Complexité CM10

Antonio E. Porreca
aeporreca.org/complexite

Diapositives basées sur le livre Computational Complexity par Christos H. Papadimitriou

Avant de commencer :

VERTEX-COVER $\leq$ DOMINATING-SET

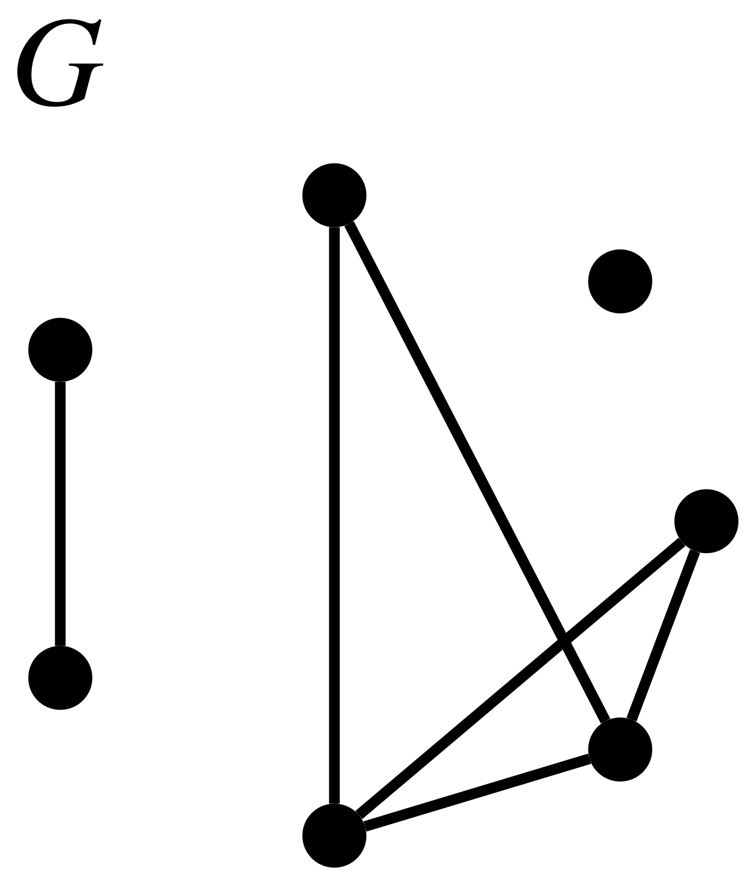
VERTEX-COVER

- Entrée : un graphe non orienté $G = (V, E)$ et un entier k
- Question : existe-t-il un ensemble $C \subseteq V$ de taille $\leq k$ tel que chaque arête dans E a au moins un sommet dans C ?

DOMINATING-SET

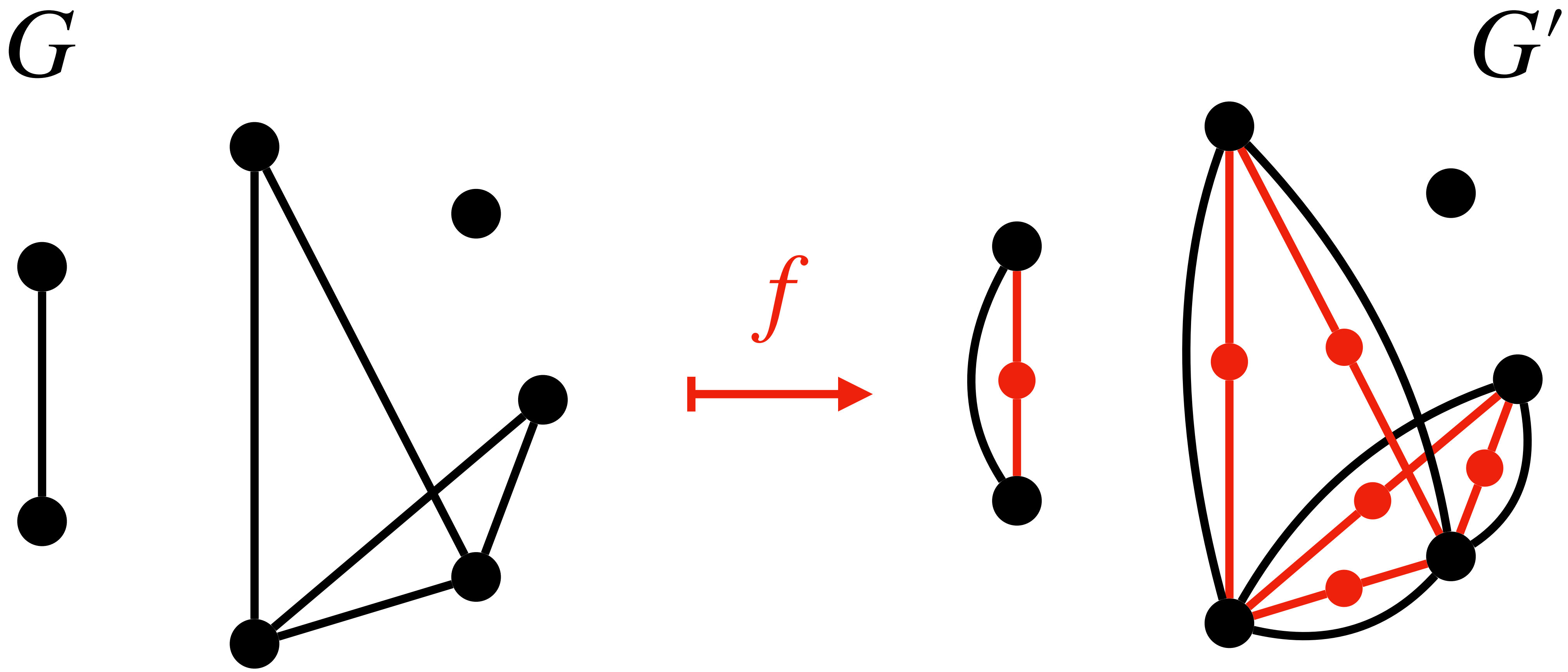
- Entrée : un graphe non orienté $G = (V, E)$ et un entier k
- Question : existe-t-il un ensemble $S \subseteq V$ de taille $\leq k$ qui domine G, c'est-à-dire que chaque sommet est soit lui-même dans S, soit adjacent à un sommet dans S ?

