1, Σ_2 deux couvertures (= deux ensembles de DF),
- Si on peut déduire toutes les DF de Σ_2 depuis Σ_1 on note $\Sigma_1 \models \Sigma_2$: l'information contenue dans Σ_2 est *aussi* contenue dans Σ_1.
- Si de plus $\Sigma_2 \models \Sigma_1$, alors l'information contenue dans Σ_1 *est la même* que celle contenue dans Σ_2 : les deux couvertures sont *équivalentes*.

Définition - Équivalence

Deux ensembles de DF Σ_1 et Σ_2 sont *équivalents*, noté $\Sigma_1 \equiv \Sigma_2$, si $\Sigma_1 \models \Sigma_2$ et $\Sigma_2 \models \Sigma_1$.

- Parmi ces trois couvertures, lesquelles sont équivalentes ?
  - ✗ $\Sigma_1 = \{AB \rightarrow C, BC \rightarrow A, ABD \rightarrow E\}$
  - ✓ $\Sigma_2 = \{AB \rightarrow C, BC \rightarrow A, ACD \rightarrow E\}$
  - ✗ $\Sigma_3 = \{AB \rightarrow C, BC \rightarrow A, BCD \rightarrow E\}$

Problème de concision : Mesures

- On a plein de couvertures $\Sigma_1, \ldots, \Sigma_n$ équivalentes : quelle est la meilleure (plus concise) ?
- Plein de mesures différentes !
  - le nombre de DF, $|\,\Sigma\,|$,
  - la taille des prémisses, $\|\,\Sigma\,\|_l = \sum_{X \to Y \in \Sigma} |X|$ (l pour left),
  - la taille des conclusions, $\|\,\Sigma\,\|_r = \sum_{X \to Y \in \Sigma} |Y|$ (r pour right),
  - la taille totale de Σ, $\|\,\Sigma\,\| = \|\,\Sigma\,\|_l + \|\,\Sigma\,\|_r$.
- Exemple introductif réduit : $\Sigma = \{AB \to CDGIJ, EF \to H, C \to A, A \to E, D \to B, B \to F\}$
  - $|\,\Sigma\,| = 6$, $\|\,\Sigma\,\|_l = 8$, $\|\,\Sigma\,\|_r = 10$ et $\|\,\Sigma\,\| = 18$.

Problème de concision : Minimalité

- Les mesures admettent forcément un *minimum* puisque numériques.
- Les « meilleures couvertures » sont donc celles qui atteignent ce minimum

Définition - couvertures minimum, optimum

Soit Σ un ensemble de DF. On dit que Σ est :

- *minimum* si quelque soit $\Sigma' \equiv \Sigma$, $|\Sigma| \leq |\Sigma'|$,
- *optimum à gauche* si quelque soit $\Sigma' \equiv \Sigma$, $\|\Sigma\|_l \leq \|\Sigma'\|_l$,
- *optimum à droite* si quelque soit $\Sigma' \equiv \Sigma$, $\|\Sigma\|_r \leq \|\Sigma'\|_r$,
- *optimum* si quelque soit $\Sigma' \equiv \Sigma$, $\|\Sigma\| \leq \|\Sigma'\|$.

Problème de concision : Énoncé

- Soit Σ une couverture sur R et s une des quatre mesures définies précédemment : $|\cdot|$, $\|\cdot\|_l$, $\|\cdot\|_r$, $\|\cdot\|$.

- Question : peut-on trouver une couverture Σ' équivalente à Σ qui minimise s en temps $poly(|\Sigma|, |R|)$?

- C'est-à-dire, trouver Σ' telle que $s(\Sigma') \leq s(\Sigma'')$ pour tout $\Sigma'' \equiv \Sigma$, depuis Σ.

Problème de concision : Énoncé

- Soit Σ une couverture sur R et s une des quatre mesures définies précédemment : $|\cdot|$, $\|\cdot\|_l$, $\|\cdot\|_r$, $\|\cdot\|$.

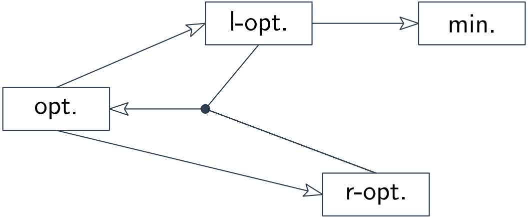
- Question : peut-on trouver une couverture Σ' équivalente à Σ qui minimise s en temps $poly(|\Sigma|, |\mathrm{R}|)$?

- C'est-à-dire, trouver Σ' telle que $s(\Sigma') \leq s(\Sigma'')$ pour tout $\Sigma'' \equiv \Sigma$, depuis Σ.

ET LA RÉPONSE EST ... ça dépend de s ...

Problème de concision : Relation

- Voilà quelques relations que l'on trouve dans la littérature [Ausiello, 1986], [Wild, 2017].
- Par exemple, si Σ est *optimum* à gauche, alors elle est aussi *minimum*.



Redondance

- Précisons la notion de « *DF inutile* ».
- Une DF est *inutile* dans Σ si on peut la retirer de Σ et la retrouver à partir des DF restantes.
- Elle apporte une information déjà contenue dans Σ : elle est *redondante*.

Définition - Redondance

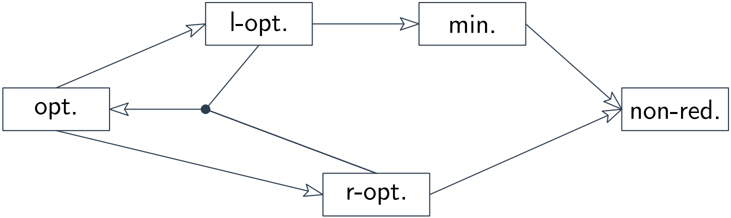
Soit Σ une couverture et $X \rightarrow Y \in \Sigma$. On dit que $X \rightarrow Y$ est *redondante* dans Σ si $\Sigma \setminus \{X \rightarrow Y\} \models X \rightarrow Y$. Si Σ ne contient pas de DF redondante, c'est une couverture *non-redondante*.

- Exemple si $\Sigma = \{A \rightarrow B, B \rightarrow C, A \rightarrow C\}$
  - $\Sigma \setminus \{A \rightarrow C\} \models A \rightarrow C$. Donc $A \rightarrow C$ est redondante.
  - Par contre $\Sigma \setminus \{A \rightarrow C\}$ est non-redondante.

Propriétés

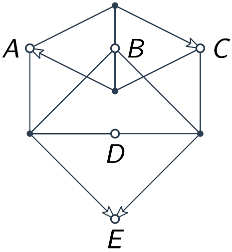
Propriété : Une couverture minimum est non-redondante.

Propriété : Une couverture optimum à droite est non-redondante.



Propriétés

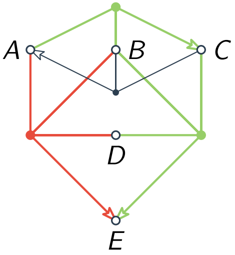
- Soit $\Sigma = \{AB \rightarrow C, BC \rightarrow A, ABD \rightarrow E, BCD \rightarrow E\}$.
-



-

Propriétés

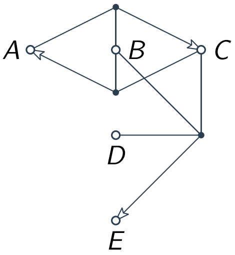
- Soit $\Sigma = \{AB \rightarrow C, BC \rightarrow A, ABD \rightarrow E, BCD \rightarrow E\}$.
-



- $ABD \rightarrow E$ peut être déduite de $AB \rightarrow C$ et $BCD \rightarrow E$, donc $\Sigma \setminus \{ABD \rightarrow E\} \models ABD \rightarrow E$.

Propriétés

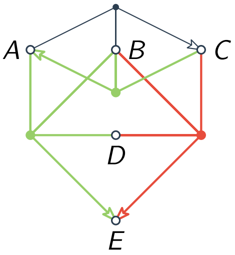
- Soit $\Sigma = \{AB \rightarrow C, BC \rightarrow A, ABD \rightarrow E, BCD \rightarrow E\}$.
-



- $ABD \rightarrow E$ peut être déduite de $AB \rightarrow C$ et $BCD \rightarrow E$, donc $\Sigma \setminus \{ABD \rightarrow E\} \models ABD \rightarrow E$.

Propriétés

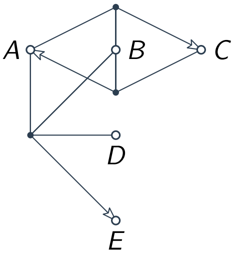
- Soit $\Sigma = \{AB \to C, BC \to A, ABD \to E, BCD \to E\}$.
-



- symétriquement, $BCD \to E$ est déduite de $BC \to A$, $ABD \to E$, donc $\Sigma \setminus \{BCD \to E\} \models BCD \to E$.

Propriétés

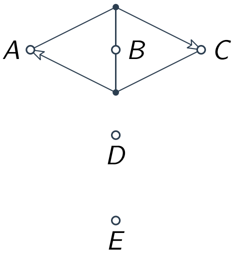
- Soit $\Sigma = \{AB \to C, BC \to A, ABD \to E, BCD \to E\}$.
-



- symétriquement, $BCD \to E$ est déduite de $BC \to A$, $ABD \to E$, donc $\Sigma \setminus \{BCD \to E\} \models BCD \to E$.

Propriétés

- Soit $\Sigma = \{AB \to C, BC \to A, ABD \to E, BCD \to E\}$.
- $ABD \to E$ et $BCD \to E$ sont « mutuellement redondantes »



- Par contre on ne peut pas enlever les deux à la fois car $\Sigma \setminus \{ABD \to E, BCD \to E\} \not\models \{ABD \to E, BCD \to E\}$.

