# Master Informatique - M1 - UE Complexité
# Chapitre 4 : Cadre formel

**Philippe Jégou**

Laboratoire d'Informatique et Systèmes - LIS - UMR CNRS 7020

Équipe COALA - COntraintes, ALgorithmes et Applications

Campus de Saint-Jérôme

Département Informatique et Interactions

Faculté des Sciences

Université d'Aix-Marseille

philippe.jegou@univ-amu.fr

6 septembre 2020

# En préambule

## Ce qui a été vu

- **Problèmes de décision :**

  si un problème de décision est "difficile", alors les problèmes qui en découlent (recherche, optimisation, etc.) le sont

  $\Rightarrow$ **on commence par l'étude des problèmes de décision**

# En préambule

## Ce qui a été vu

- **Problèmes de décision :**

  si un problème de décision est "difficile", alors les problèmes qui en découlent (recherche, optimisation, etc.) le sont

  $\Rightarrow$ **on commence par l'étude des problèmes de décision**

- **Difficulté (i.e. complexité) d'un problème :**

  complexité du meilleur algorithme de résolution

# En préambule

## Ce qui a été vu

- **Problèmes de décision :**

  si un problème de décision est "difficile", alors les problèmes qui en découlent (recherche, optimisation, etc.) le sont

  $\Rightarrow$ **on commence par l'étude des problèmes de décision**

- **Difficulté (i.e. complexité) d'un problème :**

  complexité du meilleur algorithme de résolution

- **Complexité d'un algortihme :**

  fonction de la taille de la donnée en entrée

## Ce qui a été vu

- **Problèmes de décision :**

  si un problème de décision est "difficile", alors les problèmes qui en découlent (recherche, optimisation, etc.) le sont

  $\Rightarrow$ **on commence par l'étude des problèmes de décision**

- **Difficulté (i.e. complexité) d'un problème :**

  complexité du meilleur algorithme de résolution

- **Complexité d'un algortihme :**

  fonction de la taille de la donnée en entrée

- **Taille d'une donnée :**

  liée à son codage

# En préambule

**Ce qui va être vu :**

- **Proposition d'un "système" de codage :**
  - raisonnable

# En préambule

**Ce qui va être vu :**

- **Proposition d'un "système" de codage :**
  - raisonnable
  - permettant de coder tout type de données : des nombres, des noms, des graphes, des bases de données, etc.

# En préambule

**Ce qui va être vu :**

- **Proposition d'un "système" de codage :**
  - raisonnable
  - permettant de coder tout type de données : des nombres, des noms, des graphes, des bases de données, etc.
  - dont la taille des objets codés sera précisément connue :
    - toute donnée sera représentée par un mot au sens des langages formels
    - taille d'une donnée en entree : longueur du mot la codant

# En préambule

**Ce qui va être vu :**

- **Proposition d'un "système" de codage :**
  - raisonnable
  - permettant de coder tout type de données : des nombres, des noms, des graphes, des bases de données, etc.
  - dont la taille des objets codés sera précisément connue :
    - toute donnée sera représentée par un mot au sens des langages formels
    - taille d'une donnée en entree : longueur du mot la codant
- **Résolution d'un problème de décision :**
  - équivaut à un problème de reconnaissance de langage (formel)
  - question à traiter : est-ce qu'un mot appartient à un langage ?

# En préambule

**Ce qui va être vu :**

- **Proposition d'un "système" de codage :**
  - raisonnable
  - permettant de coder tout type de données : des nombres, des noms, des graphes, des bases de données, etc.
  - dont la taille des objets codés sera précisément connue :
    - toute donnée sera représentée par un mot au sens des langages formels
    - taille d'une donnée en entree : longueur du mot la codant
- **Résolution d'un problème de décision :**
  - équivaut à un problème de reconnaissance de langage (formel)
  - question à traiter : est-ce qu'un mot appartient à un langage ?
- **Temps de calcul connu sans ambiguïté :**
  nombre de transitions nécessaires par la "machine" utilisée : automate fini, machine de Turing, etc.

# Plan

1. Problèmes de décision : partition de l'ensemble des instances
2. Codage des instances : c'est facile avec des mots
3. Problèmes de décision : résolution par reconnaissance de langages

# Plan

# Problèmes de décision

## Le problème de l'isomorphisme de sous-graphe

**ISO-SOUS-GRAPHE**
**Donnée** : Deux graphes non-orientés $G_1 = (S_1, A_1)$ et $G_2 = (S_2, A_2)$
**Question** : $G_1$ contient-il un sous-graphe partiel isomorphe à $G_2$ ?

# Problèmes de décision

## Le problème de l'isomorphisme de sous-graphe

**ISO-SOUS-GRAPHE**
**Donnée** : Deux graphes non-orientés $G_1 = (S_1, A_1)$ et $G_2 = (S_2, A_2)$
**Question** : $G_1$ contient-il un sous-graphe partiel isomorphe à $G_2$ ?

$G_1$ contient un sous-graphe partiel $G' = (S', A')$ isomorphe à $G_2$ si
$G' = (S', A')$ sous-graphe partiel de $G_1$ avec $S' \subseteq S_1$ et $A' \subseteq A_1$, tels que :

# Problèmes de décision

## Le problème de l'isomorphisme de sous-graphe

**ISO-SOUS-GRAPHE**
**Donnée** : Deux graphes non-orientés $G_1 = (S_1, A_1)$ et $G_2 = (S_2, A_2)$
**Question** : $G_1$ contient-il un sous-graphe partiel isomorphe à $G_2$ ?

$G_1$ contient un sous-graphe partiel $G' = (S', A')$ isomorphe à $G_2$ si $G' = (S', A')$ sous-graphe partiel de $G_1$ avec $S' \subseteq S_1$ et $A' \subseteq A_1$, tels que :

- $|S'| = |S_2|$ et $|A'| = |A_2|$, et

# Problèmes de décision

## Le problème de l'isomorphisme de sous-graphe

**ISO-SOUS-GRAPHE**
**Donnée** : Deux graphes non-orientés $G_1 = (S_1, A_1)$ et $G_2 = (S_2, A_2)$
**Question** : $G_1$ contient-il un sous-graphe partiel isomorphe à $G_2$ ?

$G_1$ contient un sous-graphe partiel $G' = (S', A')$ isomorphe à $G_2$ si $G' = (S', A')$ sous-graphe partiel de $G_1$ avec $S' \subseteq S_1$ et $A' \subseteq A_1$, tels que :

- $|S'| = |S_2|$ et $|A'| = |A_2|$, et
- $\exists$ une bijection $f : S_2 \rightarrow S'$ vérifiant $\{x, y\} \in A_2 \Leftrightarrow \{f(x), f(y)\} \in A'$

# Problèmes de décision

## Le problème de l'isomorphisme de sous-graphe

**ISO-SOUS-GRAPHE**
**Donnée** : Deux graphes non-orientés $G_1 = (S_1, A_1)$ et $G_2 = (S_2, A_2)$
**Question** : $G_1$ contient-il un sous-graphe partiel isomorphe à $G_2$ ?

$G_1$ contient un sous-graphe partiel $G' = (S', A')$ isomorphe à $G_2$ si $G' = (S', A')$ sous-graphe partiel de $G_1$ avec $S' \subseteq S_1$ et $A' \subseteq A_1$, tels que :

- $|S'| = |S_2|$ et $|A'| = |A_2|$, et
- $\exists$ une bijection $f : S_2 \rightarrow S'$ vérifiant $\{x, y\} \in A_2 \Leftrightarrow \{f(x), f(y)\} \in A'$

**Intérêt pratique** : Est-ce que la "forme" représentée par $G_2$ se trouve dans $G_1$ ? (applications pour la recherche de motifs structurels).