Réduction



$k = 3$

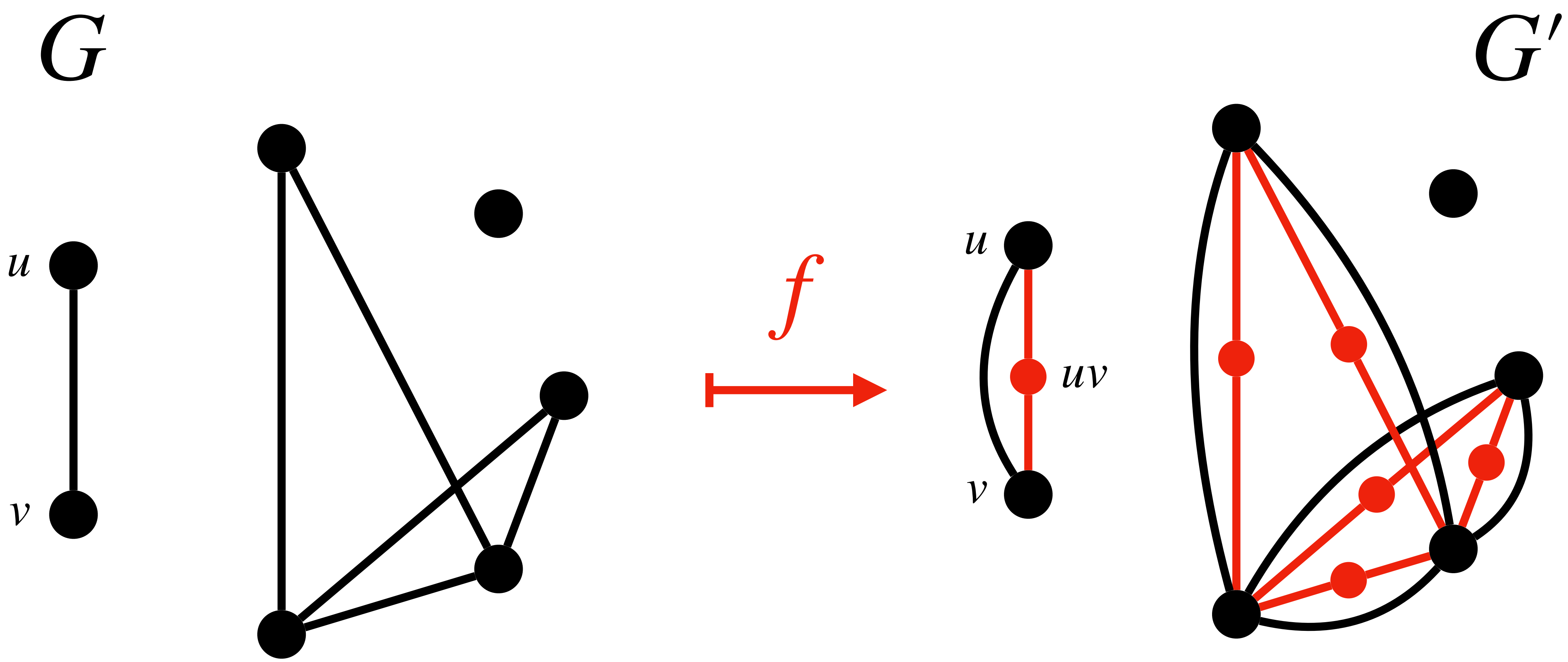
Réduction



$k = 3$

$k' = k + n_i = 4$

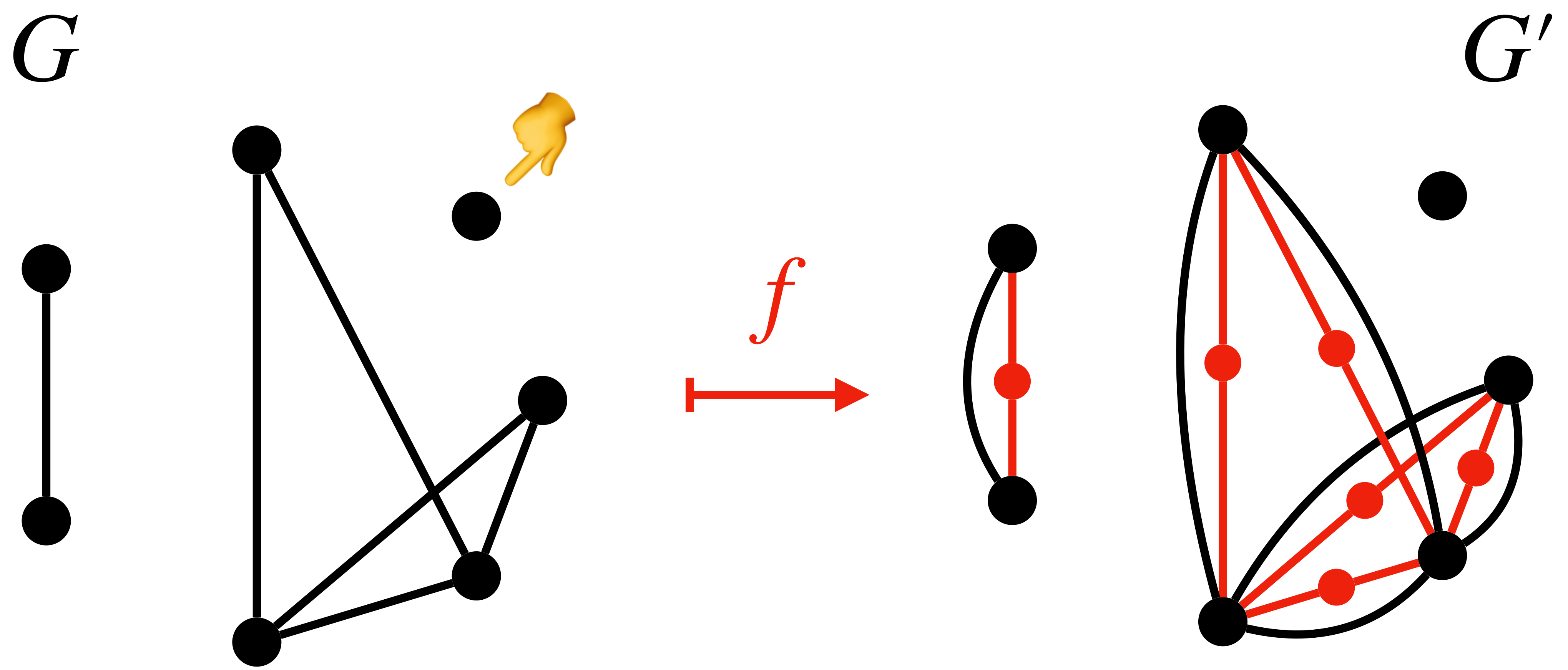
Réduction



$k = 3$

$k' = k + n_i = 4$

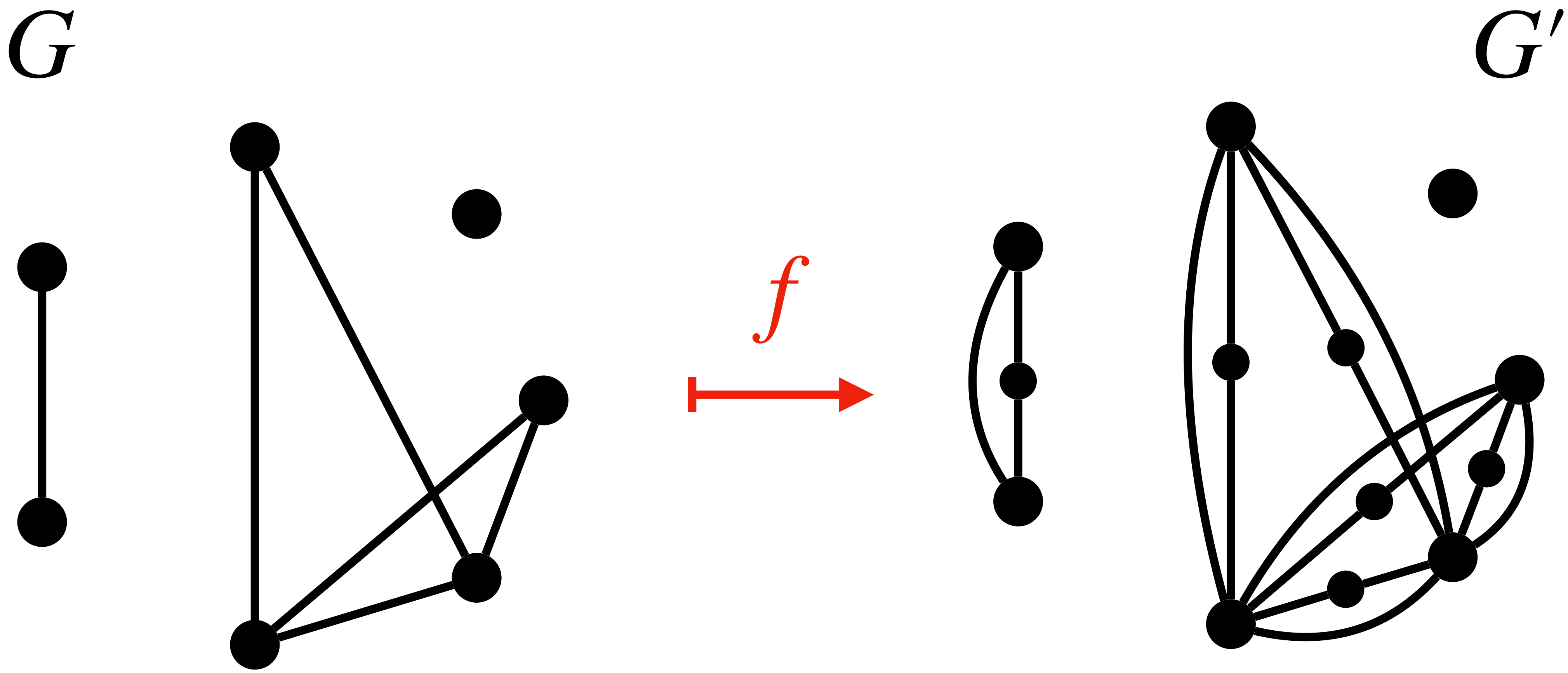
Réduction



$k = 3$

$k' = k + n_i = 4$

Réduction



$k = 3$

$k' = k + n_i = 4$

G a une vertex cover de taille $\leq k \Rightarrow$ G' a un dominating set de taille $\leq k'$

- Si C est une vertex cover de G, alors soit $S = C \cup \{$sommets isolés de $G\}$, donc $|S| \leq |C| + n_i = k'$

- Tous les sommets isolés de G' sont dominés (dans S)

- Pour chaque nouveau sommet uv de G', soit $u \in C \subseteq S$, soit $v \in C \subseteq S$ pour couvrir (u, v) dans G, donc uv est dominé

- Chaque sommet original de G est soit dans $C \subseteq S$, soit adjacent à un sommet dans $C \subseteq S$ et donc il est dominé

G' a un dominating set de taille $\leq k'$ $\Rightarrow G$ a une vertex cover de taille $\leq k$

- Soit S un dominating set de G' : alors tous les n_i sommets isolés sont dans S ; soit $S' = S - \{$sommets isolés de $G\}$, donc $|S'| = k' - n_i = k$

- Démontrons qu'on n'a jamais besoin des nouveaux sommets uv pour dominer G'

- Si $uv \in S'$, il domine lui-même, u et v ; donc, si on le remplace avec u, les sommets u, v et uv restent dominés

- Soit C l'ensemble obtenu en faisant ces remplacements : alors $|C| \leq k$ et C est une vertex cover pour G

- Sinon il y aurait (u, v) dans G qui n'est pas couvert par C, c'est-à-dire que $u, v \notin C$; mais alors uv ne serait adjacent à aucun sommet de S, contradiction

Et maintenant,
la suite

Problèmes d’optimisation

Problèmes d'optimisation

- Pour chaque instance x (un mot dans $\Sigma^\star$) on a un ensemble $F(x) \subseteq \Sigma^\star$ de solutions réalisables et une fonction de coût (ou fonction objectif) $c : F(x) \to \mathbb{R}^+$
- Le coût optimal est $\mathrm{OPT}(x) = \min\{c(s) : s \in F(x)\}$ pour les problèmes de minimisation 📉, ou bien $\mathrm{OPT}(x) = \max\{c(s) : s \in F(x)\}$ pour les problèmes de maximisation 📈
- On veut trouver une solution optimale ✌️, c'est-à-dire un $s \in F(x)$ tel que $c(s) = \mathrm{OPT}(x)$

Problème du voyageur de commerce (TSP) 🧳

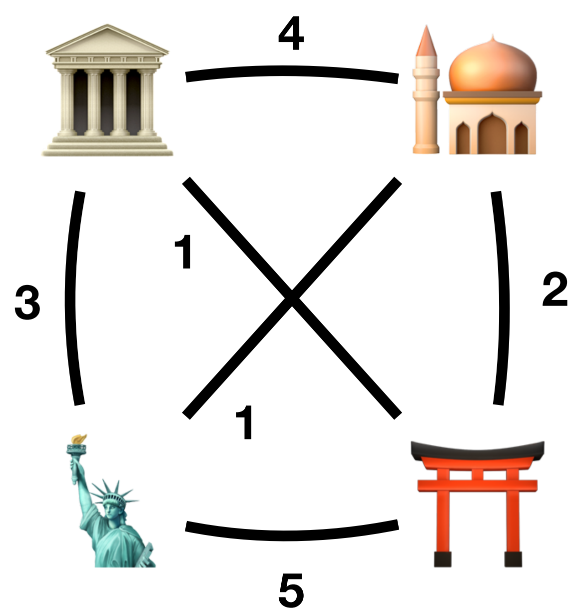
- Entrée : un graphe non orienté complet pondéré $G = (V, E = V^2, w)$ où $w : E \to \mathbb{R}^+$ est la distance entre chaque paire de points

- Sortie : un cycle hamiltonien (un cycle qui traverse chaque sommet une et une seule fois) de poids minimal 📉

Problème du voyageur de commerce (TSP)

- Entrée : un graphe non orienté complet pondéré $G = (V, E = V^2, w)$ où $w : E \to \mathbb{R}^+$ est la distance entre chaque paire de points

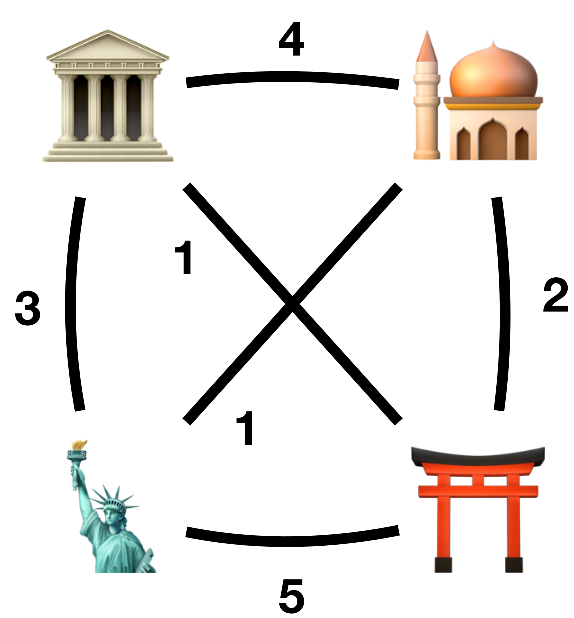
- Sortie : un cycle hamiltonien (un cycle qui traverse chaque sommet une et une seule fois) de poids minimal