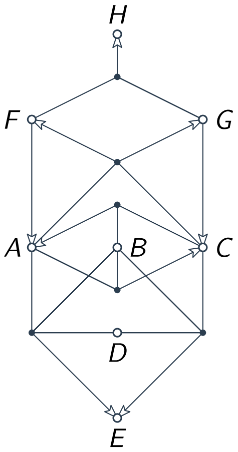
Algorithme de non-redondance

Algorithme REDONDANT(Σ)

Data: Σ un ensemble de DF sur R
Result: Σ sans DF redondantes

foreach $X \rightarrow Y \in \Sigma$ **do**
  $\Sigma = \Sigma \setminus \{X \rightarrow Y\}$;
  if $\Sigma \not\models X \rightarrow Y$ **then**
    $\Sigma = \Sigma \cup \{X \rightarrow Y\}$
  end
end

$$\Sigma = \begin{bmatrix} FG \rightarrow H & BC \rightarrow A \\ F \rightarrow A & AB \rightarrow C \\ G \rightarrow C & ABD \rightarrow E \\ AC \rightarrow FG & BCD \rightarrow E \end{bmatrix}$$



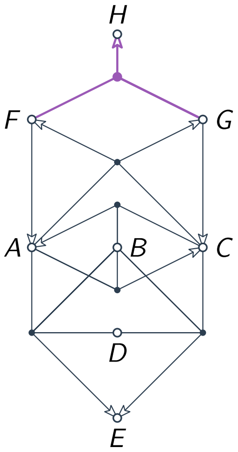
Algorithme de non-redondance

Algorithme REDONDANT(Σ)

Data: Σ un ensemble de DF sur R
Result: Σ sans DF redondantes

foreach $X \rightarrow Y \in \Sigma$ **do**
  $\Sigma = \Sigma \setminus \{X \rightarrow Y\}$;
  if $\Sigma \not\models X \rightarrow Y$ **then**
    $\Sigma = \Sigma \cup \{X \rightarrow Y\}$
  end
end

$$\Sigma = \begin{bmatrix} FG \rightarrow H & BC \rightarrow A \\ F \rightarrow A & AB \rightarrow C \\ G \rightarrow C & ABD \rightarrow E \\ AC \rightarrow FG & BCD \rightarrow E \end{bmatrix}$$



Algorithme de non-redondance

Algorithme REDONDANT(Σ)

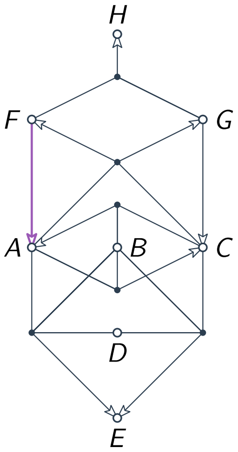
Data: Σ un ensemble de DF sur R

Result: Σ sans DF redondantes

foreach $X \to Y \in \Sigma$ **do**
- $\Sigma = \Sigma \setminus \{X \to Y\}$;
- **if** $\Sigma \not\models X \to Y$ **then**
  - $\Sigma = \Sigma \cup \{X \to Y\}$
- **end**

end

$$\Sigma = \begin{bmatrix} FG \to H & BC \to A \\ F \to A & AB \to C \\ G \to C & ABD \to E \\ AC \to FG & BCD \to E \end{bmatrix}$$



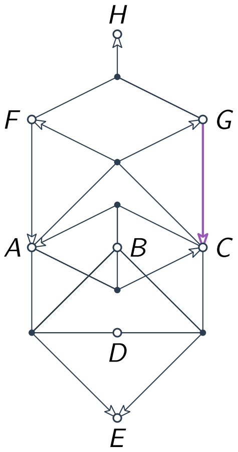
Algorithme de non-redondance

Algorithme REDONDANT(Σ)

Data: Σ un ensemble de DF sur R
Result: Σ sans DF redondantes

foreach $X \to Y \in \Sigma$ **do**
  $\Sigma = \Sigma \setminus \{X \to Y\}$;
  if $\Sigma \not\models X \to Y$ **then**
    $\Sigma = \Sigma \cup \{X \to Y\}$
  end
end

$$\Sigma = \begin{bmatrix} FG \to H & BC \to A \\ F \to A & AB \to C \\ G \to C & ABD \to E \\ AC \to FG & BCD \to E \end{bmatrix}$$



Algorithme de non-redondance

Algorithme REDONDANT(Σ)

Data: Σ un ensemble de DF sur R

Result: Σ sans DF redondantes

foreach $X \to Y \in \Sigma$ **do**

  $\Sigma = \Sigma \setminus \{X \to Y\}$;

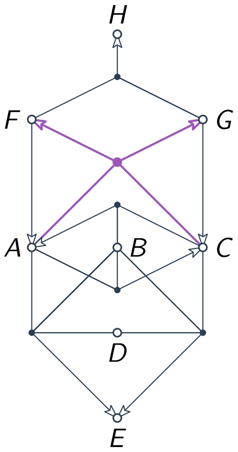
  if $\Sigma \not\models X \to Y$ **then**

    $\Sigma = \Sigma \cup \{X \to Y\}$

  end

end

$$\Sigma = \begin{bmatrix} FG \to H & BC \to A \\ F \to A & AB \to C \\ G \to C & ABD \to E \\ AC \to FG & BCD \to E \end{bmatrix}$$



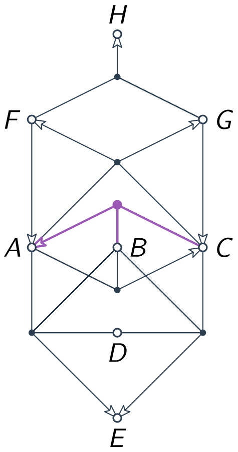
Algorithme de non-redondance

Algorithme REDONDANT(Σ)

Data: Σ un ensemble de DF sur R
Result: Σ sans DF redondantes

foreach $X \to Y \in \Sigma$ **do**
  $\Sigma = \Sigma \setminus \{X \to Y\}$;
  if $\Sigma \not\models X \to Y$ **then**
    $\Sigma = \Sigma \cup \{X \to Y\}$
  end
end

$$\Sigma = \begin{bmatrix} FG \to H & BC \to A \\ F \to A & AB \to C \\ G \to C & ABD \to E \\ AC \to FG & BCD \to E \end{bmatrix}$$



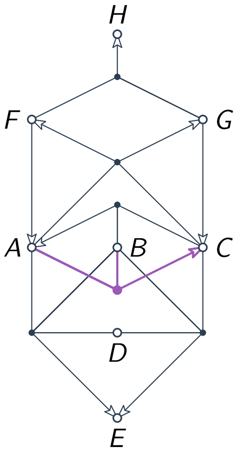
Algorithme de non-redondance

Algorithme REDONDANT(Σ)

Data: Σ un ensemble de DF sur R
Result: Σ sans DF redondantes

foreach $X \to Y \in \Sigma$ **do**
  $\Sigma = \Sigma \setminus \{X \to Y\}$;
  if $\Sigma \not\models X \to Y$ **then**
    $\Sigma = \Sigma \cup \{X \to Y\}$
  end
end

$$\Sigma = \begin{bmatrix} FG \to H & BC \to A \\ F \to A & AB \to C \\ G \to C & ABD \to E \\ AC \to FG & BCD \to E \end{bmatrix}$$



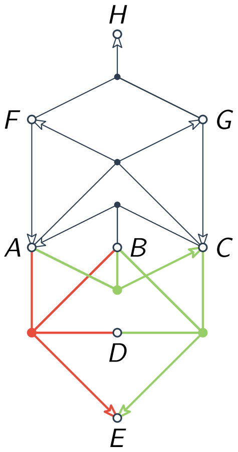
Algorithme de non-redondance

Algorithme REDONDANT(Σ)

Data: Σ un ensemble de DF sur R
Result: Σ sans DF redondantes

foreach $X \rightarrow Y \in \Sigma$ **do**
  $\Sigma = \Sigma \setminus \{X \rightarrow Y\}$;
  if $\Sigma \not\models X \rightarrow Y$ **then**
    $\Sigma = \Sigma \cup \{X \rightarrow Y\}$
  end
end

$$\Sigma = \begin{bmatrix} FG \rightarrow H & BC \rightarrow A \\ F \rightarrow A & AB \rightarrow C \\ G \rightarrow C & ABD \rightarrow E \\ AC \rightarrow FG & BCD \rightarrow E \end{bmatrix}$$



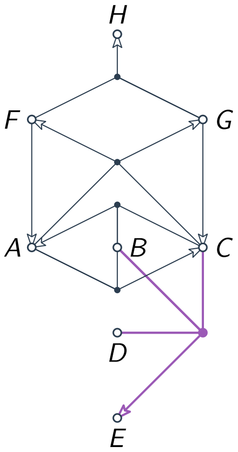
Algorithme de non-redondance

Algorithme REDONDANT(Σ)

Data: Σ un ensemble de DF sur R
Result: Σ sans DF redondantes

foreach $X \rightarrow Y \in \Sigma$ **do**
  $\Sigma = \Sigma \setminus \{X \rightarrow Y\}$;
  if $\Sigma \not\models X \rightarrow Y$ **then**
    $\Sigma = \Sigma \cup \{X \rightarrow Y\}$
  end
end

$$\Sigma = \begin{bmatrix} FG \rightarrow H & BC \rightarrow A \\ F \rightarrow A & AB \rightarrow C \\ G \rightarrow C & \\ AC \rightarrow FG & BCD \rightarrow E \end{bmatrix}$$



Algorithme de non-redondance

Algorithme REDONDANT(Σ)

Data: Σ un ensemble de DF sur R

Result: Σ sans DF redondantes

foreach $X \to Y \in \Sigma$ **do**

  $\Sigma = \Sigma \setminus \{X \to Y\}$;

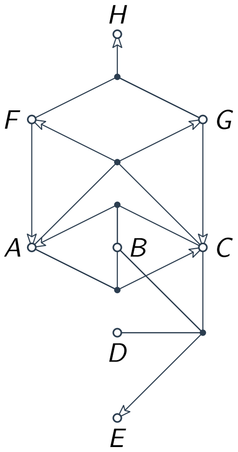
  if $\Sigma \not\models X \to Y$ **then**

    $\Sigma = \Sigma \cup \{X \to Y\}$

  end

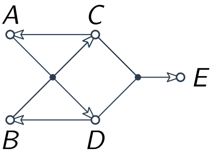
end

$$\Sigma = \begin{bmatrix} FG \to H & BC \to A \\ F \to A & AB \to C \\ G \to C & \\ AC \to FG & BCD \to E \end{bmatrix}$$



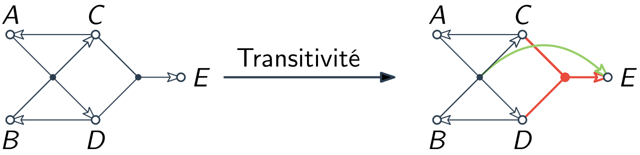
A-t-on une couverture minimum ?