# Problèmes de décision

## Le problème de l'isomorphisme de sous-graphe

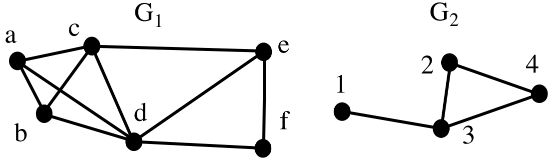
**ISO-SOUS-GRAPHE**
**Donnée** : Deux graphes non-orientés $G_1 = (S_1, A_1)$ et $G_2 = (S_2, A_2)$
**Question** : $G_1$ contient-il un sous-graphe partiel isomorphe à $G_2$ ?

$G_1$ contient un sous-graphe partiel $G' = (S', A')$ isomorphe à $G_2$ si $G' = (S', A')$ sous-graphe partiel de $G_1$ avec $S' \subseteq S_1$ et $A' \subseteq A_1$, tels que :

- $|S'| = |S_2|$ et $|A'| = |A_2|$, et
- $\exists$ une bijection $f : S_2 \rightarrow S'$ vérifiant $\{x, y\} \in A_2 \Leftrightarrow \{f(x), f(y)\} \in A'$

**Intérêt pratique** : Est-ce que la "forme" représentée par $G_2$ se trouve dans $G_1$ ? (applications pour la recherche de motifs structurels).

**Exemple d'instance du problème ISO-SOUS-GRAPHE :** un couple $(G_1, G_2)$




La réponse à la question du problème de décision ici est...

# Problèmes de décision

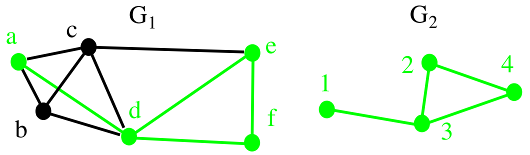
## Le problème de l'isomorphismes de sous-graphe

**ISO-SOUS-GRAPHE**
**Donnée** : Deux graphes non-orientés $G_1 = (S_1, A_1)$ et $G_2 = (S_2, A_2)$
**Question** : $G_1$ contient-il un sous-graphe partiel isomorphe à $G_2$ ?

$G_1$ contient un sous-graphe partiel isomorphe à $G_2$ si $\exists S' \subseteq S_1$ et $\exists A' \subseteq A_1$ tels que :

- $|S'| = |S_2|$ et $|A'| = |A_2|$, et
- $\exists$ une bijection $b : S_2 \to S'$ vérifiant $\{x, y\} \in A_2 \Leftrightarrow \{b(x), b(y)\} \in A'$

**Exemple d'instance du problème ISO-SOUS-GRAPHE :** un couple $(G_1, G_2)$




La réponse à la question du problème de décision ici est **OUI**

$$S' = \{a, d, e, f\} \text{ et } A' = \{\{a, d\}, \{d, e\}, \{d, f\}, \{e, f\}\}$$

et la bijection $b$ est $b(1) = a, b(2) = e, b(3) = d, b(4) = f$

# Problèmes de décision

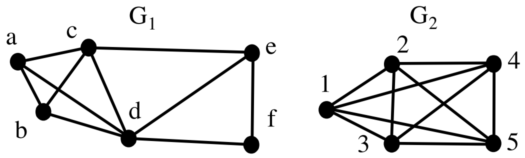
## Le problème de l'isomorphismes de sous-graphe

**ISO-SOUS-GRAPHE**
**Donnée** : Deux graphes non-orientés $G_1 = (S_1, A_1)$ et $G_2 = (S_2, A_2)$
**Question** : $G_1$ contient-il un sous-graphe partiel isomorphe à $G_2$ ?

$G_1$ contient un sous-graphe partiel isomorphe à $G_2$ si $\exists S' \subseteq S_1$ et $\exists A' \subseteq A_1$ tels que :

- $|S'| = |S_2|$ et $|A'| = |A_2|$, et
- $\exists$ une bijection $b : S_2 \rightarrow S'$ vérifiant $\{x, y\} \in A_2 \Leftrightarrow \{b(x), b(y)\} \in A'$

**Exemple d'instance du problème ISO-SOUS-GRAPHE :** un couple $(G_1, G_2)$




Si la question est posée avec un autre graphe $G_2$

qui est un graphe complet à 5 sommets :

La réponse à la question du problème de décision ici est **NON**

# Problèmes de décision

**L'ensemble des instances du problème ISO-SOUS-GRAPHE** peut se partitionner en

- 1 sous-ensemble des **instances positives** : la réponse à la question du problème de décision pour ces instances est **OUI**
- 1 sous-ensemble des **instances négatives** : la réponse à la question du problème de décision pour ces instances est **NON**

# Problèmes de décision

**L'ensemble des instances du problème ISO-SOUS-GRAPHE** peut se partitionner en

- 1 sous-ensemble des **instances positives** : la réponse à la question du problème de décision pour ces instances est **OUI**
- 1 sous-ensemble des **instances négatives** : la réponse à la question du problème de décision pour ces instances est **NON**

**Plus généralement**, pour tout problème de décision $\pi$ on a :

- $D_\pi$ : l'ensemble des instances du problème $\pi$

qui contient deux sous-ensembles disjoints :

- $V_\pi$ : l'ensemble des instances positives du problème $\pi$ ($V_\pi \subseteq D_\pi$)
- $D_\pi \setminus V_\pi$ : l'ensemble des instances négatives du problème $\pi$

# Problèmes de décision

**L'ensemble des instances du problème ISO-SOUS-GRAPHE** peut se partitionner en

- 1 sous-ensemble des **instances positives** : la réponse à la question du problème de décision pour ces instances est **OUI**
- 1 sous-ensemble des **instances négatives** : la réponse à la question du problème de décision pour ces instances est **NON**

**Plus généralement**, pour tout problème de décision $\pi$ on a :

- $D_\pi$ : l'ensemble des instances du problème $\pi$

qui contient deux sous-ensembles disjoints :

- $V_\pi$ : l'ensemble des instances positives du problème $\pi$ ($V_\pi \subseteq D_\pi$)
- $D_\pi \backslash V_\pi$ : l'ensemble des instances négatives du problème $\pi$

**Conséquence :** résoudre un problème de décision $\pi$, étant donnée une instance $I \in D_\pi$ consiste à savoir :

- si $I \in V_\pi$ (instance positive : la réponse à la question est oui)
- si $I \in D_\pi \backslash V_\pi$ (instance négative : la réponse à la question est non)

# Plan

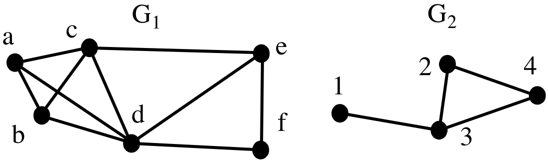
# Codage des instances

**Question :** comment coder des instances "complexes" par des mots ?

("mot" : à prendre au sens de la Théorie des Langages Formels)

**Exemple d'instance "complexes" :**

Une instance du problème ISO-SOUS-GRAPHE : un couple ($G_1$, $G_2$)



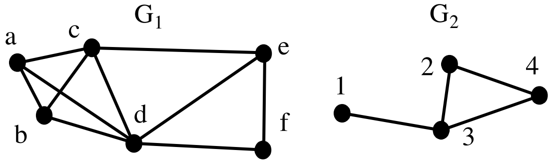
# Codage des instances

**Question :** comment coder des instances "complexes" par des mots ?

("mot" : à prendre au sens de la Théorie des Langages Formels)

**Exemple d'instance "complexes" :**

Une instance du problème ISO-SOUS-GRAPHE : un couple $(G_1, G_2)$




**Réponse :** à l'aide de "systèmes de codage" !

# Systèmes de codage

## Définition

On appelle **système de codage** $S$ pour un problème de décision $\pi$, la manière de décrire chaque donnée de $\pi$ par un mot approprié sur un alphabet fixé $\Sigma$.