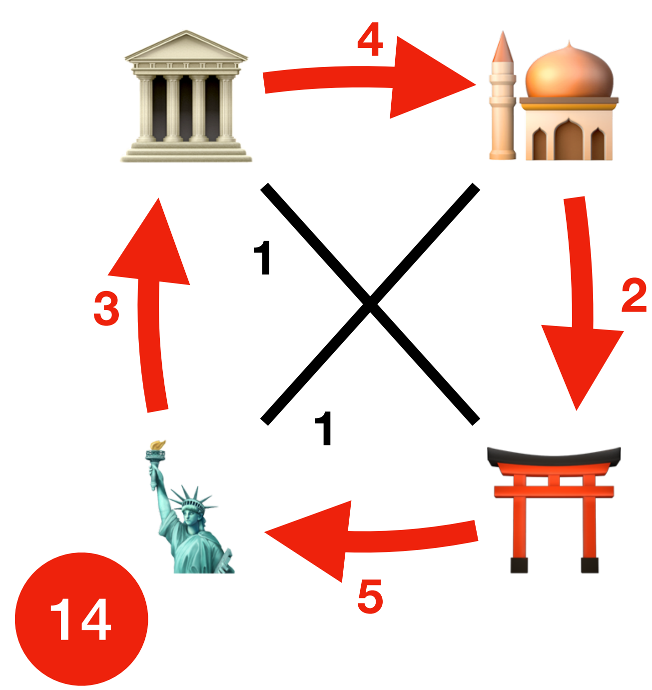


Problème du voyageur de commerce (TSP) 🧳

- Entrée : un graphe non orienté complet pondéré $G = (V, E = V^2, w)$ où $w: E \to \mathbb{R}^+$ est la distance entre chaque paire de points

- Sortie : un cycle hamiltonien (un cycle qui traverse chaque sommet une et une seule fois) de poids minimal 📉

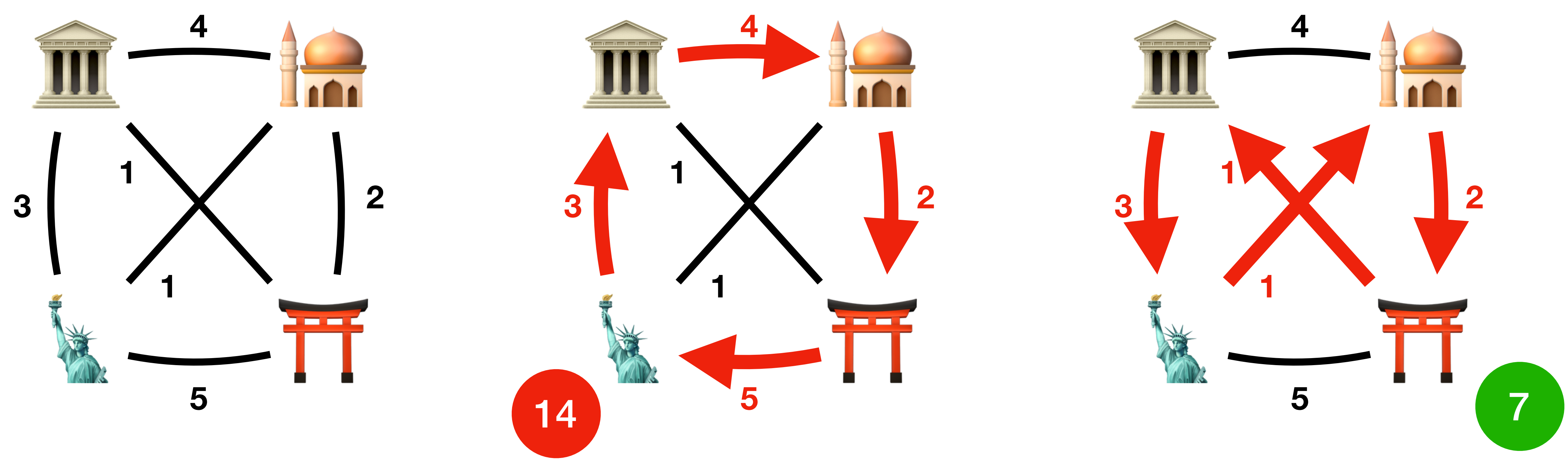




Problème du voyageur de commerce (TSP) 🧳

- Entrée : un graphe non orienté complet pondéré $G = (V, E = V^2, w)$ où $w : E \to \mathbb{R}^+$ est la distance entre chaque paire de points

- Sortie : un cycle hamiltonien (un cycle qui traverse chaque sommet une et une seule fois) de poids minimal 📉



Problème du voyageur de commerce (TSP) 🧳

- Instance $x =$ un graphe non orienté complet $G = (V, E = V^2)$ et sa fonction de coût $w\colon E \to \mathbb{R}^+$ codés sur un alphabet Σ

- Solutions réalisables $F(x) =$ toutes les cycles hamiltoniens de G, c'est-à-dire, toutes les permutations de son ensemble de sommets V

- Fonction de coût $c(v_0, \ldots, v_{n-1}) = \sum_{i=0}^{n-1} w\left(v_i, v_{(i+1) \bmod n}\right)$

- Coût optimal $\mathrm{OPT}(x) =$ longueur de l'un des cycles hamiltoniens les plus courts

- Solution optimale $s =$ un cycle hamiltonien tel que $c(s) = \mathrm{OPT}(x)$

Problèmes de décision vs problèmes d'optimisation

Chaque problème d'optimisation de maximisation 📈 (resp., de minimisation 📉) A a un problème de décision associé DECISION-A :

- Etant donné une entrée x du même type que A et un entier k, existe-t-il une solution réalisable $s \in F(x)$ telle que $c(s) \geq k$ 📈 (resp., $c(s) \leq k$ 📉) ?

Problèmes de décision vs problèmes d'optimisation

- Si on peut résoudre A avec un algorithme déterministe en temps polynomial, alors DECISION-$A \in \mathbf{P}$
    - Il suffit de résoudre A en temps polynomial et vérifier si le coût de la solution optimale obtenue est $\geq k$ 📈 ($\leq k$ pour les problèmes de minimisation 📉)
- Inversement, si on ne peut pas résoudre en temps polynomial DECISION-A, par exemple s'il est $\mathbf{NP}$-complet, alors on ne peut pas résoudre A en temps polynomial non plus*

* sous l'hypothèse que $\mathbf{P} \neq \mathbf{NP}$

Algorithmes d’approximation

Algorithmes d'approximation

Un algorithme (MT déterministe) M est dit algorithme de ε-approximation (avec $0 \leq \varepsilon \leq 1$) si pour toute entrée x il renvoie une solution réalisable $M(x) \in F(x)$ telle que

$$\frac{|\, c(M(x)) - \mathrm{OPT}(x)\, |}{\max\{\mathrm{OPT}(x), c(M(x))\}} \leq \varepsilon$$

Signification de $\dfrac{|c(M(x)) - \mathrm{OPT}(x)|}{\max\{\mathrm{OPT}(x), c(M(x))\}} \leq \varepsilon$

- Pour un problème de maximisation 📈 ça veut dire que

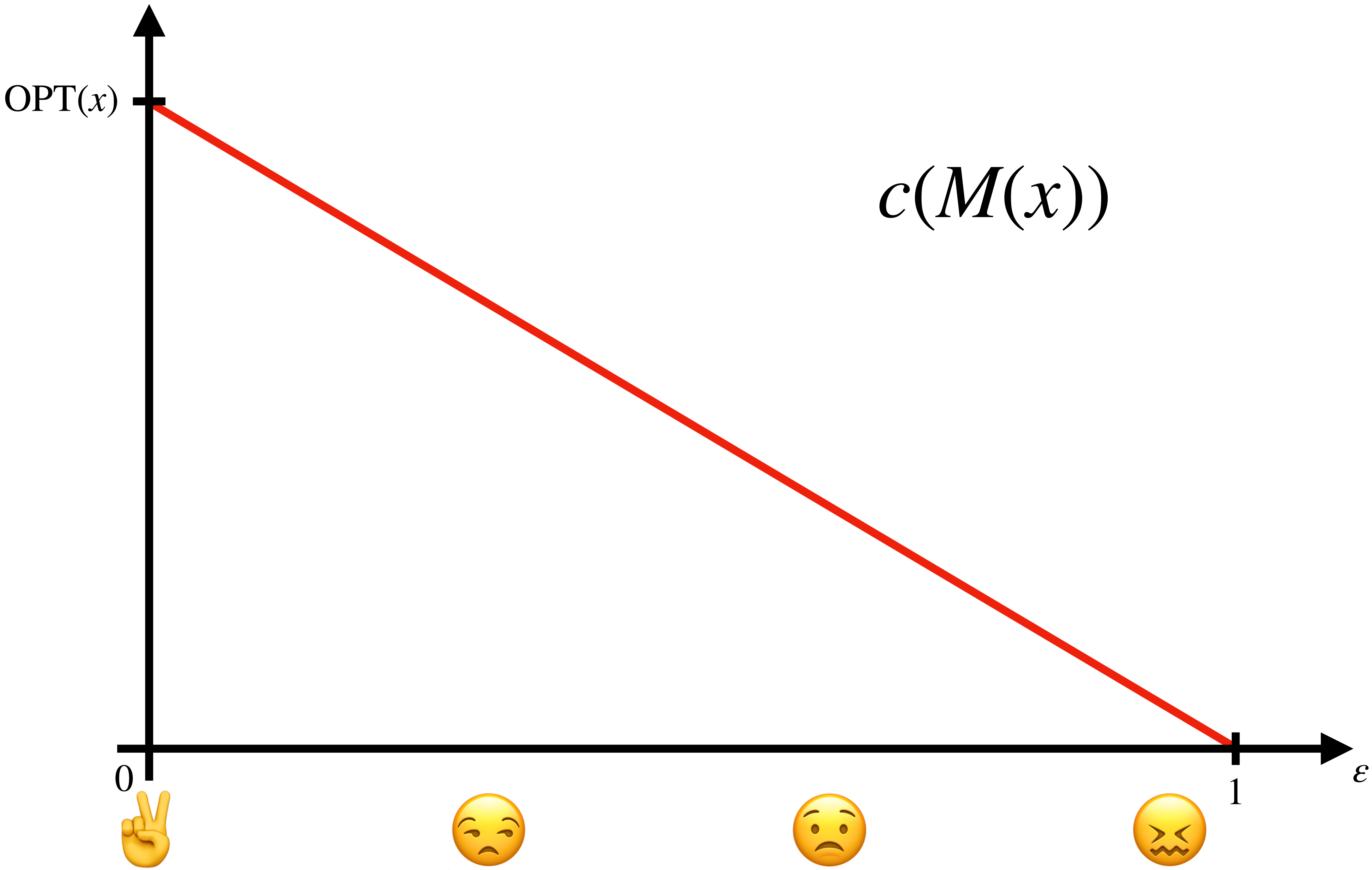
$$\frac{\mathrm{OPT}(x) - c(M(x))}{\mathrm{OPT}(x)} \leq \varepsilon$$