- Dans le « graphe des DF », une DF $X \rightarrow Y$ est redondante s'il y a un chemin qui part de X et qui arrive à tous les éléments de Y
- $\approx X \rightarrow Y$ marque la *transitivité* !
- *Problème* : si une transitivité n'est pas marquée, peut-on oublier de la redondance « cachée » ?



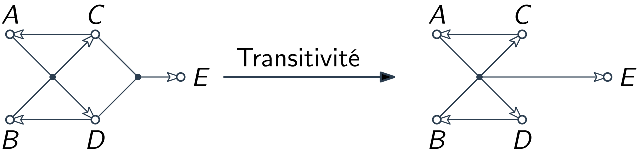
A-t-on une couverture minimum ?

- Dans le « graphe des DF », une DF $X \rightarrow Y$ est redondante s'il y a un chemin qui part de X et qui arrive à tous les éléments de Y
- $\approx X \rightarrow Y$ marque la *transitivité* !
- *Problème* : si une transitivité n'est pas marquée, peut-on oublier de la redondance « cachée » ?



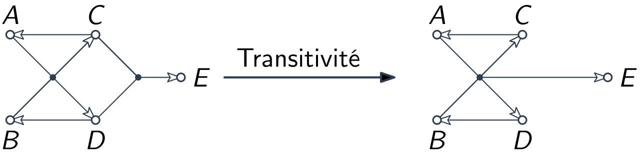
A-t-on une couverture minimum ?

- Dans le « graphe des DF », une DF $X \rightarrow Y$ est redondante s'il y a un chemin qui part de X et qui arrive à tous les éléments de Y
- $\approx X \rightarrow Y$ marque la *transitivité* !
- *Problème* : si une transitivité n'est pas marquée, peut-on oublier de la redondance « cachée » ? *OUI ...*



A-t-on une couverture minimum ?

- Dans le « graphe des DF », une DF $X \rightarrow Y$ est redondante s'il y a un chemin qui part de X et qui arrive à tous les éléments de Y
- $\approx X \rightarrow Y$ marque la *transitivité* !
- *Problème* : si une transitivité n'est pas marquée, peut-on oublier de la redondance « cachée » ? *OUI ...*



- Conséquence : une couverture *non-redondante* n'est *pas forcément minimum* !

Comment faire ?

- Pour révéler la redondance cachée, on *ferme* chaque DF, $X \to Y$ devient $X \to X^{\Sigma}$ (ou $X \to X^{\Sigma} \setminus X$).
- $\approx$ on « rend explicite » toute l'information que l'on peut avoir depuis $X \implies$ marquage des transitivités.
- Si toutes les DF de Σ sont fermées, on dit qu'elle est *fermée à droite*.
- Question : est-ce suffisant pour éliminer toute redondance ?

Comment faire ?

- Pour révéler la redondance cachée, on *ferme* chaque DF, $X \rightarrow Y$ devient $X \rightarrow X^{\Sigma}$ (ou $X \rightarrow X^{\Sigma} \setminus X$).
- $\approx$ on « rend explicite » toute l'information que l'on peut avoir depuis $X \implies$ marquage des transitivités.
- Si toutes les DF de Σ sont fermées, on dit qu'elle est *fermée à droite*.
- Question : est-ce suffisant pour éliminer toute redondance ?

OUI ! ! !

Théorème [Shock, 1986] : Une couverture fermée à droite et non-redondante est minimum.

Comment faire ?

- Pour révéler la redondance cachée, on *ferme* chaque DF, $X \to Y$ devient $X \to X^{\Sigma}$ (ou $X \to X^{\Sigma} \setminus X$).
- $\approx$ on « rend explicite » toute l'information que l'on peut avoir depuis $X \implies$ marquage des transitivités.
- Si toutes les DF de Σ sont fermées, on dit qu'elle est *fermée à droite*.
- Question : est-ce suffisant pour éliminer toute redondance ?

OUI ! ! !

Théorème [Shock, 1986] : Une couverture fermée à droite et non-redondante est minimum.

- En bonus, le théorème donne un algo tout simple :
  1. On ferme Σ à droite
  2. On élimine la redondance.

Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**

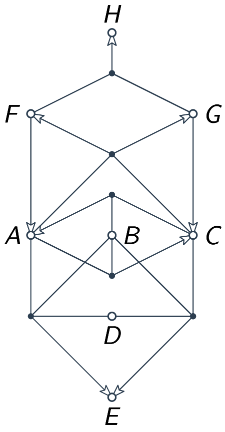
Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 1 : Fermeture à droite

$$\Sigma = \left[\begin{array}{ll} FG \rightarrow H & BC \rightarrow A \\ F \rightarrow A & AB \rightarrow C \\ G \rightarrow C & ABD \rightarrow E \\ AC \rightarrow FG & BCD \rightarrow E \end{array}\right]$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \to Y \in \Sigma$ **do**

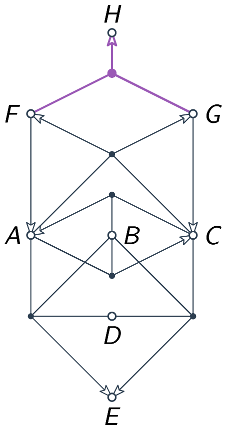
   Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 1 : Fermeture à droite

$$\Sigma = \begin{bmatrix} FG \to H & BC \to A \\ F \to A & AB \to C \\ G \to C & ABD \to E \\ AC \to FG & BCD \to E \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

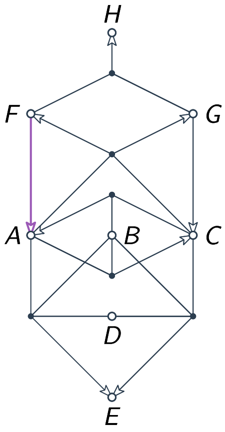
Data: Σ une couverture sur R
Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**
  Remplacer Y par $X^{\Sigma} \setminus X$;
end

REDONDANCE(Σ)

Étape 1 : Fermeture à droite

$$\Sigma = \begin{bmatrix} FG \rightarrow H & BC \rightarrow A \\ F \rightarrow A & AB \rightarrow C \\ G \rightarrow C & ABD \rightarrow E \\ AC \rightarrow FG & BCD \rightarrow E \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**

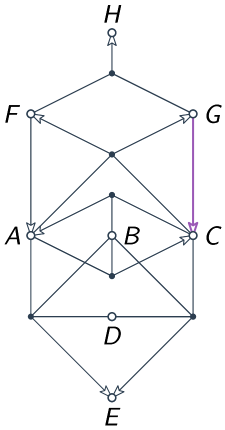
  Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 1 : Fermeture à droite

$$\Sigma = \begin{bmatrix} FG \rightarrow H & BC \rightarrow A \\ F \rightarrow A & AB \rightarrow C \\ G \rightarrow C & ABD \rightarrow E \\ AC \rightarrow FG & BCD \rightarrow E \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**

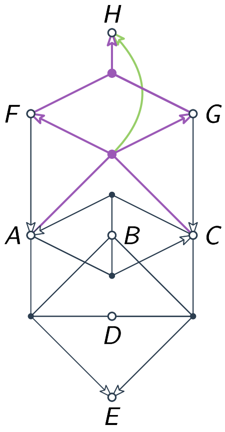
  Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 1 : Fermeture à droite

$$\Sigma = \begin{bmatrix} FG \rightarrow H & BC \rightarrow A \\ F \rightarrow A & AB \rightarrow C \\ G \rightarrow C & ABD \rightarrow E \\ AC \rightarrow FGH & BCD \rightarrow E \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**

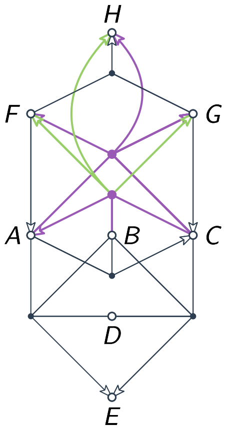
  Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 1 : Fermeture à droite

$$\Sigma = \begin{bmatrix} FG \rightarrow H & BC \rightarrow AFGH \\ F \rightarrow A & AB \rightarrow C \\ G \rightarrow C & ABD \rightarrow E \\ AC \rightarrow FGH & BCD \rightarrow E \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**

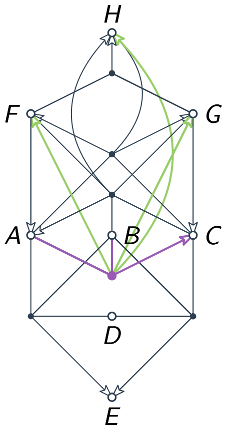
  Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 1 : Fermeture à droite

$$\Sigma = \begin{bmatrix} FG \rightarrow H & BC \rightarrow AFGH \\ F \rightarrow A & AB \rightarrow CFGH \\ G \rightarrow C & ABD \rightarrow E \\ AC \rightarrow FGH & BCD \rightarrow E \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \to Y \in \Sigma$ **do**

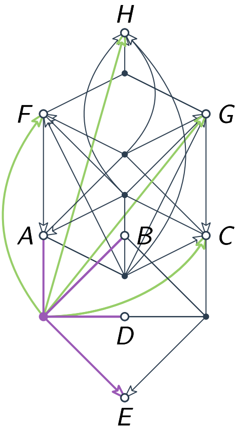
  Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 1 : Fermeture à droite

$$\Sigma = \begin{bmatrix} FG \to H & BC \to AFGH \\ F \to A & AB \to CFGH \\ G \to C & ABD \to ECFGH \\ AC \to FGH & BCD \to E \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