# Systèmes de codage

## Définition

On appelle **système de codage** $S$ pour un problème de décision $\pi$, la manière de décrire chaque donnée de $\pi$ par un mot approprié sur un alphabet fixé $\Sigma$.

Les systèmes de codage devront être "**raisonnables**" :

- **décodables** : décodage faisable en temps polynomial sur modèle de calcul déterministe
- **concis** : la taille du codage de l'information en entrée ne doit pas être excessive (par exemple ne pas coder un entier en codage unaire)

# Systèmes de codage

**Définition**

On appelle **système de codage** $S$ pour un problème de décision $\pi$, la manière de décrire chaque donnée de $\pi$ par un mot approprié sur un alphabet fixé $\Sigma$.

Les systèmes de codage devront être "**raisonnables**" :

- **décodables** : décodage faisable en temps polynomial sur modèle de calcul déterministe
- **concis** : la taille du codage de l'information en entrée ne doit pas être excessive (par exemple ne pas coder un entier en codage unaire)

Tout codage raisonnable aura une taille "**polynomialement équivalente**" à celle du "**codage standard**" que nous allons définir

# Systèmes de codage

**Avant d'en parler :** rappels sur la notion de langage formel vue en L2 !
(avec une diapositive du cours de L2...)

# Systèmes de codage

**Avant d'en parler :** rappels sur la notion de langage formel vue en L2 !
(avec une diapositive du cours de L2...)

## Définition

### Symboles et mots

- Les symboles sont des éléments indivisibles qui vont servir de briques de base pour construire des mots.
- Un alphabet est un ensemble fini de symboles. On désigne conventionnellement un alphabet par la lettre grecque $\Sigma$.
- Une suite de symboles, appartenant à un alphabet $\Sigma$, mis bout à bout est appelé un mot (ou une *chaîne*) sur $\Sigma$.
- On note $|m|$ la longueur du mot $m$ (le nombre de symboles qui le composent).
- On note $|m|_s$ le nombre de symboles $s$ que possède le mot $m$.
- Le mot de longueur zéro, appelé mot vide, est noté $\varepsilon$.
- Si $m$ est un mot et $1 \leq i \leq |m|$, on note $m[i]$ le $i^{eme}$ symbole de $m$.

**Attention :** un mot est suite finie de symboles

# Un systèmes de codage : le codage standard

## Définition

Le Codage Standard est défini sur l'alphabet $\Sigma = \{0, 1, -, [, ], (, ), ;\}$ par des mots structurés comme suit :

# Un systèmes de codage : le codage standard

## Définition

Le Codage Standard est défini sur l'alphabet $\Sigma = \{0, 1, -, [, ], (, ), ;\}$ par des mots structurés comme suit :

- **entiers relatifs** : codage binaire sans bit redondant avec poids forts à gauche et précédé du symbole '$-$' pour les nombres négatifs.

# Un systèmes de codage : le codage standard

## Définition

Le Codage Standard est défini sur l'alphabet $\Sigma = \{0, 1, -, [, ], (, ), ;\}$ par des mots structurés comme suit :

- **entiers relatifs** : codage binaire sans bit redondant avec poids forts à gauche et précédé du symbole '$-$' pour les nombres négatifs.
- **noms** : on associe un entier $k$ au nom et le codage est $[u]$ où $u$ code l'entier $k$

# Un systèmes de codage : le codage standard

## Définition

Le Codage Standard est défini sur l'alphabet $\Sigma = \{0, 1, -, [, ], (, ), ;\}$ par des mots structurés comme suit :

- **entiers relatifs** : codage binaire sans bit redondant avec poids forts à gauche et précédé du symbole '$-$' pour les nombres négatifs.
- **noms** : on associe un entier $k$ au nom et le codage est $[u]$ où $u$ code l'entier $k$
- **k-uplets** : le k-uplet d'éléments codés $u_i$ et codé par le mot $(u_1; u_2; \ldots u_k)$

# Un systèmes de codage : le codage standard

## Définition

Le Codage Standard est défini sur l'alphabet $\Sigma = \{0, 1, -, [, ], (, ), ;\}$ par des mots structurés comme suit :

- **entiers relatifs** : codage binaire sans bit redondant avec poids forts à gauche et précédé du symbole '$-$' pour les nombres négatifs.
- **noms** : on associe un entier $k$ au nom et le codage est $[u]$ où $u$ code l'entier $k$
- **k-uplets** : le k-uplet d'éléments codés $u_i$ et codé par le mot $(u_1; u_2; \dots u_k)$
- **rationnels** : pour le rationnel $q = \frac{x}{y}$ où $x$ et $y$ sont des entiers relatifs, le codage est $(u_1; u_2)$ si $u_1$ code $x$ et si $u_2$ code $y$.

# Un systèmes de codage : le codage standard

## Définition

Le Codage Standard est défini sur l'alphabet $\Sigma = \{0, 1, -, [, ], (, ), ;\}$ par des mots structurés comme suit :

- **entiers relatifs** : codage binaire sans bit redondant avec poids forts à gauche et précédé du symbole '$-$' pour les nombres négatifs.
- **noms** : on associe un entier $k$ au nom et le codage est $[u]$ où $u$ code l'entier $k$
- **k-uplets** : le k-uplet d'éléments codés $u_i$ et codé par le mot $(u_1; u_2; \ldots u_k)$
- **rationnels** : pour le rationnel $q = \frac{x}{y}$ où $x$ et $y$ sont des entiers relatifs, le codage est $(u_1; u_2)$ si $u_1$ code $x$ et si $u_2$ code $y$.
- **fonctions finies** : la fonction $f : E \rightarrow F$ (avec $E$ et $F$ deux ensembles finis) est codée par un k-uplet $(([\ldots]; [\ldots]); ([\ldots]; [\ldots]); \ldots ([\ldots]; [\ldots]))$ où tout couple $([\ldots]; [\ldots])$ est de la forme $([u]; [v])$ où $u$ code un objet $x \in E$ et $v$ code un objet $y \in F$ tels que $f(x) = y$

# Un systèmes de codage : le codage standard

## Définition

Le Codage Standard est défini sur l'alphabet $\Sigma = \{0, 1, -, [, ], (, ), ;\}$ par des mots structurés comme suit :

- **entiers relatifs** : codage binaire sans bit redondant avec poids forts à gauche et précédé du symbole '$-$' pour les nombres négatifs.
- **noms** : on associe un entier $k$ au nom et le codage est $[u]$ où $u$ code l'entier $k$
- **k-uplets** : le k-uplet d'éléments codés $u_i$ et codé par le mot $(u_1; u_2; \dots u_k)$
- **rationnels** : pour le rationnel $q = \frac{x}{y}$ où $x$ et $y$ sont des entiers relatifs, le codage est $(u_1; u_2)$ si $u_1$ code $x$ et si $u_2$ code $y$.
- **fonctions finies** : la fonction $f : E \rightarrow F$ (avec $E$ et $F$ deux ensembles finis) est codée par un k-uplet $(([\dots]; [\dots]); ([\dots]; [\dots]); \dots ([\dots]; [\dots]))$ où tout couple $([\dots]; [\dots])$ est de la forme $([u]; [v])$ où $u$ code un objet $x \in E$ et $v$ code un objet $y \in F$ tels que $f(x) = y$
- **graphes** : le graphe $G = (S, A)$ est codé par un couple $(u\ ;\ v)$ où $u$ représente le codage de l'ensemble de sommets et $v$ celui de l'ensemble d'arcs (ou d'arêtes) dans lequel chaque arc (ou arête) est codé par un 2-uplet