- Donc $c(M(x)) \geq (1 - \varepsilon)\mathrm{OPT}(x)$

- Un algorithme de 0-approximation renvoie donc toujours une solution optimale : $c(M(x)) \geq \mathrm{OPT}(x)$ donc $c(M(x)) = \mathrm{OPT}(x)$

- Par contre, un algorithme de 1-approximation peut renvoyer n'importe quelle solution réalisable : $c(M(x)) \geq 0$

Max $c(M(x)) \geq (1 - \varepsilon)\mathrm{OPT}(x)$



Signification de $\dfrac{|c(M(x)) - \mathrm{OPT}(x)|}{\max\{\mathrm{OPT}(x), c(M(x))\}} \leq \varepsilon$

- Pour un problème de minimisation 📉 ça veut dire que

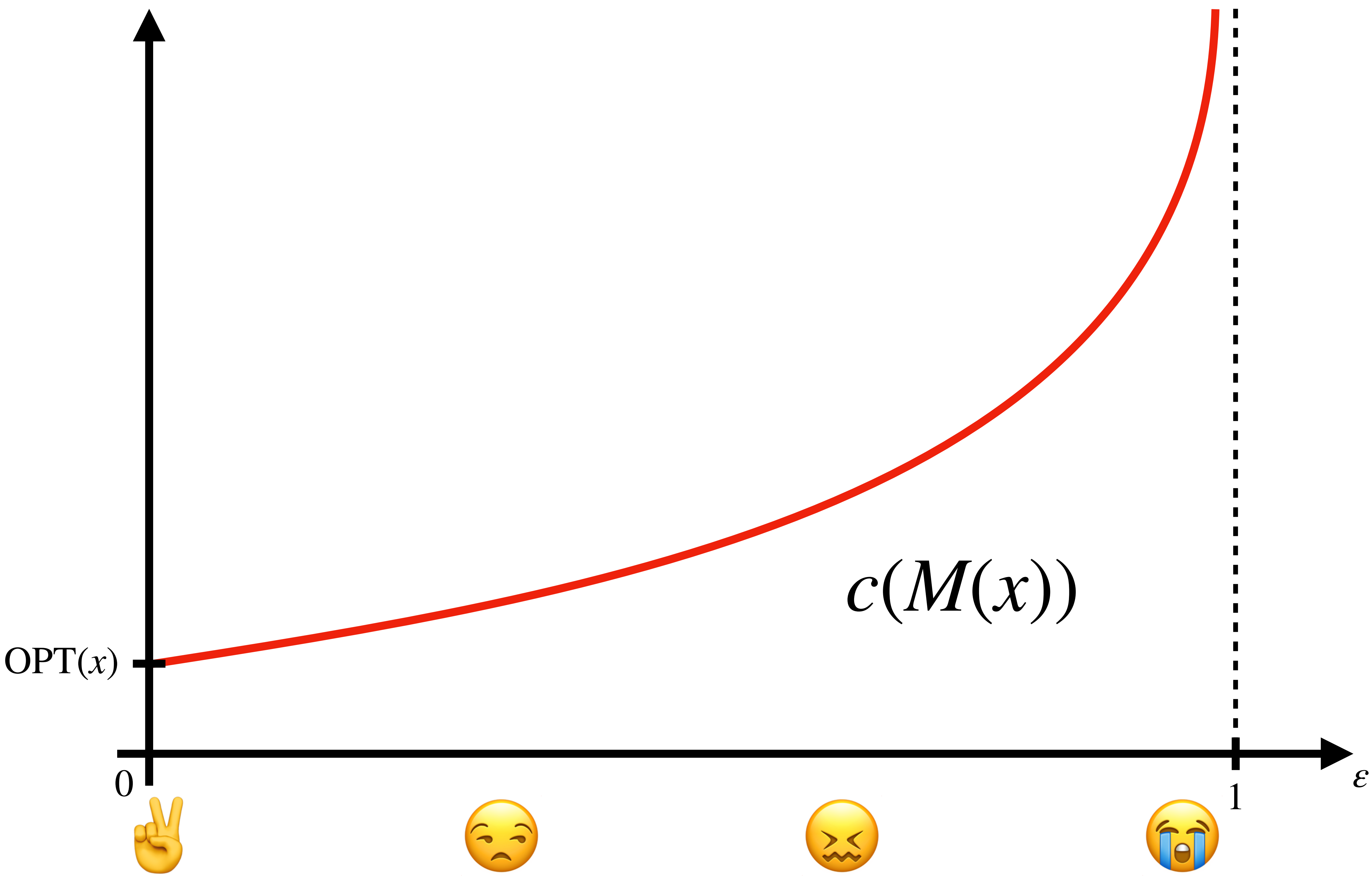
$$\frac{c(M(x)) - \mathrm{OPT}(x)}{c(M(x))} \leq \varepsilon$$

- Donc $c(M(x)) \leq \frac{1}{1-\varepsilon}\mathrm{OPT}(x)$

- Un algorithme de 0-approximation renvoie donc toujours une solution optimale : $c(M(x)) \leq \mathrm{OPT}(x)$ donc $c(M(x)) = \mathrm{OPT}(x)$

- Par contre, un algorithme de 1-approximation peut renvoyer n'importe quelle solution réalisable : $c(M(x)) \leq \infty$ (👈 petit abus de notation ici)

Min $c(M(x)) \leq \frac{1}{1-\varepsilon}\mathrm{OPT}(x)$



Problèmes approximables

- On est intéressé à trouver des algorithmes de ε-approximation (avec un petit ε !) qui fonctionnent en temps polynomial pour les problèmes dont la version de décision est **NP**-complete

- Où possible, on voudrait trouver le plus petit ε, mais parfois ça n'existe pas, même si on peut trouver des ε arbitrairement petits

- La seuil d'approximation d'un problème d'optimisation A est le plus petit ε tel qu'il existe un algorithme de ε-approximation pour A

Un problème difficile
mais approximable 🤷‍♀️

Problème d'optimisation VERTEX-COVER

- Entrée : un graphe non orienté $G = (V, E)$

- Sortie : le plus petit ensemble $C \subseteq V$ tel que chaque arête dans E a au moins un sommet dans C

- Il s'agit donc de minimiser 📉 le coût $c(C) = |C|$ parmi toutes les couvertures $C \subseteq V$

- Rappel : DECISION-VERTEX-COVER (existe-t-il une couverture de taille $\leq k$?) est $\mathbf{NP}$-complet, donc il n'existe pas* un algo exact polynomial pour le problème d'optimisation

* sous l'hypothèse que $\mathbf{P} \neq \mathbf{NP}$

Algo M pour VERTEX-COVER

- $C := \emptyset$
- **tant que** $E \neq \emptyset$ **faire**
    - choisir n'importe quelle arête $(u, v) \in E$
    - ajouter u et v à C
    - éliminer tout les arêtes avec u et v de E
- **renvoyer** C

Algo M pour VERTEX-COVER

- $C := \varnothing$
- **tant que** $E \neq \varnothing$ **faire**
    - choisir n'importe quelle arête $(u, v) \in E$
    - ajouter u et v à C
    - éliminer tout les arêtes avec u et v de E
- **renvoyer** C

🤔

M est un algo de $\frac{1}{2}$-approx 😲

- Ça veut dire qu'il renvoie toujours une solution de taille au pire $\frac{1}{1-\varepsilon} = \frac{1}{1-1/2} = 2$ fois la taille d'une solution optimale 👏

- C contient $\frac{1}{2}|C|$ arêtes de G qui ne partagent jamais un sommet

- Chaque couverture, y compris une optimale, doit contenir au moins un sommet de chacune des arêtes de C (sinon il y aurait une arête qui n'est pas couverte), donc $\mathrm{OPT}(G) \geq \frac{1}{2}|C|$

- Et donc $\frac{|c(M(G)) - \mathrm{OPT}(G)|}{\max\{\mathrm{OPT}(G), c(M(G))\}} = \frac{|C| - \mathrm{OPT}(G)}{|C|} \leq \frac{|C| - \frac{1}{2}|C|}{|C|} \leq \frac{1}{2}$

Considerations sur VERTEX-COVER

- Il est possible d'obtenir une solution de coût tout au plus double comparé à une solution optimale en temps polynomial

- Par contre, s'il existe un algo polynomial avec $\varepsilon = 0$, c'est-à-dire qui donne toujours une solution optimale, alors $\mathbf{P} = \mathbf{NP}$

- En effet, on pourrait résoudre DECISION-VERTEX-COVER en temps polynomial en calculant la solution optimale s et en vérifiant si $c(s) \leq k$

Interlude : HAMILTONIAN-CYCLE est $\mathbf{NP}$-complet

HAMILTONIAN-CYCLE

- Entrée : un graphe orienté $G = (V, E)$

- Question : existe-t-il un cycle dans G qui visite chaque sommet exactement une fois (un cycle hamiltonien) ?

HAM-CYCLE $\in$ **NP**

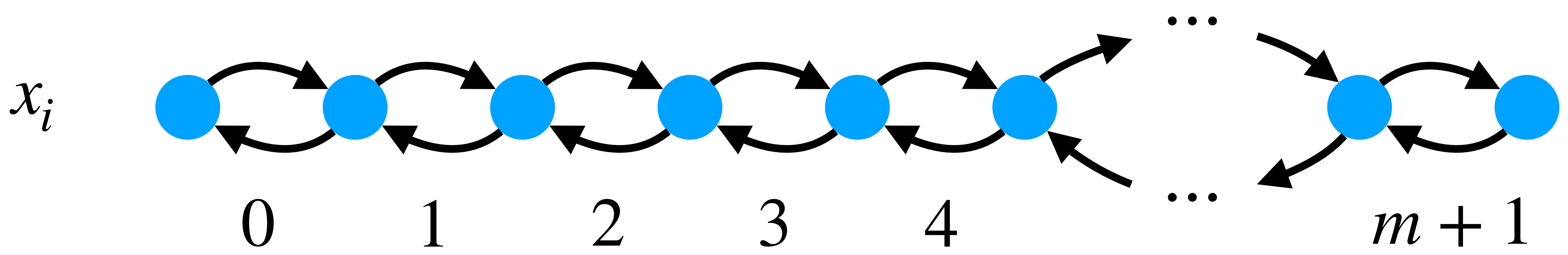
fonction hamiltonien(V, E)
  $n := |V|$
  $perm :=$ tableau(n)
  pour $i := 0$ **à** $n - 1$ **faire**
    $perm[i] :=$ devine($0, \ldots, n - 1$)
  si $perm$ ne contient pas tous les sommets **alors**
    rejeter
  pour $i := 0$ **à** $n - 1$ **faire**
    si $(perm[i], perm[(i + 1) \bmod n]) \notin E$ **alors**
      rejeter
  accepter
fin