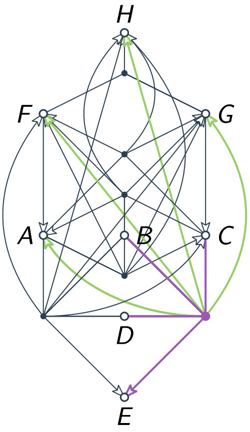
Data: Σ une couverture sur R
Result: Σ transformée en couverture minimum

foreach $X \to Y \in \Sigma$ **do**
  Remplacer Y par $X^{\Sigma} \setminus X$;
end

REDONDANCE(Σ)

Étape 1 : Fermeture à droite

$$\Sigma = \begin{bmatrix} FG \to H & BC \to AFGH \\ F \to A & AB \to CFGH \\ G \to C & ABD \to ECFGH \\ AC \to FGH & BCD \to EAFGH \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**

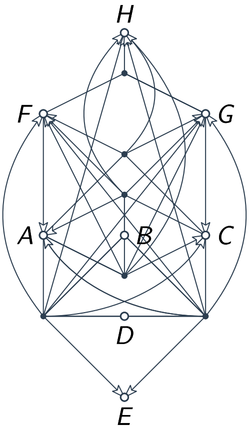
  Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 1 : Fermeture à droite

$$\Sigma = \begin{bmatrix} FG \rightarrow H & BC \rightarrow AFGH \\ F \rightarrow A & AB \rightarrow CFGH \\ G \rightarrow C & ABD \rightarrow ECFGH \\ AC \rightarrow FGH & BCD \rightarrow EAFGH \end{bmatrix}$$



Algorithme de minimalité

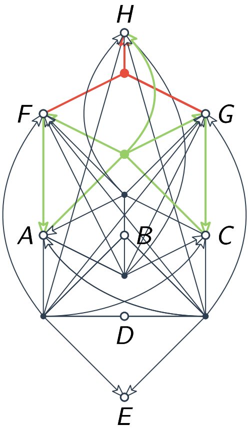
Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R
Result: Σ transformée en couverture minimum

foreach $X \to Y \in \Sigma$ **do**
  Remplacer Y par $X^{\Sigma} \setminus X$;
end
REDONDANCE(Σ)

Étape 2 : Enlever la redondance

$$\Sigma = \begin{bmatrix} FG \to H & BC \to AFGH \\ F \to A & AB \to CFGH \\ G \to C & ABD \to ECFGH \\ AC \to FGH & BCD \to EAFGH \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**

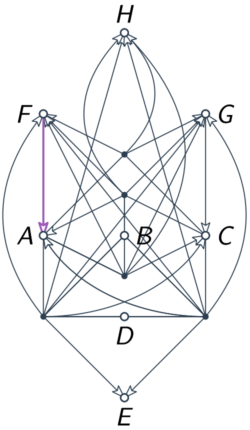
  Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 2 : Enlever la redondance

$$\Sigma = \begin{bmatrix} & BC \rightarrow AFGH \\ F \rightarrow A & AB \rightarrow CFGH \\ G \rightarrow C & ABD \rightarrow ECFGH \\ AC \rightarrow FGH & BCD \rightarrow EAFGH \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \to Y \in \Sigma$ **do**

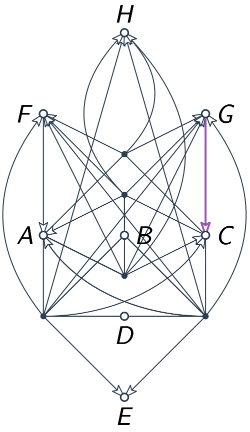
  Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 2 : Enlever la redondance

$$\Sigma = \begin{bmatrix} & BC \to AFGH \\ F \to A & AB \to CFGH \\ G \to C & ABD \to ECFGH \\ AC \to FGH & BCD \to EAFGH \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

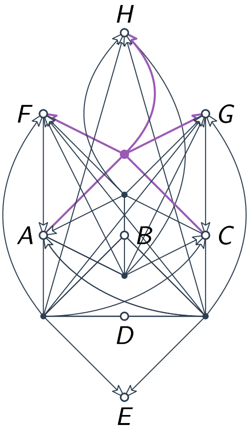
Data: Σ une couverture sur R
Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**
  Remplacer Y par $X^{\Sigma} \setminus X$;
end

REDONDANCE(Σ)

Étape 2 : Enlever la redondance

$$\Sigma = \left[\begin{array}{ll} & BC \rightarrow AFGH \\ F \rightarrow A & AB \rightarrow CFGH \\ G \rightarrow C & ABD \rightarrow ECFGH \\ AC \rightarrow FGH & BCD \rightarrow EAFGH \end{array}\right]$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \to Y \in \Sigma$ **do**

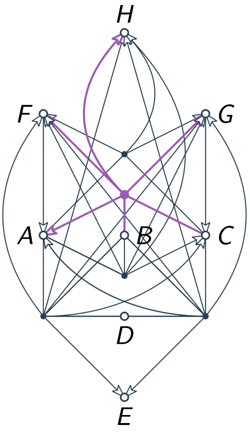
Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 2 : Enlever la redondance

$$\Sigma = \begin{bmatrix} & BC \to AFGH \\ F \to A & AB \to CFGH \\ G \to C & ABD \to ECFGH \\ AC \to FGH & BCD \to EAFGH \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**

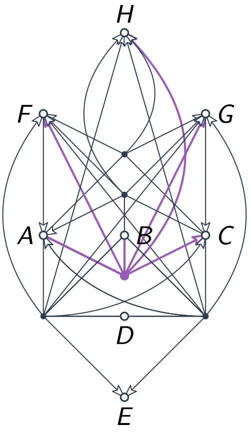
  Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 2 : Enlever la redondance

$$\Sigma = \begin{bmatrix} & BC \rightarrow AFGH \\ F \rightarrow A & AB \rightarrow CFGH \\ G \rightarrow C & ABD \rightarrow ECFGH \\ AC \rightarrow FGH & BCD \rightarrow EAFGH \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**

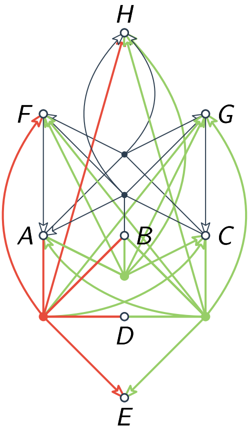
  Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 2 : Enlever la redondance

$$\Sigma = \begin{bmatrix} & BC \rightarrow AFGH \\ F \rightarrow A & AB \rightarrow CFGH \\ G \rightarrow C & ABD \rightarrow ECFGH \\ AC \rightarrow FGH & BCD \rightarrow EAFGH \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

Data: Σ une couverture sur R

Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**

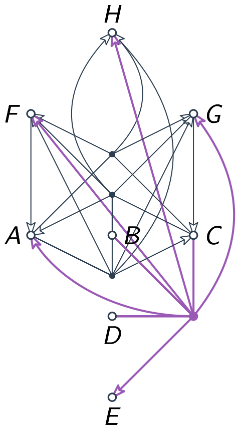
  Remplacer Y par $X^{\Sigma} \setminus X$;

end

REDONDANCE(Σ)

Étape 2 : Enlever la redondance

$$\Sigma = \begin{bmatrix} & BC \rightarrow AFGH \\ F \rightarrow A & AB \rightarrow CFGH \\ G \rightarrow C & \\ AC \rightarrow FGH & BCD \rightarrow EAFGH \end{bmatrix}$$



Algorithme de minimalité

Algorithme MINIMUM(Σ)

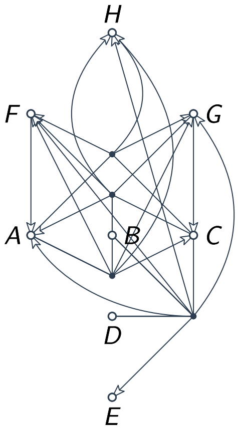
Data: Σ une couverture sur R
Result: Σ transformée en couverture minimum

foreach $X \rightarrow Y \in \Sigma$ **do**
  Remplacer Y par $X^{\Sigma} \setminus X$;
end

REDONDANCE(Σ)

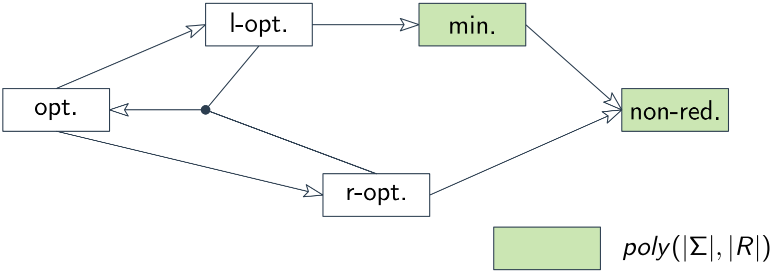
Étape 2 : Enlever la redondance

$$\Sigma = \begin{bmatrix} & BC \rightarrow AFGH \\ F \rightarrow A & AB \rightarrow CFGH \\ G \rightarrow C & \\ AC \rightarrow FGH & BCD \rightarrow EAFGH \end{bmatrix}$$



Retour sur les mesures

- Bonne nouvelle : les deux algos sont $poly(|\Sigma|, |R|)$.

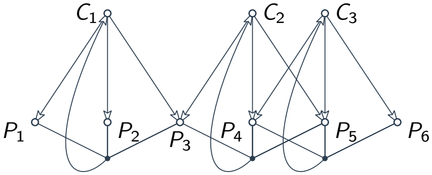


DF champêtres

- Une forêt dispose de 3 « coins à champignons » C_1, C_2, C_3.