# Un exemple avec le codage standard

**Codage** d'une instance du problème ISO-SOUS-GRAPHE

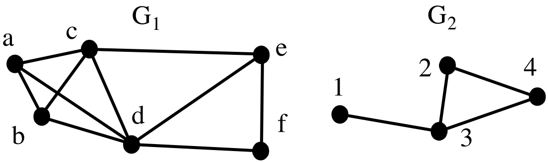



Illustration avec le codage du graphe $G_2 = (S_2, A_2)$ :

# Un exemple avec le codage standard

**Codage** d'une instance du problème ISO-SOUS-GRAPHE

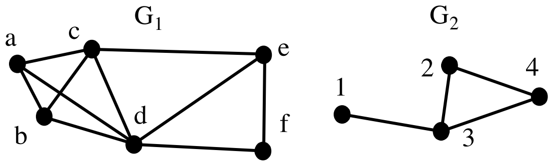



Illustration avec le codage du graphe $G_2 = (S_2, A_2)$ :

- codage des sommets (les noms des sommets sont des entiers) : 3 est codé par [11]

# Un exemple avec le codage standard

**Codage** d'une instance du problème ISO-SOUS-GRAPHE

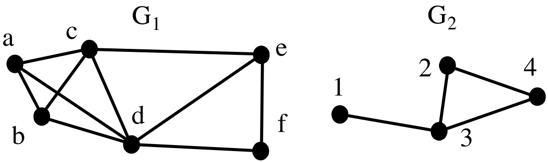



Illustration avec le codage du graphe $G_2 = (S_2, A_2)$ :

- codage des sommets (les noms des sommets sont des entiers) : 3 est codé par [11]
- ensemble des sommets $S_2 = \{1, 2, 3, 4\}$ codé par le 4-uplet ([1];[10];[11];[100])

# Un exemple avec le codage standard

**Codage** d'une instance du problème ISO-SOUS-GRAPHE

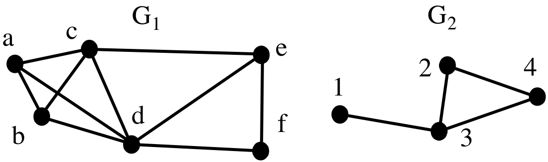



Illustration avec le codage du graphe $G_2 = (S_2, A_2)$ :

- codage des sommets (les noms des sommets sont des entiers) : 3 est codé par [11]
- ensemble des sommets $S_2 = \{1, 2, 3, 4\}$ codé par le 4-uplet ([1];[10];[11];[100])
- chaque arête est codée par un 2-uplet : l'arête $\{2, 4\}$ est codée par ( [10] ; [100] )

# Un exemple avec le codage standard

**Codage** d'une instance du problème ISO-SOUS-GRAPHE

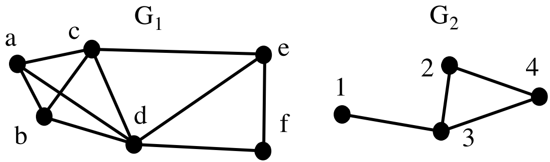



Illustration avec le codage du graphe $G_2 = (S_2, A_2)$ :

- codage des sommets (les noms des sommets sont des entiers) : 3 est codé par [11]
- ensemble des sommets $S_2 = \{1, 2, 3, 4\}$ codé par le 4-uplet ([1];[10];[11];[100])
- chaque arête est codée par un 2-uplet : l'arête $\{2, 4\}$ est codée par ( [10] ; [100] )
- l'ensemble d'arêtes $A_2 = \{\{1, 3\}, \{2, 3\}, \{2, 4\}, \{3, 4\}\}$ par un 4-uplet car $|A_2| = 4$ : le codage de $A_2$ est donc ( ([1];[11])

# Un exemple avec le codage standard

**Codage** d'une instance du problème ISO-SOUS-GRAPHE

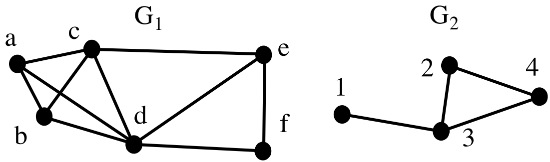



Illustration avec le codage du graphe $G_2 = (S_2, A_2)$ :

- codage des sommets (les noms des sommets sont des entiers) : 3 est codé par [11]
- ensemble des sommets $S_2 = \{1, 2, 3, 4\}$ codé par le 4-uplet ([1];[10];[11];[100])
- chaque arête est codée par un 2-uplet : l'arête $\{2, 4\}$ est codée par ( [10] ; [100] )
- l'ensemble d'arêtes $A_2 = \{\{1, 3\}, \{2, 3\}, \{2, 4\}, \{3, 4\}\}$ par un 4-uplet car $|A_2| = 4$ : le codage de $A_2$ est donc ( ([1];[11]) ; ([10];[11])

# Un exemple avec le codage standard

**Codage** d'une instance du problème ISO-SOUS-GRAPHE

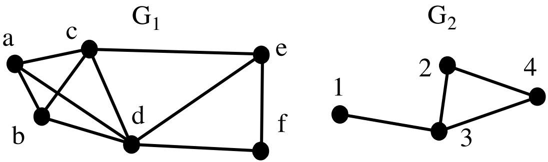



Illustration avec le codage du graphe $G_2 = (S_2, A_2)$ :

- codage des sommets (les noms des sommets sont des entiers) : 3 est codé par [11]
- ensemble des sommets $S_2 = \{1, 2, 3, 4\}$ codé par le 4-uplet ([1];[10];[11];[100])
- chaque arête est codée par un 2-uplet : l'arête $\{2, 4\}$ est codée par ( [10] ; [100] )
- l'ensemble d'arêtes $A_2 = \{\{1, 3\}, \{2, 3\}, \{2, 4\}, \{3, 4\}\}$ par un 4-uplet car $|A_2| = 4$ :
  le codage de $A_2$ est donc ( ([1];[11]) ; ([10];[11]) ; ([10];[100])

# Un exemple avec le codage standard

**Codage** d'une instance du problème ISO-SOUS-GRAPHE

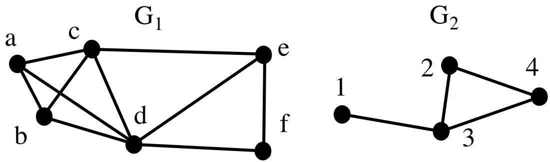



Illustration avec le codage du graphe $G_2 = (S_2, A_2)$ :

- codage des sommets (les noms des sommets sont des entiers) : 3 est codé par [11]
- ensemble des sommets $S_2 = \{1, 2, 3, 4\}$ codé par le 4-uplet ([1];[10];[11];[100])
- chaque arête est codée par un 2-uplet : l'arête $\{2, 4\}$ est codée par ( [10] ; [100] )
- l'ensemble d'arêtes $A_2 = \{\{1, 3\}, \{2, 3\}, \{2, 4\}, \{3, 4\}\}$ par un 4-uplet car $|A_2| = 4$ :
  le codage de $A_2$ est donc ( ([1];[11]) ; ([10];[11]) ; ([10];[100]) ; ([11];[100]) )