3SAT $\leq$ HAM-CYCLE

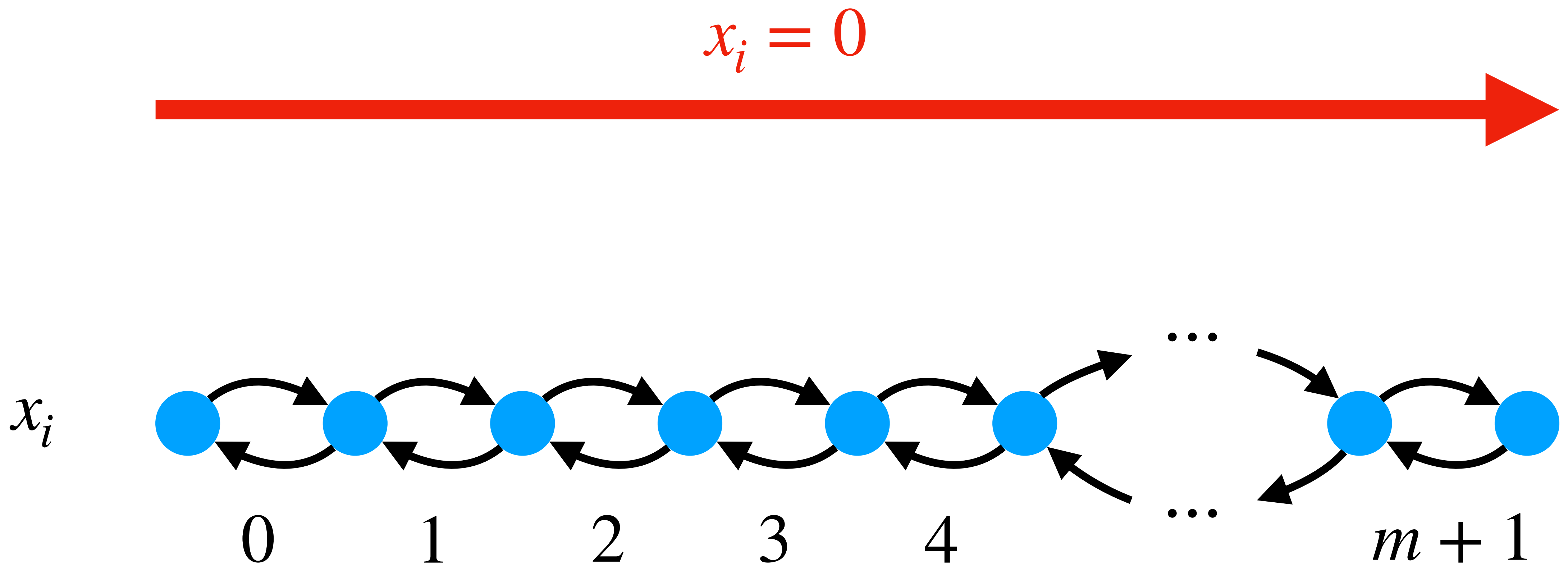
Pour chaque formule $\varphi(x_1, \ldots, x_n) = \varphi_1 \wedge \cdots \wedge \varphi_m$
on construit un graphe ayant

- un gadget pour chaque variable $x_1, \ldots, x_n$
- un gadget pour chaque clause $\varphi_1, \ldots, \varphi_m$
- un cycle hamiltonien ssi la formule est satisfaisable