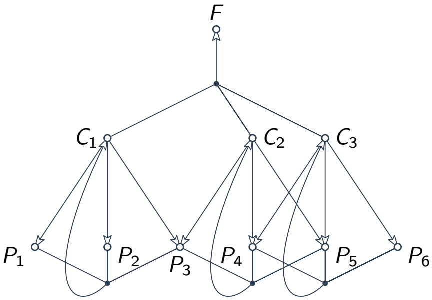
DF champêtres

- Une forêt dispose de 3 « coins à champignons » C_1, C_2, C_3.
- Un beau matin d'automne, 6 personnes $P_1, P_2, P_3, P_4, P_5, P_6$ s'en vont à la cueillette et se répartissent ainsi (une personne pourrait aller d'un coin à l'autre) :
  - P_1, P_2, P_3 vont dans le coin C_1, ce qu'on modélise par $C_1 \rightarrow P_1 P_2 P_3$ et $P_1 P_2 P_3 \rightarrow C_1$
  - P_3, P_4, P_5 vont dans C_2
  - Enfin, P_4, P_5, P_6 vont écumer le coin C_3



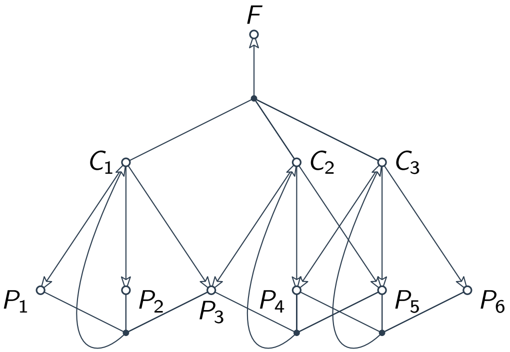
DF champêtres

- Une forêt dispose de 3 « coins à champignons » C_1, C_2, C_3.
- Un beau matin d'automne, 6 personnes $P_1, P_2, P_3, P_4, P_5, P_6$ s'en vont à la cueillette et se répartissent ainsi (une personne pourrait aller d'un coin à l'autre) :
  - P_1, P_2, P_3 vont dans le coin C_1, ce qu'on modélise par $C_1 \longrightarrow P_1 P_2 P_3$ et $P_1 P_2 P_3 \longrightarrow C_1$
  - P_3, P_4, P_5 vont dans C_2
  - Enfin, P_4, P_5, P_6 vont écumer le coin C_3
- Afin d'être sûr de n'oublier aucun champignon, on dit que la forêt est fouillée (F) quand tous les coins à champignon sont occupés, c'est à dire $C_1 C_2 C_3 \longrightarrow F$.



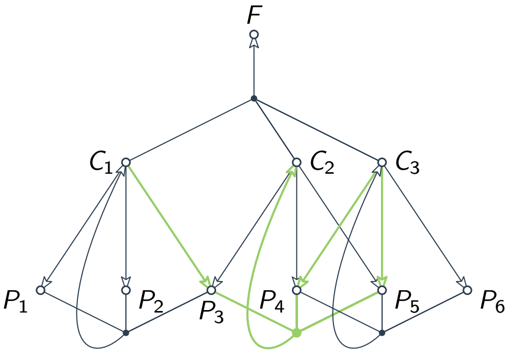
DF champêtres

- Question : est-il possible de ratisser la forêt avec seulement 2 coins à champignon ?



DF champêtres

- Question : est-il possible de ratisser la forêt avec seulement 2 coins à champignon ?
- *OUI ! ! !* Si on regarde C_1 et C_3, on a toutes les personnes du coin C_2
- On peut réduire $C_1 C_2 C_3 \longrightarrow F$ à $C_1 C_3 \longrightarrow F$.

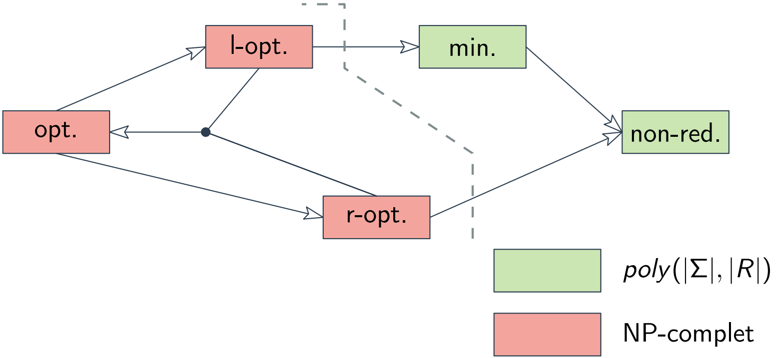


DF champêtres : difficulté

- Généralisons : on a n coins à champignon $C_1, \ldots C_n$ et m personnes $P_1, \ldots, P_m$ réparties dans ces coins.
- La question est la suivante : soit k une constante. Existe-t-il k coins à champignons qui suffisent à identifier toutes les personnes ?
- *Problème NP-complet !* (set cover)
- le rapport avec nos DF ? La seule DF que l'on peut réduire à gauche est $C_1 \ldots C_n \longrightarrow F$. Si on peut la réduire à moins de k attributs à gauche, on répond au problème.

Impact sur la concision

- $\Longrightarrow$ trouver une couverture *optimum à gauche* est *NP-complet*.
- On peut montrer de manière similaire que trouver une couverture *optimum à droite* ou *optimum* est *NP-complet* aussi [Ausiello, 1986].



Solution alternative

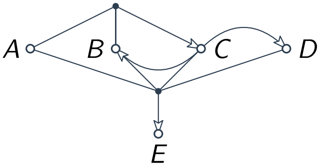
- On ne peut pas trouver d'optima, on va donc essayer de réduire « au mieux ».
- *Attention* : on ne parle plus forcément de couverture minimum !
- Notre objectif : enlever des attributs inutiles dans chaque DF $X \rightarrow Y$ de Σ.

L-réduction

- Soit Σ une couverture et $X \rightarrow Y$ dans Σ.
- On dit que $X \rightarrow Y$ est *L-réduite* si quelque soit $A \in X$, $\Sigma \not\models X \setminus \{A\} \rightarrow Y$.
- $\implies$ Chaque élément de X est *nécessaire* pour impliquer Y.

Définition - couverture L-réduite

Soit Σ un ensemble de DF. On dit que Σ est *L-réduite* si toutes ses DF sont L-réduites. Autrement dit, quelque soit $X \rightarrow Y \in \Sigma$ et $A \in X$, $\Sigma \not\models X \setminus \{A\} \rightarrow Y$.



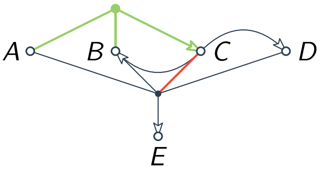
- $\Sigma = \{AB \rightarrow C, C \rightarrow BD, ABCD \rightarrow E\}$
-

L-réduction

- Soit Σ une couverture et $X \to Y$ dans Σ.
- On dit que $X \to Y$ est *L-réduite* si quelque soit $A \in X$, $\Sigma \not\models X \setminus \{A\} \to Y$.
- $\implies$ Chaque élément de X est *nécessaire* pour impliquer Y.

Définition - couverture L-réduite

Soit Σ un ensemble de DF. On dit que Σ est *L-réduite* si toutes ses DF sont L-réduites. Autrement dit, quelque soit $X \to Y \in \Sigma$ et $A \in X$, $\Sigma \not\models X \setminus \{A\} \to Y$.



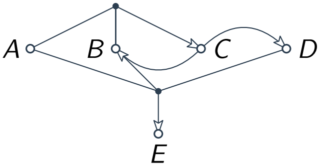
- $\Sigma = \{AB \to C, C \to BD, ABCD \to E\}$
- Puisque $AB \to C$, C ne sert à rien dans $ABCD \to E$

L-réduction

- Soit Σ une couverture et $X \rightarrow Y$ dans Σ.
- On dit que $X \rightarrow Y$ est *L-réduite* si quelque soit $A \in X$, $\Sigma \not\models X \setminus \{A\} \rightarrow Y$.
- $\implies$ Chaque élément de X est *nécessaire* pour impliquer Y.

Définition - couverture L-réduite

Soit Σ un ensemble de DF. On dit que Σ est *L-réduite* si toutes ses DF sont L-réduites. Autrement dit, quelque soit $X \rightarrow Y \in \Sigma$ et $A \in X$, $\Sigma \not\models X \setminus \{A\} \rightarrow Y$.



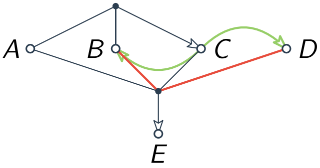
- $\Sigma = \{AB \rightarrow C, C \rightarrow BD, ABCD \rightarrow E\}$
- Puisque $AB \rightarrow C$, C ne sert à rien dans $ABCD \rightarrow E$

L-réduction

- Soit Σ une couverture et $X \to Y$ dans Σ.
- On dit que $X \to Y$ est *L-réduite* si quelque soit $A \in X$, $\Sigma \not\models X \setminus \{A\} \to Y$.
- $\Longrightarrow$ Chaque élément de X est *nécessaire* pour impliquer Y.

Définition - couverture L-réduite

Soit Σ un ensemble de DF. On dit que Σ est *L-réduite* si toutes ses DF sont L-réduites. Autrement dit, quelque soit $X \to Y \in \Sigma$ et $A \in X$, $\Sigma \not\models X \setminus \{A\} \to Y$.



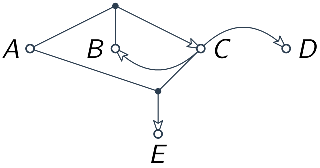
- $\Sigma = \{AB \to C, C \to BD, ABCD \to E\}$
- Mais il y a mieux ! $C \to BD$ donc B et D sont inutiles dans $ABCD \to E$

L-réduction

- Soit Σ une couverture et $X \rightarrow Y$ dans Σ.
- On dit que $X \rightarrow Y$ est *L-réduite* si quelque soit $A \in X$, $\Sigma \not\models X \setminus \{A\} \rightarrow Y$.
- $\implies$ Chaque élément de X est *nécessaire* pour impliquer Y.

Définition - couverture L-réduite

Soit Σ un ensemble de DF. On dit que Σ est *L-réduite* si toutes ses DF sont L-réduites. Autrement dit, quelque soit $X \rightarrow Y \in \Sigma$ et $A \in X$, $\Sigma \not\models X \setminus \{A\} \rightarrow Y$.