# Un exemple avec le codage standard

**Codage** d'une instance du problème ISO-SOUS-GRAPHE

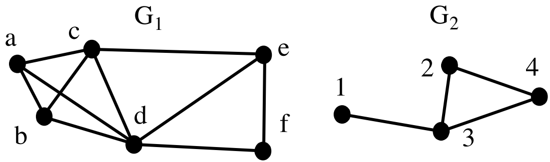



Illustration avec le codage du graphe $G_2 = (S_2, A_2)$ :

- codage des sommets (les noms des sommets sont des entiers) : 3 est codé par [11]
- ensemble des sommets $S_2 = \{1, 2, 3, 4\}$ codé par le 4-uplet ([1];[10];[11];[100])
- chaque arête est codée par un 2-uplet : l'arête $\{2, 4\}$ est codée par ( [10] ; [100] )
- l'ensemble d'arêtes $A_2 = \{\{1, 3\}, \{2, 3\}, \{2, 4\}, \{3, 4\}\}$ par un 4-uplet car $|A_2| = 4$ :
  le codage de $A_2$ est donc ( ([1];[11]) ; ([10];[11]) ; ([10];[100]) ; ([11];[100]) )
- le graphe $G_2 = (S_2, A_2)$ est donc codé par un couple :
  ( ( [1];[10];[11];[100] )

# Un exemple avec le codage standard

**Codage** d'une instance du problème ISO-SOUS-GRAPHE

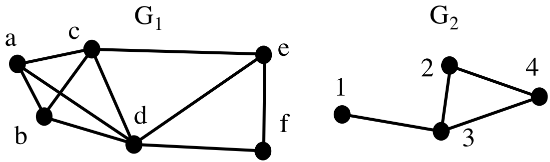



Illustration avec le codage du graphe $G_2 = (S_2, A_2)$ :

- codage des sommets (les noms des sommets sont des entiers) : 3 est codé par [11]
- ensemble des sommets $S_2 = \{1, 2, 3, 4\}$ codé par le 4-uplet ([1];[10];[11];[100])
- chaque arête est codée par un 2-uplet : l'arête $\{2, 4\}$ est codée par ( [10] ; [100] )
- l'ensemble d'arêtes $A_2 = \{\{1, 3\}, \{2, 3\}, \{2, 4\}, \{3, 4\}\}$ par un 4-uplet car $|A_2| = 4$ :
  le codage de $A_2$ est donc ( ([1];[11]) ; ([10];[11]) ; ([10];[100]) ; ([11];[100]) )
- le graphe $G_2 = (S_2, A_2)$ est donc codé par un couple :
  ( ( [1];[10];[11];[100] ) ; ( ([1];[11]) ; ([10];[11]) ; ([10];[100]) ; ([11];[100]) ) )

et si on code $G_1 = (S_1, A_1)$, les sommets seront codés par exemple avec le code ASCII (car leurs noms sont des lettres)

# Sur le codage standard

- On peut donc coder des données non triviales comme des graphes

# Sur le codage standard

- On peut donc coder des données non triviales comme des graphes

  Mais on peut aussi coder des objets bien plus complexes !

# Sur le codage standard

- On peut donc coder des données non triviales comme des graphes

  Mais on peut aussi coder des objets bien plus complexes !

  par exemple une **base de données relationnelle** car c'est :

  - un ensemble (k-uplet) de tables

# Sur le codage standard

- On peut donc coder des données non triviales comme des graphes

  Mais on peut aussi coder des objets bien plus complexes !

  par exemple une **base de données relationnelle** car c'est :

  - un ensemble (k-uplet) de tables
  - et chaque table est un ensemble (k-uplet) de tuples

# Sur le codage standard

- On peut donc coder des données non triviales comme des graphes

  Mais on peut aussi coder des objets bien plus complexes !

  par exemple une **base de données relationnelle** car c'est :

  - un ensemble (k-uplet) de tables
  - et chaque table est un ensemble (k-uplet) de tuples
  - et chaque tuple est un k-uplet

# Sur le codage standard

- On peut donc coder des données non triviales comme des graphes

  Mais on peut aussi coder des objets bien plus complexes !

  par exemple une **base de données relationnelle** car c'est :
  - un ensemble (k-uplet) de tables
  - et chaque table est un ensemble (k-uplet) de tuples
  - et chaque tuple est un k-uplet

**Rien de surprenant :** sur un ordinateur, toute donnée se retrouve toujours en mémoire par une séquence de 0 et de 1 !

# Sur le codage standard

- On peut donc coder des données non triviales comme des graphes

  Mais on peut aussi coder des objets bien plus complexes !

  par exemple une **base de données relationnelle** car c'est :
  - un ensemble (k-uplet) de tables
  - et chaque table est un ensemble (k-uplet) de tuples
  - et chaque tuple est un k-uplet

**Rien de surprenant :** sur un ordinateur, toute donnée se retrouve toujours en mémoire par une séquence de 0 et de 1 !

- Pour une même donnée, en considérant un même système de codage, il existe en général plusieurs codages

  Par exemple ([1];[10];[11];[100]) et ([11];[1];[100];[10]) codent le même ensemble de sommets $S_2 = \{1, 2, 3, 4\}$ du graphe $G_2$

# Sur le codage standard

- On peut donc coder des données non triviales comme des graphes

  Mais on peut aussi coder des objets bien plus complexes !

  par exemple une **base de données relationnelle** car c'est :
  - un ensemble (k-uplet) de tables
  - et chaque table est un ensemble (k-uplet) de tuples
  - et chaque tuple est un k-uplet

**Rien de surprenant :** sur un ordinateur, toute donnée se retrouve toujours en mémoire par une séquence de 0 et de 1 !

- Pour une même donnée, en considérant un même système de codage, il existe en général plusieurs codages

  Par exemple ([1];[10];[11];[100]) et ([11];[1];[100];[10]) codent le même ensemble de sommets $S_2 = \{1, 2, 3, 4\}$ du graphe $G_2$

- Le système de "Codage Standard" n'est qu'une proposition de codage raisonnable parmi d'autres

  Par exemple remplaçer les crochets par des accolades ou le binaire par du décimal ne changent rien de fondamental au niveau des propriétés du codage.

# Le codage standard CS et les autres codages raisonnables

Le codage standard CS sert de référence pour les codages raisonnables

(sous la réserve qu'ils offrent aussi la "décodabilité")

# Le codage standard CS et les autres codages raisonnables

Le codage standard CS sert de référence pour les codages raisonnables
(sous la réserve qu'ils offrent aussi la "décodabilité")

## Définition

Un système de codage $S$ pour un problème de décision $\pi$ est dit **raisonnable** si pour toute instance $I$ de $\pi$, la longueur de $I$ dans $S$ est **polynomialement équivalente** à la longueur de $I$ dans le codage standard $CS$

# Le codage standard CS et les autres codages raisonnables

Le codage standard CS sert de référence pour les codages raisonnables

(sous la réserve qu'ils offrent aussi la "décodabilité")

## Définition

Un système de codage $S$ pour un problème de décision $\pi$ est dit **raisonnable** si pour toute instance $I$ de $\pi$, la longueur de $I$ dans $S$ est **polynomialement équivalente** à la longueur de $I$ dans le codage standard $CS$, i.e. $\exists p, p'$ deux polynômes tels que $\forall I \in D_\pi$

- $|Codage_S(I)| \leq p(\ |Codage_{CS}(I)|\ )$ et
- $|Codage_{CS}(I)| \leq p'(\ |Codage_S(I)|\ )$

où $Codage_S(I)$ et $Codage_{CS}(I)$ désignent les codages de l'instance $I$ par $S$ et $CS$.