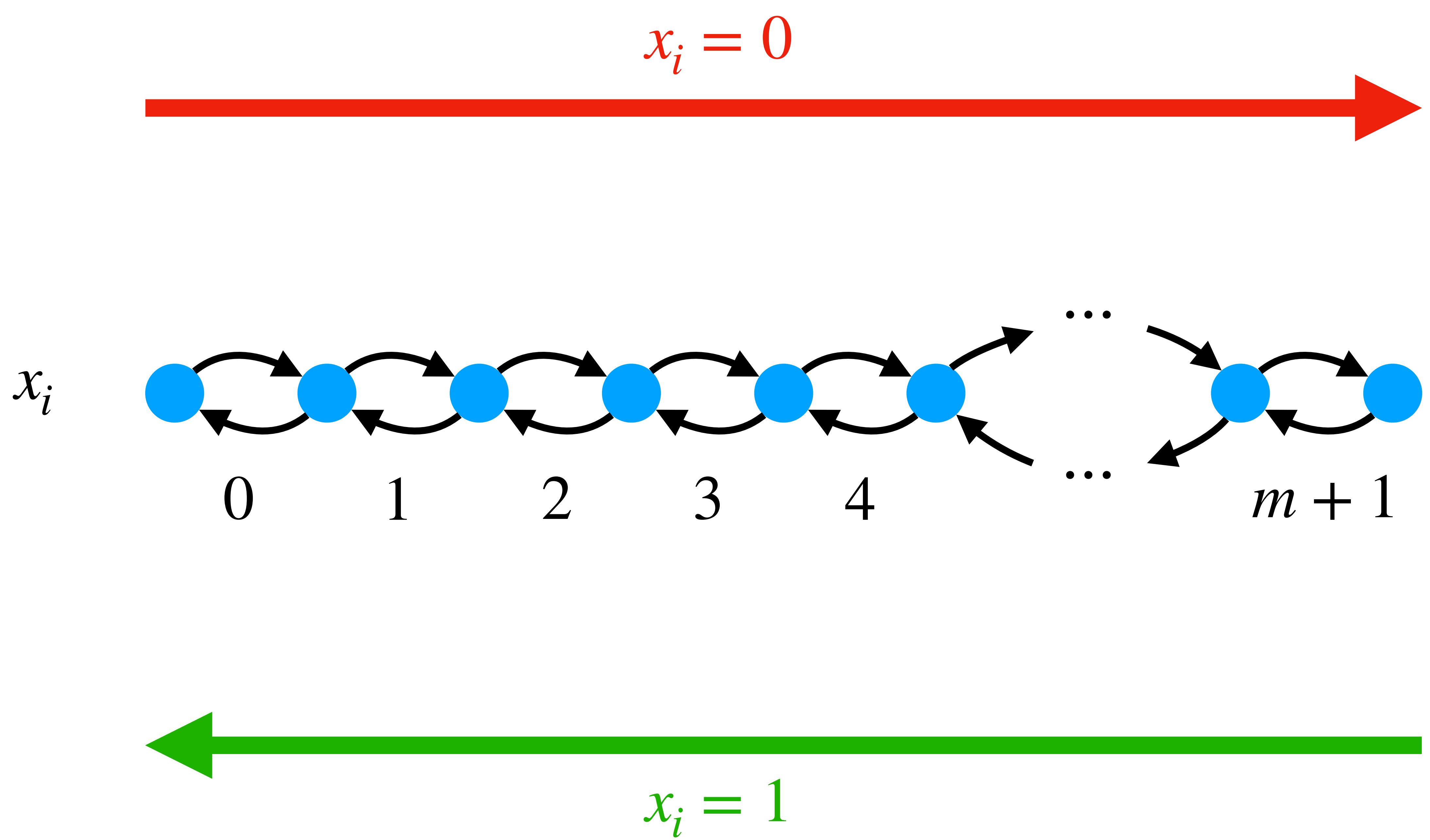
Gadget pour chaque variable x_i



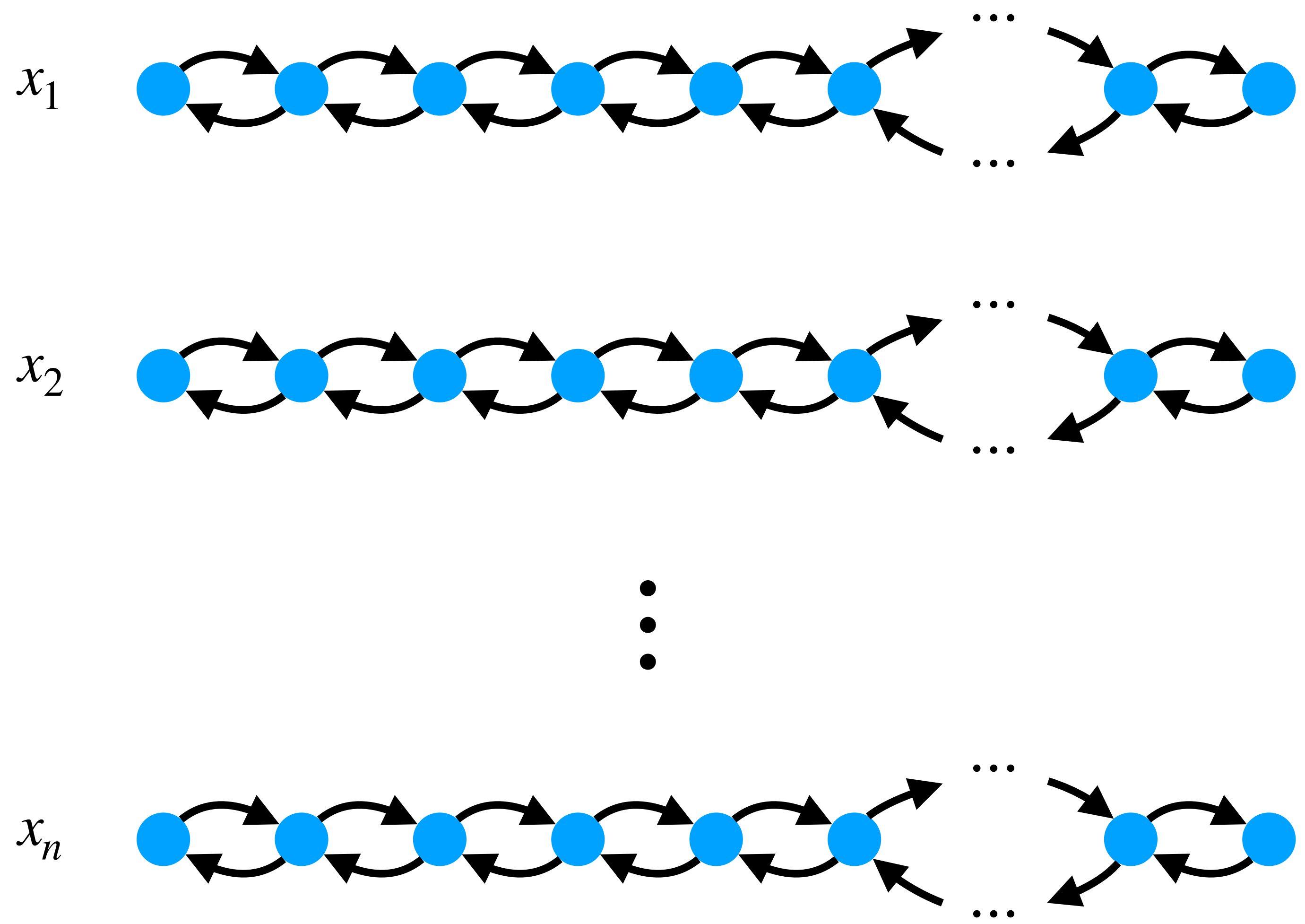
Gadget pour chaque variable x_i



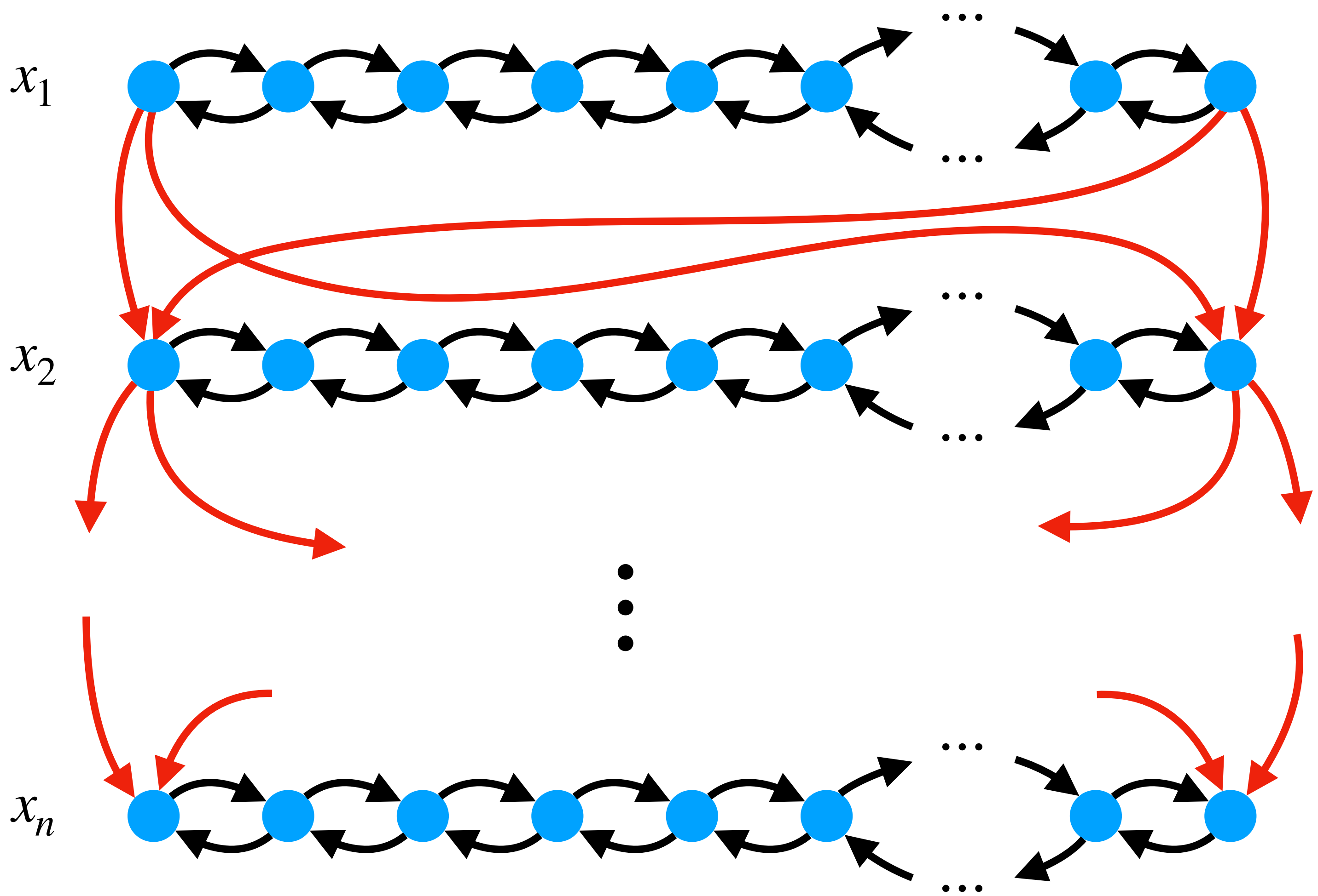
Gadget pour chaque variable x_i



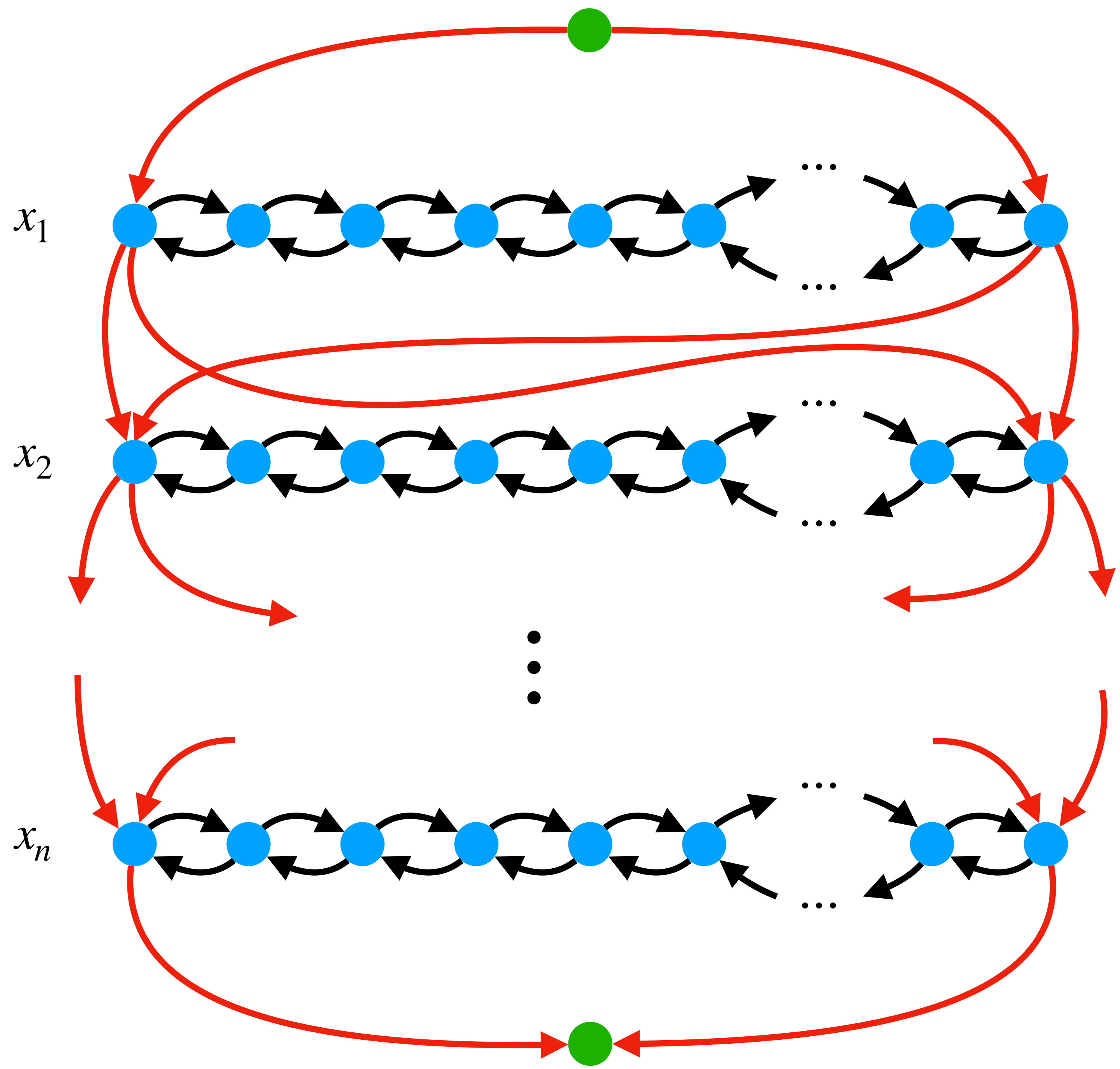
Gadgets pour les variables



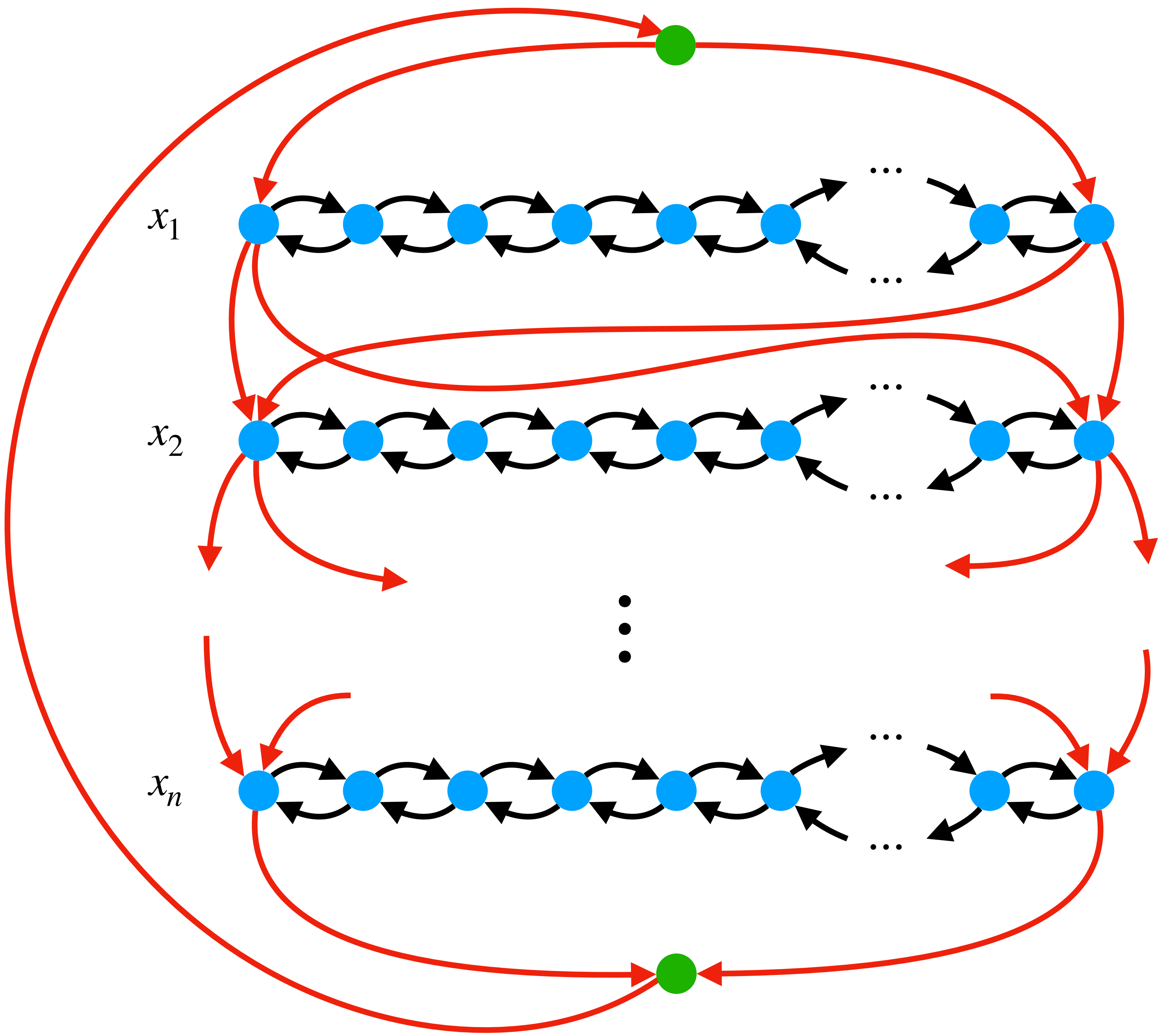
Gadgets pour les variables



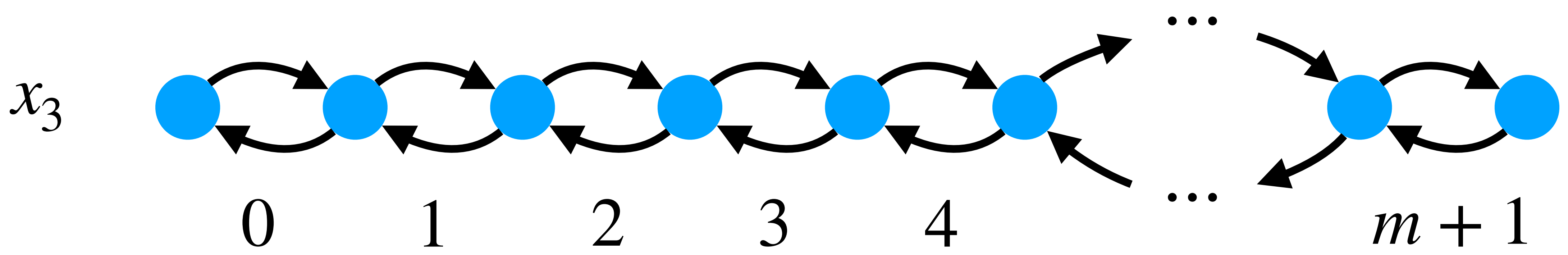
Gadgets pour les variables



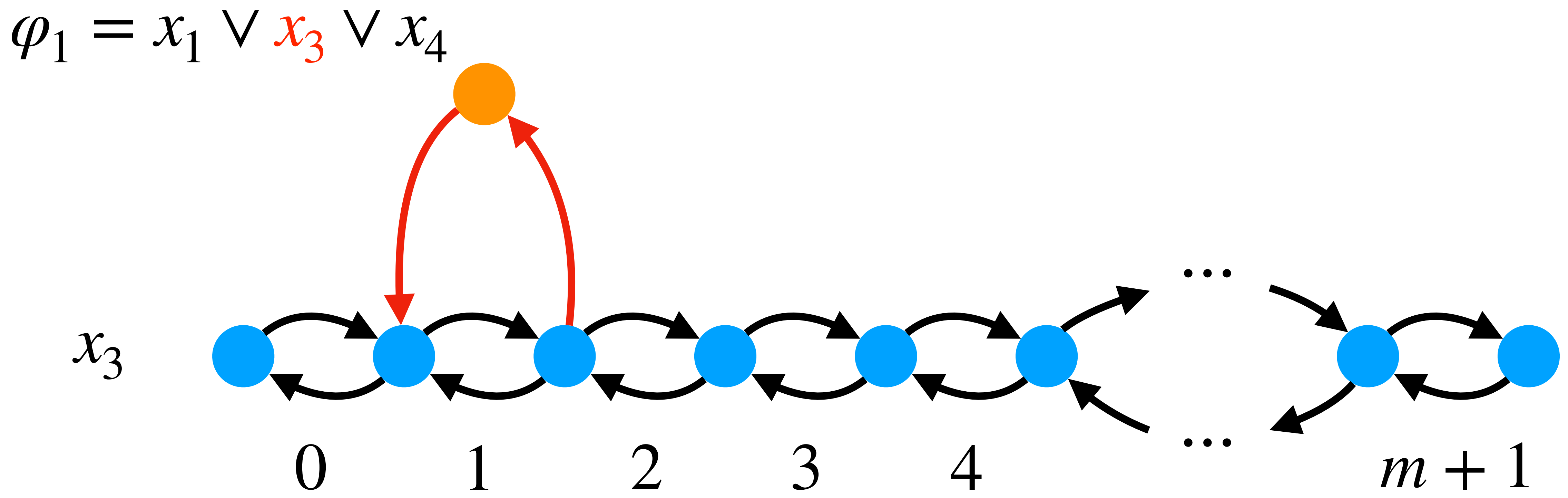
Gadgets pour les variables



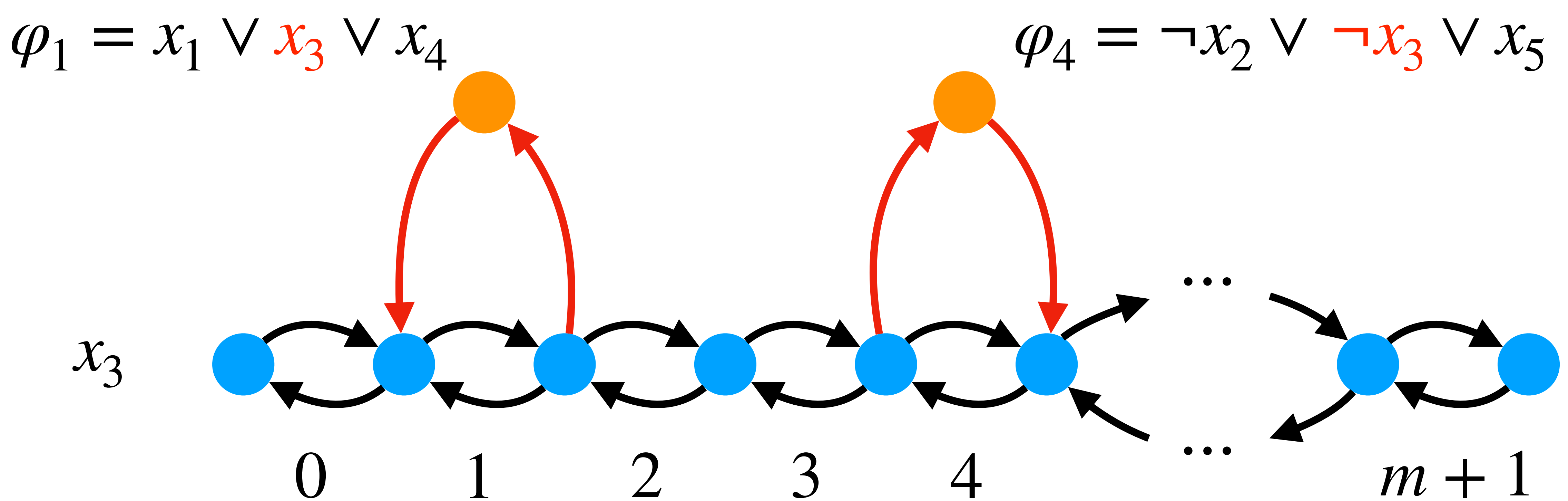
Gadget pour les clauses



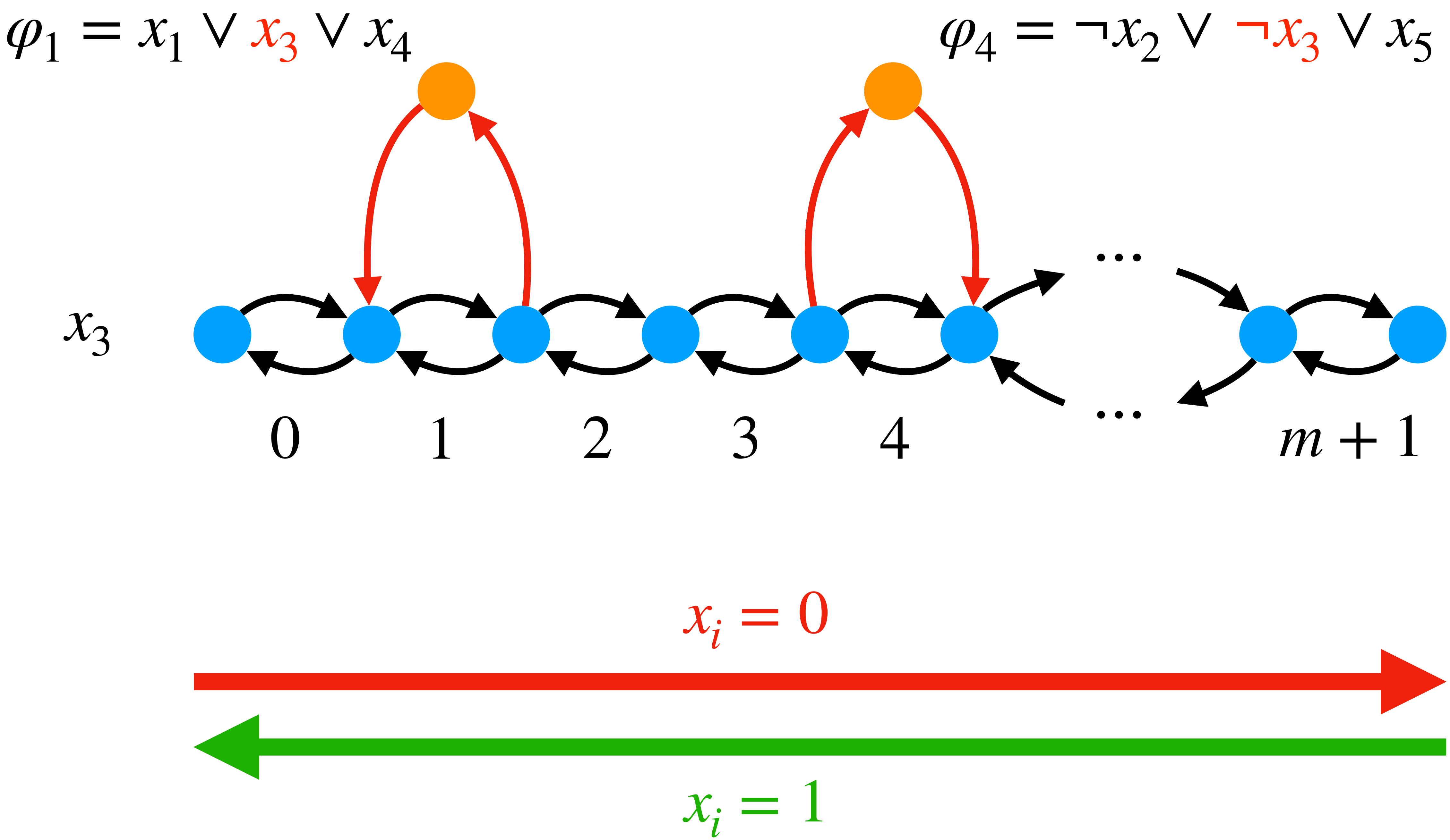
Gadget pour les clauses



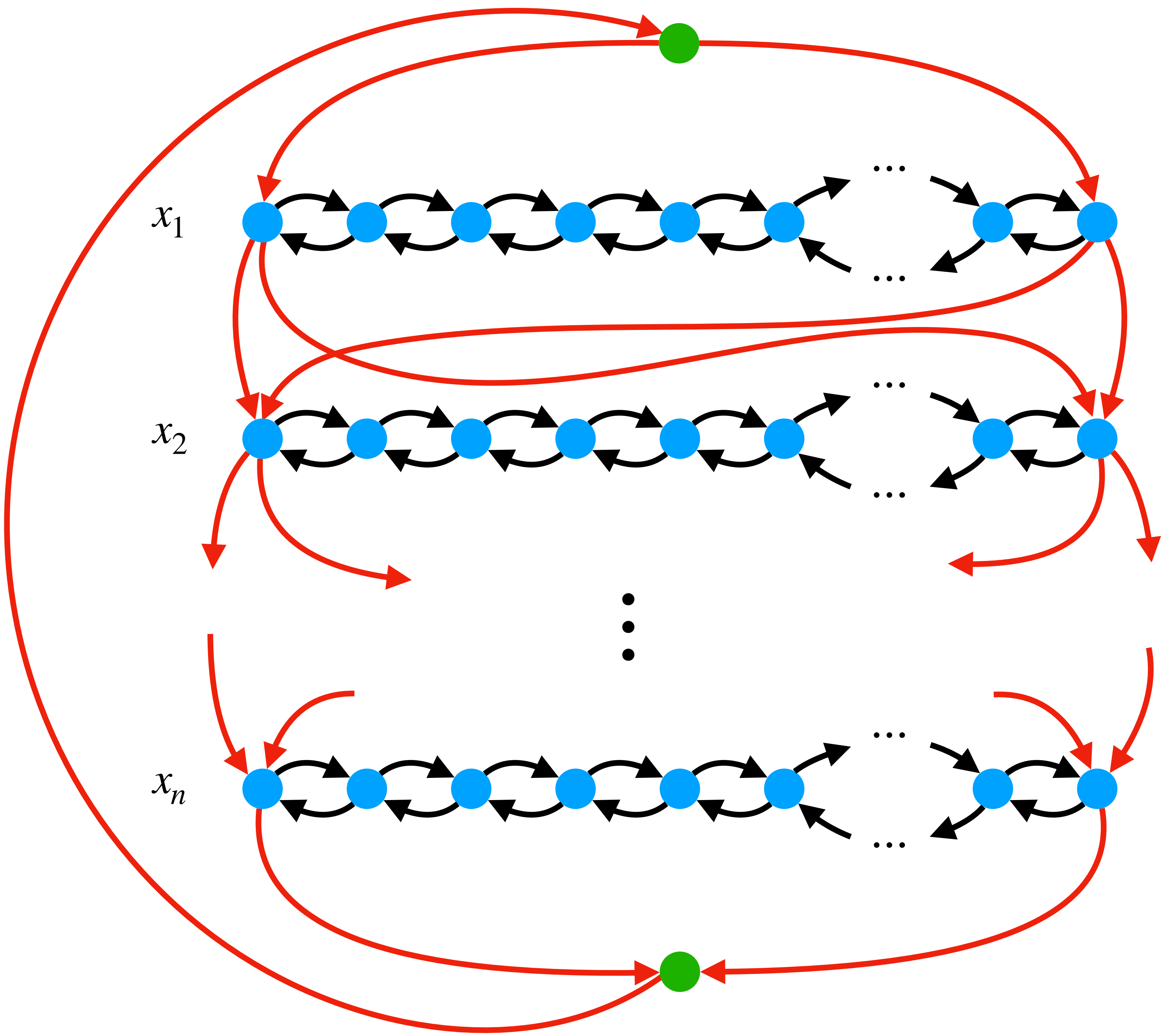
Gadget pour les clauses



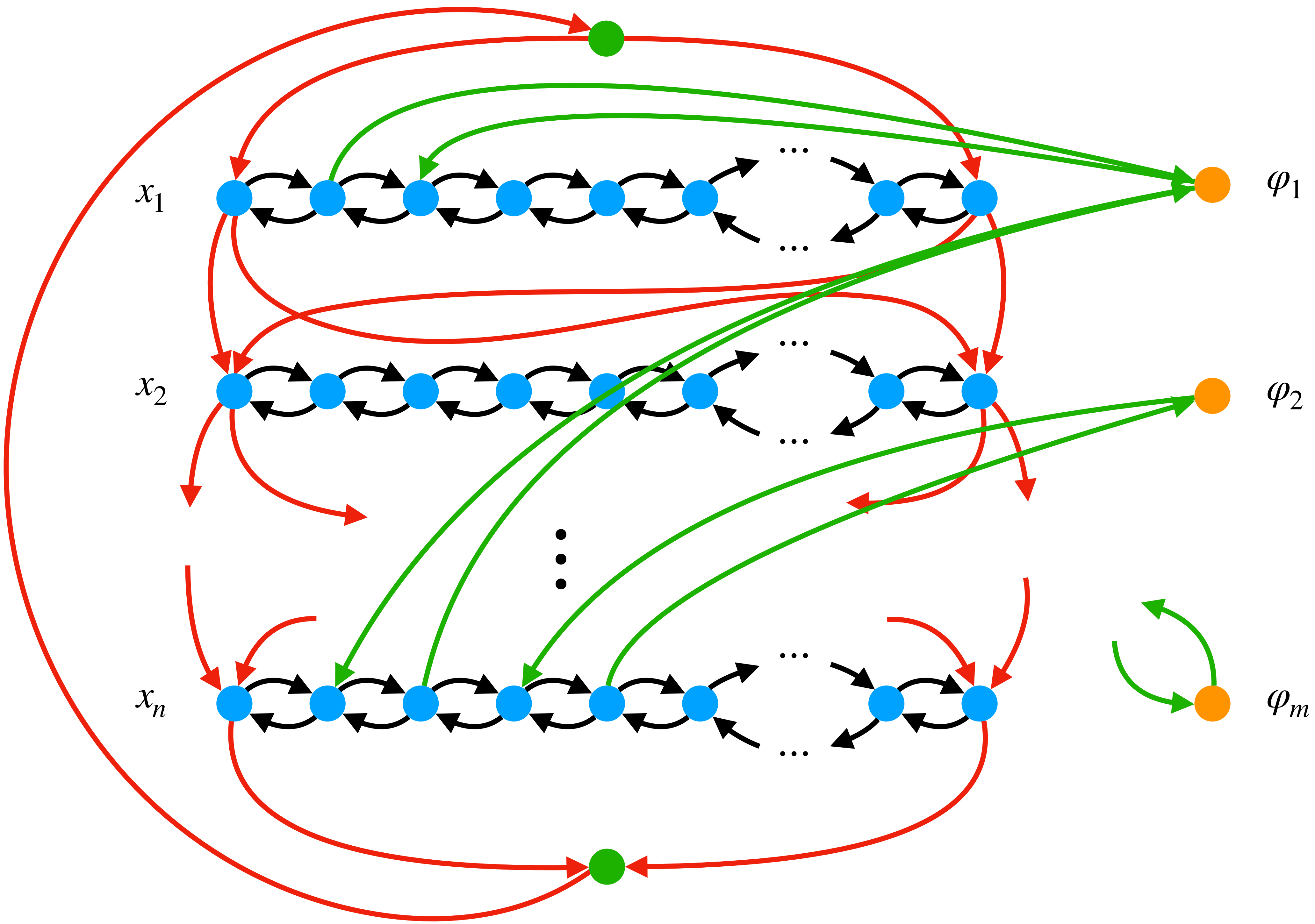
Gadget pour les clauses



Gadgets pour les variables



Gadgets pour les variables



3SAT $\leq$ HAM-CYCLE

- Un chemin hamiltonien code une affectation des variables
- Pour avoir un chemin hamiltonien, il faut visiter tous les sommets correspondents aux clauses
- On peut visiter une clause ssi elle est satisfaite (l’un des littéraux est vrai)
- Donc il y a un cycle hamiltonien ssi la formule est satisfaisable

🧳 est difficile et même pas approximable 😭*