- $\Sigma = \{AB \rightarrow C, C \rightarrow BD, ABCD \rightarrow E\}$
- Mais il y a mieux ! $C \rightarrow BD$ donc B et D sont inutiles dans $ABCD \rightarrow E$

Algorithme de L-réduction

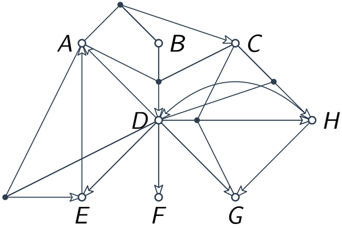
Algorithme L-REDUCE(Σ)

Data: Σ une couverture sur R

Result: Σ L-Réduite

foreach $X \rightarrow Y \in \Sigma$ **do**
  foreach $A \in X$ **do**
    if $\Sigma \models X \setminus \{A\} \rightarrow Y$ **then**
      Remplacer X par $X \setminus \{A\}$;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \rightarrow C & CDG \rightarrow H \\ ABC \rightarrow D & H \rightarrow DG \\ D \rightarrow AEFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



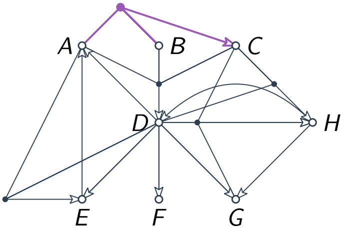
Algorithme de L-réduction

Algorithme L-REDUCE(Σ)

Data: Σ une couverture sur R
Result: Σ L-Réduite

foreach $X \rightarrow Y \in \Sigma$ **do**
  foreach $A \in X$ **do**
    if $\Sigma \models X \setminus \{A\} \rightarrow Y$ **then**
      Remplacer X par $X \setminus \{A\}$;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \rightarrow C & CDG \rightarrow H \\ ABC \rightarrow D & H \rightarrow DG \\ D \rightarrow AEFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



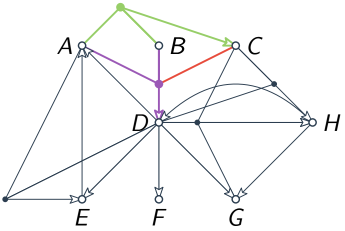
Algorithme de L-réduction

Algorithme L-REDUCE(Σ)

Data: Σ une couverture sur R
Result: Σ L-Réduite

foreach $X \rightarrow Y \in \Sigma$ **do**
  foreach $A \in X$ **do**
    if $\Sigma \models X \setminus \{A\} \rightarrow Y$ **then**
      Remplacer X par $X \setminus \{A\}$;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \rightarrow C & CDG \rightarrow H \\ ABC \rightarrow D & H \rightarrow DG \\ D \rightarrow AEFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



Algorithme de L-réduction

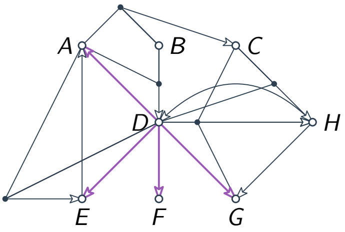
Algorithme L-REDUCE(Σ)

Data: Σ une couverture sur R

Result: Σ L-Réduite

```
foreach X → Y ∈ Σ do
    foreach A ∈ X do
        if Σ ⊨ X \ {A} → Y then
            Remplacer X par X \ {A} ;
        end
    end
end
```

$$\Sigma = \begin{bmatrix} AB \to C & CDG \to H \\ AB \to D & H \to DG \\ D \to AEFG & AD \to E \\ CD \to H & E \to A \end{bmatrix}$$



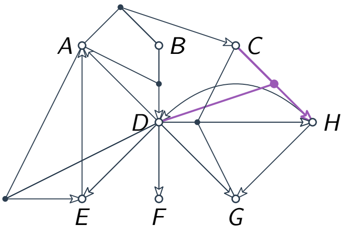
Algorithme de L-réduction

Algorithme L-REDUCE(Σ)

Data: Σ une couverture sur R
Result: Σ L-Réduite

foreach $X \rightarrow Y \in \Sigma$ **do**
  foreach $A \in X$ **do**
    if $\Sigma \models X \setminus \{A\} \rightarrow Y$ **then**
      Remplacer X par $X \setminus \{A\}$;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \rightarrow C & CDG \rightarrow H \\ AB \rightarrow D & H \rightarrow DG \\ D \rightarrow AEFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$

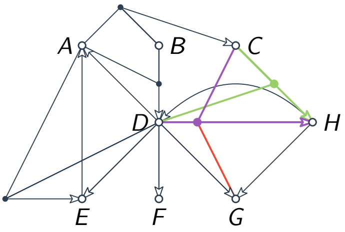


Algorithme de L-réduction

Algorithme L-REDUCE(Σ)
Data: Σ une couverture sur R
Result: Σ L-Réduite

foreach $X \to Y \in \Sigma$ **do**
  foreach $A \in X$ **do**
    if $\Sigma \models X \setminus \{A\} \to Y$ **then**
      Remplacer X par $X \setminus \{A\}$;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \to C & CDG \to H \\ AB \to D & H \to DG \\ D \to AEFG & AD \to E \\ CD \to H & E \to A \end{bmatrix}$$



Algorithme de L-réduction

Algorithme L-REDUCE(Σ)

Data: Σ une couverture sur R

Result: Σ L-Réduite

foreach $X \rightarrow Y \in \Sigma$ **do**

  foreach $A \in X$ **do**

    if $\Sigma \models X \setminus \{A\} \rightarrow Y$ **then**

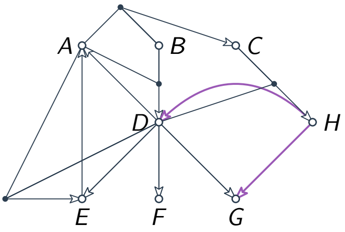
      Remplacer X par $X \setminus \{A\}$;

    end

  end

end

$$\Sigma = \begin{bmatrix} AB \rightarrow C & \\ AB \rightarrow D & H \rightarrow DG \\ D \rightarrow AEFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



Algorithme de L-réduction

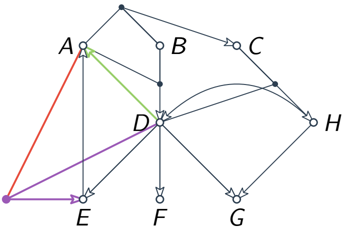
Algorithme L-REDUCE(Σ)

Data: Σ une couverture sur R

Result: Σ L-Réduite

foreach $X \rightarrow Y \in \Sigma$ **do**
  foreach $A \in X$ **do**
    if $\Sigma \models X \setminus \{A\} \rightarrow Y$ **then**
      Remplacer X par $X \setminus \{A\}$;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \rightarrow C & \\ AB \rightarrow D & H \rightarrow DG \\ D \rightarrow AEFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



Algorithme de L-réduction

Algorithme L-REDUCE(Σ)

Data: Σ une couverture sur R

Result: Σ L-Réduite

foreach $X \rightarrow Y \in \Sigma$ **do**

  foreach $A \in X$ **do**

    if $\Sigma \models X \setminus \{A\} \rightarrow Y$ **then**

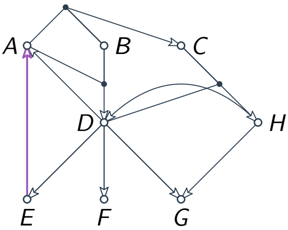
      Remplacer X par $X \setminus \{A\}$;

    end

  end

end

$$\Sigma = \begin{bmatrix} AB \rightarrow C & \\ AB \rightarrow D & H \rightarrow DG \\ D \rightarrow AEFG & \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



Algorithme de L-réduction

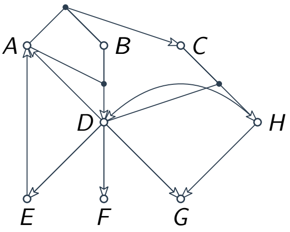
Algorithme L-REDUCE(Σ)

Data: Σ une couverture sur R

Result: Σ L-Réduite

foreach $X \rightarrow Y \in \Sigma$ **do**
  foreach $A \in X$ **do**
    if $\Sigma \models X \setminus \{A\} \rightarrow Y$ **then**
      Remplacer X par $X \setminus \{A\}$;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \rightarrow C & \\ AB \rightarrow D & H \rightarrow DG \\ D \rightarrow AEFG & \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



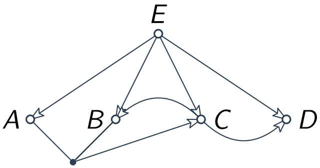
R-réduction

- Soit Σ une couverture et $X \to Y \in \Sigma$.
- On peut *R-réduire* $X \to Y$ si il y a un $A \in Y$ que l'on peut retirer, en restant équivalent à Σ.
- Autrement dit, A est atteint par X même si on ne le marque pas (transitivité).

Définition - Couverture R-réduite

Soit Σ un ensemble de DF. On dit que Σ est *R-réduite* si toutes ses DF sont R-réduites. Autrement dit, pour tout $X \to Y \in \Sigma$ et $A \in Y$,

$$(\Sigma \setminus \{X \to Y\}) \cup \{X \to Y \setminus \{A\}\} \not\models X \to Y$$



- $\Sigma = \{AB \to C, C \to BD, E \to ABCD\}$
-

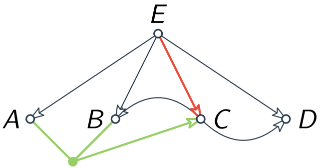
R-réduction

- Soit Σ une couverture et $X \to Y \in \Sigma$.
- On peut *R-réduire* $X \to Y$ si il y a un $A \in Y$ que l'on peut retirer, en restant équivalent à Σ.
- Autrement dit, A est atteint par X même si on ne le marque pas (transitivité).