# Le codage standard CS et les autres codages raisonnables

Le codage standard CS sert de référence pour les codages raisonnables

(sous la réserve qu'ils offrent aussi la "décodabilité")

## Définition

Un système de codage $S$ pour un problème de décision $\pi$ est dit **raisonnable** si pour toute instance $I$ de $\pi$, la longueur de $I$ dans $S$ est **polynomialement équivalente** à la longueur de $I$ dans le codage standard $CS$, i.e. $\exists p, p'$ deux polynômes tels que $\forall I \in D_\pi$

- $|Codage_S(I)| \leq p(\ |Codage_{CS}(I)|\ )$ et
- $|Codage_{CS}(I)| \leq p'(\ |Codage_S(I)|\ )$

où $Codage_S(I)$ et $Codage_{CS}(I)$ désignent les codages de l'instance $I$ par $S$ et $CS$.

Pour le codage des entiers :

- Tout codage des entiers en base $k$ (pour $k \geq 2$) est raisonnable
  car le rapport avec CS est constant car égal à $ln(k)/ln(2)$

# Le codage standard CS et les autres codages raisonnables

Le codage standard CS sert de référence pour les codages raisonnables
(sous la réserve qu'ils offrent aussi la "décodabilité")

## Définition

Un système de codage $S$ pour un problème de décision $\pi$ est dit **raisonnable** si pour toute instance $I$ de $\pi$, la longueur de $I$ dans $S$ est **polynomialement équivalente** à la longueur de $I$ dans le codage standard $CS$, i.e. $\exists p, p'$ deux polynômes tels que $\forall I \in D_\pi$

- $|Codage_S(I)| \leq p(\ |Codage_{CS}(I)|\ )$ et
- $|Codage_{CS}(I)| \leq p'(\ |Codage_S(I)|\ )$

où $Codage_S(I)$ et $Codage_{CS}(I)$ désignent les codages de l'instance $I$ par $S$ et $CS$.

Pour le codage des entiers :

- Tout codage des entiers en base $k$ (pour $k \geq 2$) est raisonnable
car le rapport avec CS est constant car égal à $ln(k)/ln(2)$
- Le codage unaire des entiers n'est donc pas raisonnable :
car le rapport existant entre ce codage et CS est exponentiel
(ces deux codages ne sont donc pas polynomialement équivalents)

# Plan

# Problèmes de décision : alphabets, mots et langages

## Définition

**Langages** (rappels sur la notion de langage formel vue en L2...)

- L'ensemble de tous les mots que l'on peut construire sur un alphabet $\Sigma$ est noté $\Sigma^*$.
- Un langage sur un alphabet $\Sigma$ est un ensemble de mots construits sur $\Sigma$.
- Tout langage défini sur $\Sigma$ est donc un sous-ensemble de $\Sigma^*$.
- L'ensemble de tous les langages que l'on peut définir sur $\Sigma^*$ est l'ensemble des parties de $\Sigma^*$, noté $\mathcal{P}(\Sigma^*)$.

# Problèmes de décision : alphabets, mots et langages

## Définition

**Langages** (rappels sur la notion de langage formel vue en L2...)

- L'ensemble de tous les mots que l'on peut construire sur un alphabet $\Sigma$ est noté $\Sigma^*$.
- Un langage sur un alphabet $\Sigma$ est un ensemble de mots construits sur $\Sigma$.
- Tout langage défini sur $\Sigma$ est donc un sous-ensemble de $\Sigma^*$.
- L'ensemble de tous les langages que l'on peut définir sur $\Sigma^*$ est l'ensemble des parties de $\Sigma^*$, noté $\mathcal{P}(\Sigma^*)$.

- Alphabet : prenons par exemple $\Sigma = \{0, 1, -, [, ], (, ), ;\}$

# Problèmes de décision : alphabets, mots et langages

## Définition

**Langages** (rappels sur la notion de langage formel vue en L2...)

- L'ensemble de tous les mots que l'on peut construire sur un alphabet $\Sigma$ est noté $\Sigma^*$.
- Un langage sur un alphabet $\Sigma$ est un ensemble de mots construits sur $\Sigma$.
- Tout langage défini sur $\Sigma$ est donc un sous-ensemble de $\Sigma^*$.
- L'ensemble de tous les langages que l'on peut définir sur $\Sigma^*$ est l'ensemble des parties de $\Sigma^*$, noté $\mathcal{P}(\Sigma^*)$.

- Alphabet : prenons par exemple $\Sigma = \{0, 1, -, [, ], (, ), ;\}$
- Mots : prenons le codage d'un graphe avec le codage standard

  Le graphe $G_2$ se code par un mot $m \in \Sigma^*$ tel que :

  $m =$ ( ( [1];[10];[11];[100 ] ) ; ( ([1];[11]); ([10];[11]); ([10];[100]); ([11];[100]) ) )

# Problèmes de décision : alphabets, mots et langages

## Définition

**Langages** (rappels sur la notion de langage formel vue en L2...)

- L'ensemble de tous les mots que l'on peut construire sur un alphabet $\Sigma$ est noté $\Sigma^*$.
- Un langage sur un alphabet $\Sigma$ est un ensemble de mots construits sur $\Sigma$.
- Tout langage défini sur $\Sigma$ est donc un sous-ensemble de $\Sigma^*$.
- L'ensemble de tous les langages que l'on peut définir sur $\Sigma^*$ est l'ensemble des parties de $\Sigma^*$, noté $\mathcal{P}(\Sigma^*)$.

- Alphabet : prenons par exemple $\Sigma = \{0, 1, -, [, ], (, ), ;\}$
- Mots : prenons le codage d'un graphe avec le codage standard

  Le graphe $G_2$ se code par un mot $m \in \Sigma^*$ tel que :

  $m =$ ( ( [1];[10];[11];[100 ] ) ; ( ([1];[11]); ([10];[11]); ([10];[100]); ([11];[100]) ) )
- **Langages : quels liens avec les problèmes de décision ?**

# Liens entre problèmes de décision et langages

## Sur un exemple trivial avec un codage simplifié

- Le test de parité, un problème de décision tres simple :

  **PARITÉ**
  **Donnée**: Un entier $n \in \mathbb{N}^*$.
  **Question** : Est-ce que $n$ est un nombre pair ?

# Liens entre problèmes de décision et langages

## Sur un exemple trivial avec un codage simplifié

- Le test de parité, un problème de décision tres simple :

  **PARITÉ**
  **Donnée**: Un entier $n \in \mathbb{N}^*$.
  **Question** : Est-ce que $n$ est un nombre pair ?

- Codage binaire sur $\Sigma = \{0, 1\}$, poids forts à gauche et sans bit redondant

# Liens entre problèmes de décision et langages

## Sur un exemple trivial avec un codage simplifié

- Le test de parité, un problème de décision tres simple :

  **PARITÉ**
  **Donnée**: Un entier $n \in \mathbb{N}^*$.
  **Question** : Est-ce que $n$ est un nombre pair ?

- Codage binaire sur $\Sigma = \{0, 1\}$, poids forts à gauche et sans bit redondant
  - le mot $m = 1100$ code l'instance $n = 12$ : instance positive

# Liens entre problèmes de décision et langages

## Sur un exemple trivial avec un codage simplifié

- Le test de parité, un problème de décision tres simple :

  **PARITÉ**
  **Donnée**: Un entier $n \in \mathbb{N}^*$.
  **Question** : Est-ce que $n$ est un nombre pair ?

- Codage binaire sur $\Sigma = \{0, 1\}$, poids forts à gauche et sans bit redondant
  - le mot $m = 1100$ code l'instance $n = 12$ : instance positive
  - le mot $m' = 1101$ code l'instance $n' = 13$ : instance négative