* sous l’hypothèse que $\mathbf{P} \neq \mathbf{NP}$

Théorème

- À moins que $\mathbf{P} = \mathbf{NP}$, le seuil d'approximation pour 🧳 est 1

- Ça veut dire que on n'a aucune garantie du tout sur le coût de la solution trouvé par M par rapport à une solution optimale

- (On sait seulement que $c(M(x)) \leq \infty$… 🙄)

Démonstration

- Par contradiction, soit M un algo de ε-approximation pour 🧳 avec $\varepsilon < 1$

- Avec ça, on va construire un algo polynomial pour HAMILTON-CYCLE (⚠️ c'est une sorte de réduction aussi !)

- Étant donnée une instance $G = (V, E)$ de HAMILTON-CYCLE, on construit un graphe complet $G' = (V, E' = V^2)$ pondéré :

$$w(u, v) = \begin{cases} 1 & \text{si } (u, v) \in E \\ \dfrac{|V|}{1 - \varepsilon} & \text{si } (u, v) \notin E \end{cases}$$

Démonstration

- Par contradiction, soit M un algo de ε-approximation pour 🧳 avec $\varepsilon < 1$

- Avec ça, on va construire un algo polynomial pour HAMILTON-CYCLE (⚠️ c'est une sorte de réduction aussi !)

- Étant donnée une instance $G = (V, E)$ de HAMILTON-CYCLE, on construit un graphe complet $G' = (V, E' = V^2)$ pondéré :