Définition - Couverture R-réduite

Soit Σ un ensemble de DF. On dit que Σ est *R-réduite* si toutes ses DF sont R-réduites. Autrement dit, pour tout $X \to Y \in \Sigma$ et $A \in Y$,

$$(\Sigma \setminus \{X \to Y\}) \cup \{X \to Y \setminus \{A\}\} \not\models X \to Y$$



- $\Sigma = \{AB \to C, C \to BD, E \to ABCD\}$
- Puisque $AB \to C$, C ne sert à rien dans $E \to ABCD$

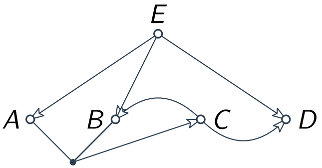
R-réduction

- Soit Σ une couverture et $X \rightarrow Y \in \Sigma$.
- On peut *R-réduire* $X \rightarrow Y$ si il y a un $A \in Y$ que l'on peut retirer, en restant équivalent à Σ.
- Autrement dit, A est atteint par X même si on ne le marque pas (transitivité).

Définition - Couverture R-réduite

Soit Σ un ensemble de DF. On dit que Σ est *R-réduite* si toutes ses DF sont R-réduites. Autrement dit, pour tout $X \rightarrow Y \in \Sigma$ et $A \in Y$,

$$(\Sigma \setminus \{X \rightarrow Y\}) \cup \{X \rightarrow Y \setminus \{A\}\} \not\models X \rightarrow Y$$



- $\Sigma = \{AB \rightarrow C, C \rightarrow BD, E \rightarrow ABCD\}$
- Puisque $AB \rightarrow C$, C ne sert à rien dans $E \rightarrow ABCD$

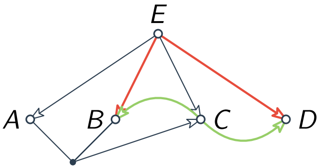
R-réduction

- Soit Σ une couverture et $X \to Y \in \Sigma$.
- On peut *R-réduire* $X \to Y$ si il y a un $A \in Y$ que l'on peut retirer, en restant équivalent à Σ.
- Autrement dit, A est atteint par X même si on ne le marque pas (transitivité).

Définition - Couverture R-réduite

Soit Σ un ensemble de DF. On dit que Σ est *R-réduite* si toutes ses DF sont R-réduites. Autrement dit, pour tout $X \to Y \in \Sigma$ et $A \in Y$,

$$(\Sigma \setminus \{X \to Y\}) \cup \{X \to Y \setminus \{A\}\} \not\models X \to Y$$



- $\Sigma = \{AB \to C, C \to BD, E \to ABCD\}$
- Mais il y a mieux ! $C \to BD$ donc B et D sont inutiles dans $E \to ABCD$

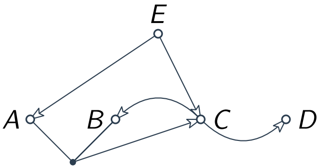
R-réduction

- Soit Σ une couverture et $X \to Y \in \Sigma$.
- On peut *R-réduire* $X \to Y$ si il y a un $A \in Y$ que l'on peut retirer, en restant équivalent à Σ.
- Autrement dit, A est atteint par X même si on ne le marque pas (transitivité).

Définition - Couverture R-réduite

Soit Σ un ensemble de DF. On dit que Σ est *R-réduite* si toutes ses DF sont R-réduites. Autrement dit, pour tout $X \to Y \in \Sigma$ et $A \in Y$,

$$(\Sigma \setminus \{X \to Y\}) \cup \{X \to Y \setminus \{A\}\} \not\models X \to Y$$



- $\Sigma = \{AB \to C, C \to BD, E \to ABCD\}$
- Mais il y a mieux ! $C \to BD$ donc B et D sont inutiles dans $E \to ABCD$

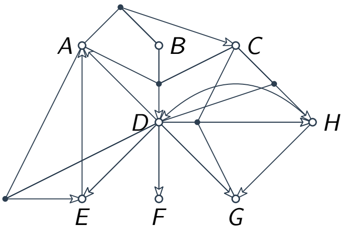
Algorithme de R-réduction

Algorithme R-REDUCE(Σ)

Data: Σ une couverture sur R
Result: Σ R-Réduite

foreach $X \rightarrow Y \in \Sigma$ **do**
  foreach $A \in Y$ **do**
    Remplacer $X \rightarrow Y$ par $X \rightarrow Y \setminus \{A\}$ dans Σ
    if $\Sigma \not\models X \rightarrow Y$ **then**
      Remettre $X \rightarrow Y$ dans Σ ;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \rightarrow C & CDG \rightarrow H \\ ABC \rightarrow D & H \rightarrow DG \\ D \rightarrow AEFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



Algorithme de R-réduction

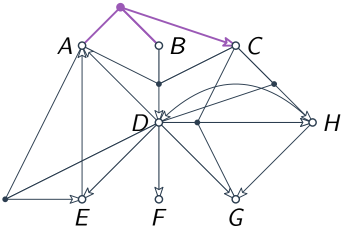
Algorithme R-REDUCE(Σ)

Data: Σ une couverture sur R

Result: Σ R-Réduite

foreach $X \to Y \in \Sigma$ **do**
  foreach $A \in Y$ **do**
    Remplacer $X \to Y$ par $X \to Y \setminus \{A\}$ dans Σ
    if $\Sigma \not\models X \to Y$ **then**
      Remettre $X \to Y$ dans Σ ;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \to C & CDG \to H \\ ABC \to D & H \to DG \\ D \to AEFG & AD \to E \\ CD \to H & E \to A \end{bmatrix}$$



Algorithme de R-réduction

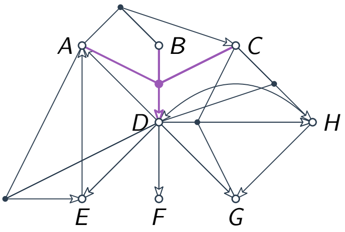
Algorithme R-REDUCE(Σ)

Data: Σ une couverture sur R

Result: Σ R-Réduite

foreach $X \rightarrow Y \in \Sigma$ **do**
    foreach $A \in Y$ **do**
        Remplacer $X \rightarrow Y$ par $X \rightarrow Y \setminus \{A\}$ dans Σ
        if $\Sigma \not\models X \rightarrow Y$ **then**
            Remettre $X \rightarrow Y$ dans Σ ;
        end
    end
end

$$\Sigma = \begin{bmatrix} AB \rightarrow C & CDG \rightarrow H \\ ABC \rightarrow D & H \rightarrow DG \\ D \rightarrow AEFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



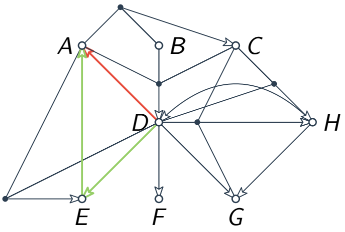
Algorithme de R-réduction

Algorithme R-REDUCE(Σ)

Data: Σ une couverture sur R
Result: Σ R-Réduite

foreach $X \rightarrow Y \in \Sigma$ **do**
  foreach $A \in Y$ **do**
    Remplacer $X \rightarrow Y$ par $X \rightarrow Y \setminus \{A\}$ dans Σ
    if $\Sigma \not\models X \rightarrow Y$ **then**
      Remettre $X \rightarrow Y$ dans Σ ;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \rightarrow C & CDG \rightarrow H \\ ABC \rightarrow D & H \rightarrow DG \\ D \rightarrow AEFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



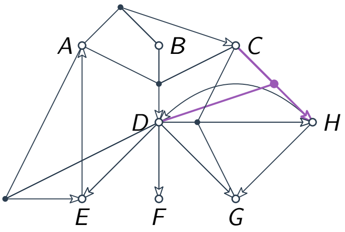
Algorithme de R-réduction

Algorithme R-REDUCE(Σ)

Data: Σ une couverture sur R
Result: Σ R-Réduite

foreach $X \to Y \in \Sigma$ **do**
  foreach $A \in Y$ **do**
    Remplacer $X \to Y$ par $X \to Y \setminus \{A\}$ dans Σ
    if $\Sigma \not\models X \to Y$ **then**
      Remettre $X \to Y$ dans Σ ;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \to C & CDG \to H \\ ABC \to D & H \to DG \\ D \to EFG & AD \to E \\ CD \to H & E \to A \end{bmatrix}$$



Algorithme de R-réduction

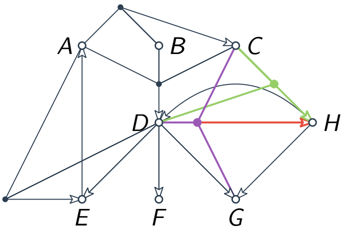
Algorithme R-REDUCE(Σ)

Data: Σ une couverture sur R

Result: Σ R-Réduite

```
foreach X → Y ∈ Σ do
    foreach A ∈ Y do
        Remplacer X → Y par X → Y \ {A}
         dans Σ
        if Σ ⊭ X → Y then
            Remettre X → Y dans Σ ;
        end
    end
end
```

$$\Sigma = \begin{bmatrix} AB \rightarrow C & CDG \rightarrow H \\ ABC \rightarrow D & H \rightarrow DG \\ D \rightarrow EFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



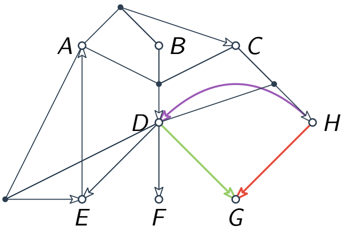
Algorithme de R-réduction

Algorithme R-REDUCE(Σ)

Data: Σ une couverture sur R
Result: Σ R-Réduite

foreach $X \rightarrow Y \in \Sigma$ **do**
  foreach $A \in Y$ **do**
    Remplacer $X \rightarrow Y$ par $X \rightarrow Y \setminus \{A\}$ dans Σ
    if $\Sigma \not\models X \rightarrow Y$ **then**
      Remettre $X \rightarrow Y$ dans Σ ;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \rightarrow C & \\ ABC \rightarrow D & H \rightarrow DG \\ D \rightarrow EFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



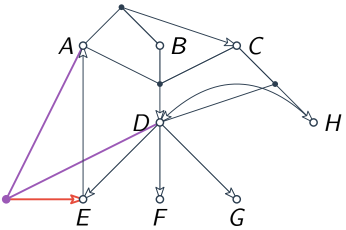
Algorithme de R-réduction

Algorithme R-REDUCE(Σ)