# Liens entre problèmes de décision et langages

## Sur un exemple trivial avec un codage simplifié

- Le test de parité, un problème de décision tres simple :

  **PARITÉ**
  **Donnée**: Un entier $n \in \mathbb{N}^*$.
  **Question** : Est-ce que $n$ est un nombre pair ?

- Codage binaire sur $\Sigma = \{0, 1\}$, poids forts à gauche et sans bit redondant
  - le mot $m = 1100$ code l'instance $n = 12$ : instance positive
  - le mot $m' = 1101$ code l'instance $n' = 13$ : instance négative
  - le mot $m" = 01101$ qui appartient bien à $\Sigma^*$ ne code pas d'instance car le premier symbole du mot est 0 qui est un bit redondant

# Liens entre problèmes de décision et langages

## Sur un exemple trivial avec un codage simplifié

- Le test de parité, un problème de décision tres simple :

  **PARITÉ**
  **Donnée**: Un entier $n \in \mathbb{N}^*$.
  **Question** : Est-ce que $n$ est un nombre pair ?

- Codage binaire sur $\Sigma = \{0, 1\}$, poids forts à gauche et sans bit redondant
  - le mot $m = 1100$ code l'instance $n = 12$ : instance positive
  - le mot $m' = 1101$ code l'instance $n' = 13$ : instance négative
  - le mot $m" = 01101$ qui appartient bien à $\Sigma^*$ ne code pas d'instance car le premier symbole du mot est 0 qui est un bit redondant

## En généralisant à tout problème de décision $\pi$

Soient $\pi$, $S$ (système de codage) et $\Sigma$ : $\Sigma^*$ se partitionne en trois classes de mots :

# Liens entre problèmes de décision et langages

## Sur un exemple trivial avec un codage simplifié

- Le test de parité, un problème de décision tres simple :

  **PARITÉ**
  **Donnée**: Un entier $n \in \mathbb{N}^*$.
  **Question** : Est-ce que $n$ est un nombre pair ?

- Codage binaire sur $\Sigma = \{0, 1\}$, poids forts à gauche et sans bit redondant
  - le mot $m = 1100$ code l'instance $n = 12$ : instance positive
  - le mot $m' = 1101$ code l'instance $n' = 13$ : instance négative
  - le mot $m" = 01101$ qui appartient bien à $\Sigma^*$ ne code pas d'instance car le premier symbole du mot est 0 qui est un bit redondant

## En généralisant à tout problème de décision $\pi$

Soient $\pi$, $S$ (système de codage) et $\Sigma$ : $\Sigma^*$ se partitionne en trois classes de mots :

- les mots de $\Sigma^*$ codant par $S$ les instances positives de $\pi$ (c'est $V_\pi$)

# Liens entre problèmes de décision et langages

## Sur un exemple trivial avec un codage simplifié

- Le test de parité, un problème de décision tres simple :

  **PARITÉ**
  **Donnée**: Un entier $n \in \mathbb{N}^*$.
  **Question** : Est-ce que $n$ est un nombre pair ?

- Codage binaire sur $\Sigma = \{0, 1\}$, poids forts à gauche et sans bit redondant
  - le mot $m = 1100$ code l'instance $n = 12$ : instance positive
  - le mot $m' = 1101$ code l'instance $n' = 13$ : instance négative
  - le mot $m" = 01101$ qui appartient bien à $\Sigma^*$ ne code pas d'instance car le premier symbole du mot est 0 qui est un bit redondant

## En généralisant à tout problème de décision $\pi$

Soient $\pi$, $S$ (système de codage) et $\Sigma$ : $\Sigma^*$ se partitionne en trois classes de mots :

- les mots de $\Sigma^*$ codant par $S$ les instances positives de $\pi$ (c'est $V_\pi$)
- les mots de $\Sigma^*$ codant par $S$ les instances négatives de $\pi$ (c'est $D_\pi \backslash V_\pi$)

# Liens entre problèmes de décision et langages

## Sur un exemple trivial avec un codage simplifié

- Le test de parité, un problème de décision tres simple :

  **PARITÉ**
  **Donnée**: Un entier $n \in \mathbb{N}^*$.
  **Question** : Est-ce que $n$ est un nombre pair ?

- Codage binaire sur $\Sigma = \{0, 1\}$, poids forts à gauche et sans bit redondant
  - le mot $m = 1100$ code l'instance $n = 12$ : instance positive
  - le mot $m' = 1101$ code l'instance $n' = 13$ : instance négative
  - le mot $m" = 01101$ qui appartient bien à $\Sigma^*$ ne code pas d'instance car le premier symbole du mot est 0 qui est un bit redondant

## En généralisant à tout problème de décision $\pi$

Soient $\pi$, $S$ (système de codage) et $\Sigma$ : $\Sigma^*$ se partitionne en trois classes de mots :

- les mots de $\Sigma^*$ codant par $S$ les instances positives de $\pi$ (c'est $V_\pi$)
- les mots de $\Sigma^*$ codant par $S$ les instances négatives de $\pi$ (c'est $D_\pi \setminus V_\pi$)
- les mots de $\Sigma^*$ ne codant par $S$ aucune instance de $\pi$ (donc sans lien avec $D_\pi$)

# Liens entre problèmes de décision et langages

À tout problème de décision $\pi$ codé via un système $S$ (avec $\Sigma$)

**on associe le langage $L(\pi, S)$ des instances positives de $\pi$ :**

$$L(\pi, S) = \{\ m \in \Sigma^* : m \text{ code par le système } S \text{ une instance } I \in V_\pi\ \}$$

# Liens entre problèmes de décision et langages

À tout problème de décision $\pi$ codé via un système $S$ (avec $\Sigma$)

**on associe le langage $L(\pi, S)$ des instances positives de $\pi$ :**

$$L(\pi, S) = \{\ m \in \Sigma^* : m \text{ code par le système } S \text{ une instance } I \in V_\pi\ \}$$

Cette approche à base de langages recèle au moins trois avantages ici :

1. **La taille d'une instance est connue à l'unité près :**
   - c'est la longueur d'un mot de $\Sigma^*$ la codant
   - crucial pour l'analyse de la complexité d'un algorithme de résolution

# Liens entre problèmes de décision et langages

À tout problème de décision $\pi$ codé via un système $S$ (avec $\Sigma$)

**on associe le langage $L(\pi, S)$ des instances positives de $\pi$ :**

$$L(\pi, S) = \{\ m \in \Sigma^* : m \text{ code par le système } S \text{ une instance } I \in V_\pi\ \}$$

Cette approche à base de langages recèle au moins trois avantages ici :

1. **La taille d'une instance est connue à l'unité près :**
   - c'est la longueur d'un mot de $\Sigma^*$ la codant
   - crucial pour l'analyse de la complexité d'un algorithme de résolution
2. **Résoudre un problème de décision $\pi$ :**
   - c'est résoudre le problème de reconnaissance du langage $L(\pi, S)$
   - c'est un problème bien connu
   - cela se traite avec des automates, des machines de Turing, etc.