$$w(u, v) = \begin{cases} 1 & \text{si } (u, v) \in E \\ \dfrac{|V|}{1 - \varepsilon} & \text{si } (u, v) \notin E \end{cases}$$

👈

Démonstration

- Maintenant on utilise notre algo de ε-approximation hypothétique M sur le graphe G' pondéré par w

- Si $M(G', w)$ donne une solution de coût $|V|$, alors on ne parcourt que des arêtes originales de G qui, du coup, aura un cycle hamiltonien

Démonstration

- Si, par contre, $M(G', w)$ donne une solution avec au moins une des « nouvelles » arêtes de poids $\frac{|V|}{1-\varepsilon}$, alors le poids de la solution sera $c(M(G', w)) > \frac{|V|}{1-\varepsilon}$

- Vu que M est un algo de ε-approximation, on a $c(M(G', w)) \leq \frac{1}{1-\varepsilon}\mathrm{OPT}(G', w)$, donc

$$\mathrm{OPT}(G', w) \geq (1-\varepsilon)c(M(G', w)) > (1-\varepsilon)\frac{|V|}{1-\varepsilon} = |V|$$

- Donc une solution optimale a coût $> |V|$, ce qui implique que G n'a pas de cycle hamiltonien

Conclusions

- VERTEX-COVER est difficile, mais au moins on peut l'approximer avec un facteur $1/2$
- Par contre, 🧳 en général est difficile et on ne peut même pas l'approximer
- Donc soit on se débrouille avec les algos qu'on connaît, soit on trouve une sous-classe d'instances de 🧳 qu'on peut bien approximer

The End