Data: Σ une couverture sur R
Result: Σ R-Réduite

```
foreach X → Y ∈ Σ do
    foreach A ∈ Y do
        Remplacer X → Y par X → Y \ {A}
         dans Σ
        if Σ ⊭ X → Y then
            Remettre X → Y dans Σ ;
        end
    end
end
```

$$\Sigma = \begin{bmatrix} AB \rightarrow C & \\ ABC \rightarrow D & H \rightarrow D \\ D \rightarrow EFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



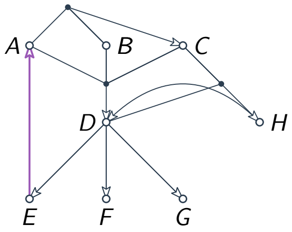
Algorithme de R-réduction

Algorithme R-REDUCE(Σ)

Data: Σ une couverture sur R
Result: Σ R-Réduite

foreach $X \to Y \in \Sigma$ **do**
  foreach $A \in Y$ **do**
    Remplacer $X \to Y$ par $X \to Y \setminus \{A\}$ dans Σ
    if $\Sigma \not\models X \to Y$ **then**
      Remettre $X \to Y$ dans Σ ;
    end
  end
end

$$\Sigma = \begin{bmatrix} AB \to C & \\ ABC \to D & H \to D \\ D \to EFG & \\ CD \to H & E \to A \end{bmatrix}$$



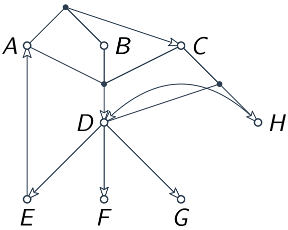
Algorithme de R-réduction

Algorithme R-REDUCE(Σ)

Data: Σ une couverture sur R
Result: Σ R-Réduite

```
foreach X → Y ∈ Σ do
    foreach A ∈ Y do
        Remplacer X → Y par X → Y \ {A}
         dans Σ
        if Σ ⊭ X → Y then
            Remettre X → Y dans Σ ;
        end
    end
end
```

$$\Sigma = \begin{bmatrix} AB \rightarrow C & \\ ABC \rightarrow D & H \rightarrow D \\ D \rightarrow EFG & \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$



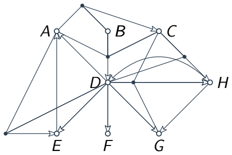
Couverture Réduite

Définition - Couverture réduite

Une couverture Σ est *réduite* si elle non-redondante, L-réduite et R-réduite.

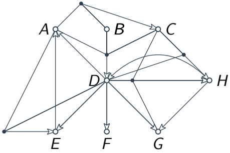
- Une couverture réduite n'est pas forcément minimum.
- Procédure pour transformer Σ en *une* couverture réduite :
  - appliquer une L-réduction, puis une R-réduction.
  - enlever les redondances (en particulier, les doublons et les DF $X \to \emptyset$).
- la R-réduction *préserve* la L-réduction, mais le contraire n'est *pas vrai* !
- Tout est $poly(|\Sigma|, |R|)$

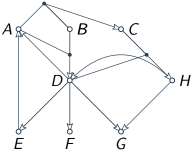
Exemple de réduction



$$\Sigma = \begin{bmatrix} AB \rightarrow C & CDG \rightarrow H \\ ABC \rightarrow D & H \rightarrow DG \\ D \rightarrow AEFG & AD \rightarrow E \\ CD \rightarrow H & E \rightarrow A \end{bmatrix}$$

Exemple de réduction



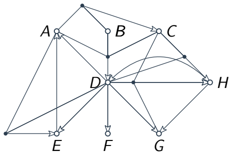


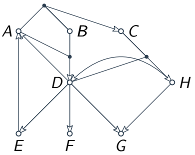
$$\Sigma = \begin{bmatrix} AB \to C & CDG \to H \\ ABC \to D & H \to DG \\ D \to AEFG & AD \to E \\ CD \to H & E \to A \end{bmatrix}$$

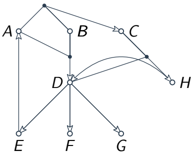
$$\Sigma = \begin{bmatrix} AB \to C & \\ AB \to D & H \to DG \\ D \to AEFG & \\ CD \to H & E \to A \end{bmatrix}$$

L-réduction

Exemple de réduction







$$\Sigma = \begin{bmatrix} AB \to C & CDG \to H \\ ABC \to D & H \to DG \\ D \to AEFG & AD \to E \\ CD \to H & E \to A \end{bmatrix}$$

$\xrightarrow{\text{L-réduction}}$

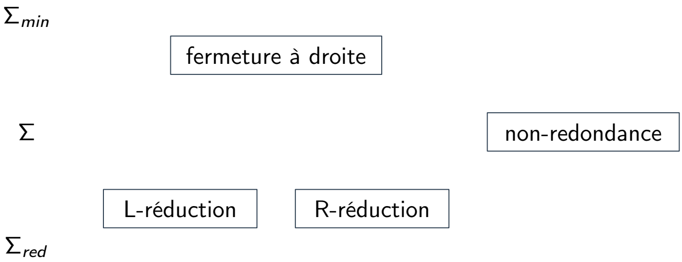
$$\Sigma = \begin{bmatrix} AB \to C & \\ AB \to D & H \to DG \\ D \to AEFG & \\ CD \to H & E \to A \end{bmatrix}$$

$\xrightarrow{\text{R-réduction}}$

$$\Sigma = \begin{bmatrix} AB \to C & \\ AB \to D & H \to D \\ D \to EFG & \\ CD \to H & E \to A \end{bmatrix}$$

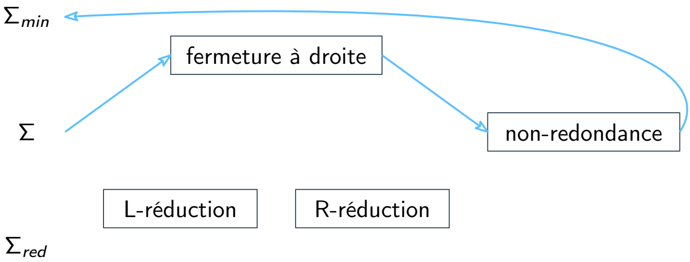
Résumé des opérations

- Σ_{min} couverture minimum, Σ_{red} couverture réduite
-



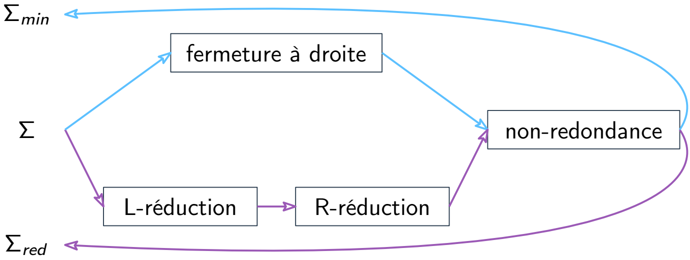
Résumé des opérations

- Σ_{min} couverture minimum, Σ_{red} couverture réduite
-



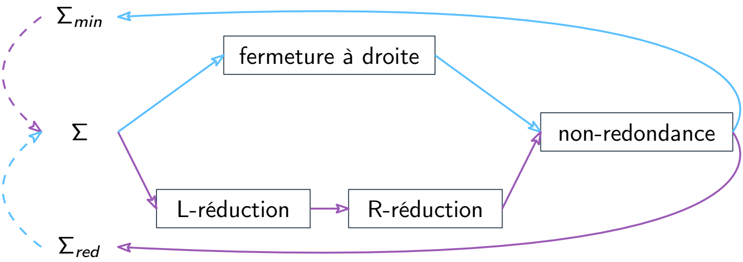
Résumé des opérations

- Σ_{min} couverture minimum, Σ_{red} couverture réduite
-



Résumé des opérations

- Σ_{min} couverture minimum, Σ_{red} couverture réduite
- On peut aussi voyager entre Σ_{min} à Σ_{red}



Bonus : Extension d'une base

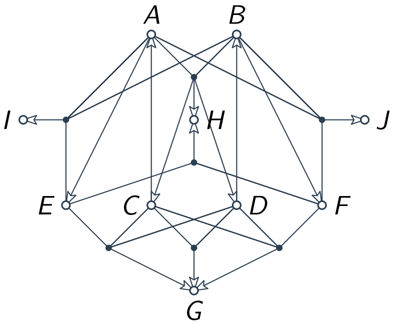
- Intuitivement, $X \rightarrow ABC$ a le même sens que $X \rightarrow A$, $X \rightarrow B$, $X \rightarrow C$.
- L'*extension* de $X \rightarrow Y$ est $\{X \rightarrow A \mid A \in Y\}$.
- Plus généralement, on obtient l'*extension* de Σ en remplaçant chacune de ses DF par leurs extensions.
- L'*aggrégation* est l'opération inverse ! On passe de $\{X \rightarrow A \mid A \in Y\}$ à $X \rightarrow Y$, de même pour Σ.

Définition - Couverture Canonique

L'extension d'une couverture réduite est dite *canonique*.

Pour finir

- On reprend l'introduction $\mathrm{R} = \{A, B, C, D, E, F, G, H, I, J\}$
- $\Sigma = \{C \to A, A \to E, D \to B, B \to F, AB \to CDH, CD \to G, EF \to H,$ $ABE \to I, ABF \to J, ECD \to G, CDF \to G\}$
  1. Minimiser puis réduire Σ,
  2. Réduire puis minimiser Σ,
  3. Montrer qu'une base étendue est non-redondante si et seulement si elle est R-réduite.



Références

- G. Ausiello, A. D'Atri and D. Sacca.
  Minimal representation of directed hypergraphs.
  SIAM Journal on Computing, 15 :418-431, 1986.
- D. Maier.
  The theory of relational databases
  Computer science press Rockville, 1983.
- R. Shock.
  Computing the minimum cover of functional dependencies.
  Information Processing Letters, 22 :157-159, 1986.
- M. Wild.
  The joy of implications, aka pure Horn formulas : mainly a survey.
  Theoretical Computer Science, 658 :264-292, 2017.