# Liens entre problèmes de décision et langages

À tout problème de décision $\pi$ codé via un système $S$ (avec $\Sigma$)

**on associe le langage $L(\pi, S)$ des instances positives de $\pi$** :

$$L(\pi, S) = \{ \ m \in \Sigma^* : m \text{ code par le système } S \text{ une instance } I \in V_\pi \ \}$$

Cette approche à base de langages recèle au moins trois avantages ici :

1. **La taille d'une instance est connue à l'unité près :**
   - c'est la longueur d'un mot de $\Sigma^*$ la codant
   - crucial pour l'analyse de la complexité d'un algorithme de résolution
2. **Résoudre un problème de décision $\pi$ :**
   - c'est résoudre le problème de reconnaissance du langage $L(\pi, S)$
   - c'est un problème bien connu
   - cela se traite avec des automates, des machines de Turing, etc.
3. **Pour l'analyse de la complexité d'un algorithme de résolution :**
   - temps $=$ nombre de transitions du dispositif utilisé
   - $\Rightarrow$ plus aucune imprécision dans l'évaluation ($\neq$ modèle du 1er cours)

# Comme illustration : le problème PARITÉ

*Ce problème n'a d'intérêt que pédagogique...*

- Rappel du probléme PARITÉ

  **PARITÉ**
  **Donnée**: Un entier $n \in \mathbb{N}^*$.
  **Question** : Est-ce que $n$ est un nombre pair ?

## Comme illustration : le problème PARITÉ

*Ce problème n'a d'intérêt que pédagogique...*

- Rappel du probléme PARITÉ

  **PARITÉ**
  **Donnée**: Un entier $n \in \mathbb{N}^*$.
  **Question** : Est-ce que $n$ est un nombre pair ?

- Codage binaire ($\Sigma = \{0, 1\}$), poids forts à gauche sans bit redondant :

  la taille d'une instances $n \in \mathbb{N}^*$ est exactement égale à $\lfloor log_2(n) \rfloor + 1$

# Comme illustration : le problème PARITÉ

*Ce problème n'a d'intérêt que pédagogique...*

- Rappel du probléme PARITÉ

  **PARITÉ**
  **Donnée**: Un entier $n \in \mathbb{N}^*$.
  **Question** : Est-ce que $n$ est un nombre pair ?

- Codage binaire ($\Sigma = \{0, 1\}$), poids forts à gauche sans bit redondant :

  la taille d'une instances $n \in \mathbb{N}^*$ est exactement égale à $\lfloor log_2(n) \rfloor + 1$

- Résolution de PARITÉ : reconnaissance du langage $L(PARIT\acute{E}, S)$

  $L(PARIT\acute{E}, S) = \{m \in \Sigma^* : m$ code par le système $S$ une instance $I \in V_{PARITE}\}$

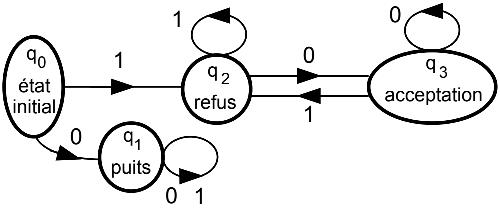
  et comme avec le système de codage, les entiers pairs se terminent par 0 :

  $$L(PARIT\acute{E}, S) = \{m \in \Sigma^* : m \text{ débute par 1 et termine par 0 }\}$$

# Comme illustration : le probléme PARITÉ

- **Résolution de PARITÉ** par un Automate d'États Fini Déterministe :

  en reconnaissant $L(PARIT\acute{E}, S) = \{m \in \Sigma^* : m$ débute par 1 et termine par 0 $\}$



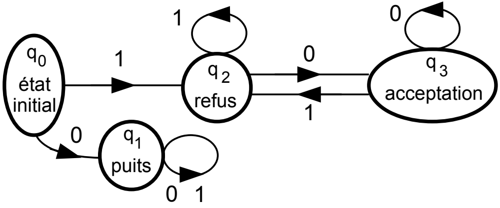

0 au début (puits donc refus) ou 1 pour commencer

0 à la fin (acceptation) ou 1 (refus)

# Comme illustration : le probléme PARITÉ

- **Résolution de PARITÉ** par un Automate d'États Fini Déterministe :

  en reconnaissant $L(PARIT\acute{E}, S) = \{m \in \Sigma^* : m$ débute par 1 et termine par 0 $\}$




0 au début (puits donc refus) ou 1 pour commencer

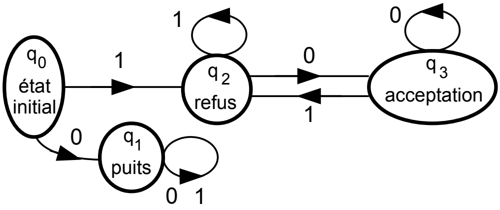
0 à la fin (acceptation) ou 1 (refus)

- **Analyse de la complexité :**
  - nombre maximum de transitions (cf. pire des cas) : taille de l'instance

# Comme illustration : le probléme PARITÉ

- **Résolution de PARITÉ** par un Automate d'États Fini Déterministe :

  en reconnaissant $L(PARIT\acute{E}, S) = \{m \in \Sigma^* : m$ débute par 1 et termine par 0 $\}$




0 au début (puits donc refus) ou 1 pour commencer

0 à la fin (acceptation) ou 1 (refus)

- **Analyse de la complexité :**
  - nombre maximum de transitions (cf. pire des cas) : taille de l'instance
  - c'est exactement $\lfloor log_2(n) \rfloor + 1$ soit en $\Theta(log(n))$ : complexité linéaire en la taille de l'entrée (taille du codage de $n$ en binaire)

# En conclusion

**Ce qui a été vu :**

- **Proposition d'un "système" de codage :**
  - raisonnable
  - permettant de coder tout type de données : des nombres, des noms, des graphes, des bases de données, etc.
  - dont la taille des objets codés sera précisément connue :
    - toute donnée sera représentée par un mot au sens des langages formels
    - la taille exacte d'une donnée en entrée : longueur du mot la codant

# En conclusion

## Ce qui a été vu :

- **Proposition d'un "système" de codage :**
  - raisonnable
  - permettant de coder tout type de données : des nombres, des noms, des graphes, des bases de données, etc.
  - dont la taille des objets codés sera précisément connue :
    - toute donnée sera représentée par un mot au sens des langages formels
    - la taille exacte d'une donnée en entrée : longueur du mot la codant
- **Résolution d'un problème de décision :**
  - équivaut à un problème de reconnaissance de langage (formel)
  - question à traiter : est-ce qu'un mot appartient à un langage ?

# En conclusion

## Ce qui a été vu :

- **Proposition d'un "système" de codage :**
  - raisonnable
  - permettant de coder tout type de données : des nombres, des noms, des graphes, des bases de données, etc.
  - dont la taille des objets codés sera précisément connue :
    - toute donnée sera représentée par un mot au sens des langages formels
    - la taille exacte d'une donnée en entrée : longueur du mot la codant
- **Résolution d'un problème de décision :**
  - équivaut à un problème de reconnaissance de langage (formel)
  - question à traiter : est-ce qu'un mot appartient à un langage ?
- **Temps de calcul connu sans ambiguïté :**
  nombre de transitions nécessaires par la "machine" utilisée : automate fini, machine de Turing, etc.

# En conclusion

## Ce qui a été vu :

- **Proposition d'un "système" de codage :**
  - raisonnable
  - permettant de coder tout type de données : des nombres, des noms, des graphes, des bases de données, etc.
  - dont la taille des objets codés sera précisément connue :
    - toute donnée sera représentée par un mot au sens des langages formels
    - la taille exacte d'une donnée en entrée : longueur du mot la codant
- **Résolution d'un problème de décision :**
  - équivaut à un problème de reconnaissance de langage (formel)
  - question à traiter : est-ce qu'un mot appartient à un langage ?
- **Temps de calcul connu sans ambiguïté :**
  nombre de transitions nécessaires par la "machine" utilisée : automate fini, machine de Turing, etc.

## Ce que nous pouvons et allons faire maintenant :

# Étudier avec précision la complexité des problèmes