Dernier cours : modèles de calcul

- programmes WHILE
- machines de Turing

- Les programmes WHILE manipulent des variables entières (non-négatives), en utilisant des opérations arithmétiques et des boucles (WHILE).

- Les machines de Turing utilisent une bande infinie, dont le contenu peut être lu et réécrit, en utilisant un contrôle fini (les états/transitions).

(voir poly)

---

Les programmes WHILE et les machines de Turing ont le même pouvoir de calcul (ils calculent les mêmes fonctions)

On verra maintenant les notions suivantes :

- problème décidable / indécidable
- réductions
- problème de l'arrêt (des programmes WHILE ou des machines de Turing)

Rq les deux modèles, programmes WHILE et MT, ont le problème de non-terminaison potentielle : il y a des programmes ou machines avec des calculs infinis.

## Problème de l'arrêt (terminaison) :

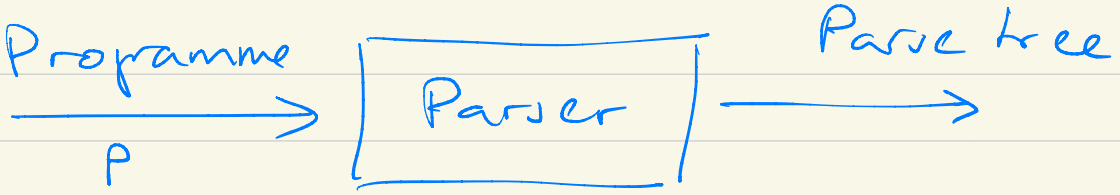
Entrée : programme WHILE P (ou machine de Turing), et entrée $n$ de P

Question est-ce que le calcul de P sur $n$ termine ?

On verra que ce problème est indécidable, mais ça veut dire quoi ?

Exemple (compilateur)

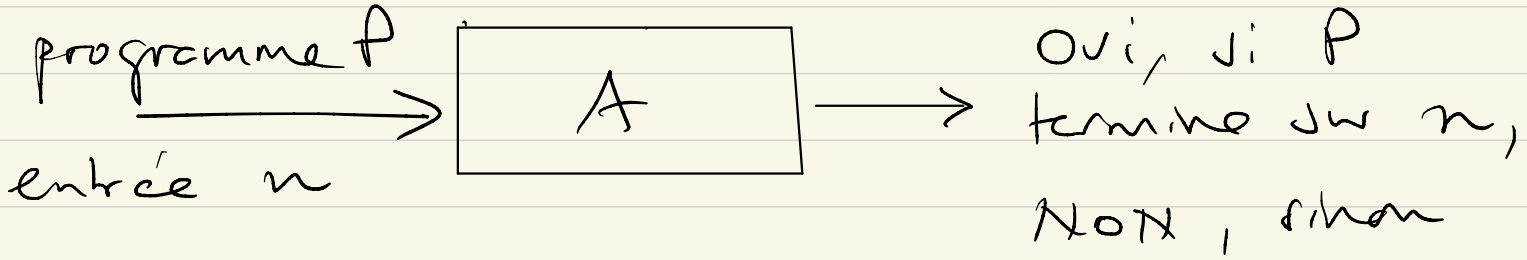
- analyse lexicale
- parsing : est-ce que un programme P est conforme à la grammaire du langage haut-niveau ?




Le Parser (eg. YACC) est lui-même un programme, qui analyse un programme P quelconque (pour un langage de programmation fixé).

Notre question ("décidabilité de l'arrêt") est la suivante :

est-ce qu'il existe un programme A qui s'arrête toujours tel que :



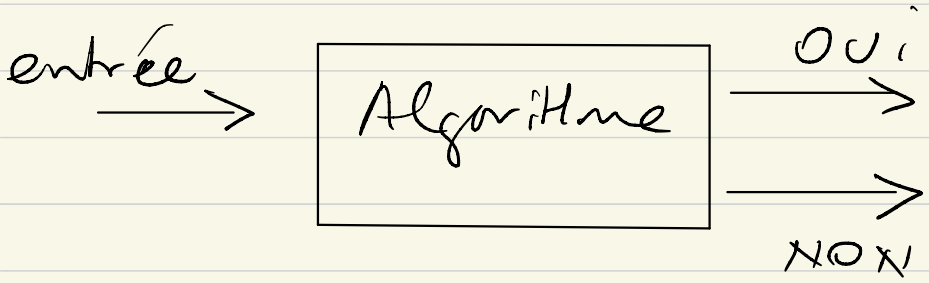
C'est évident qu'on peut répondre "OUI" sur $(P, n)$ si $P$ termine sur $n$, juste en lançant $P$ sur $n$.

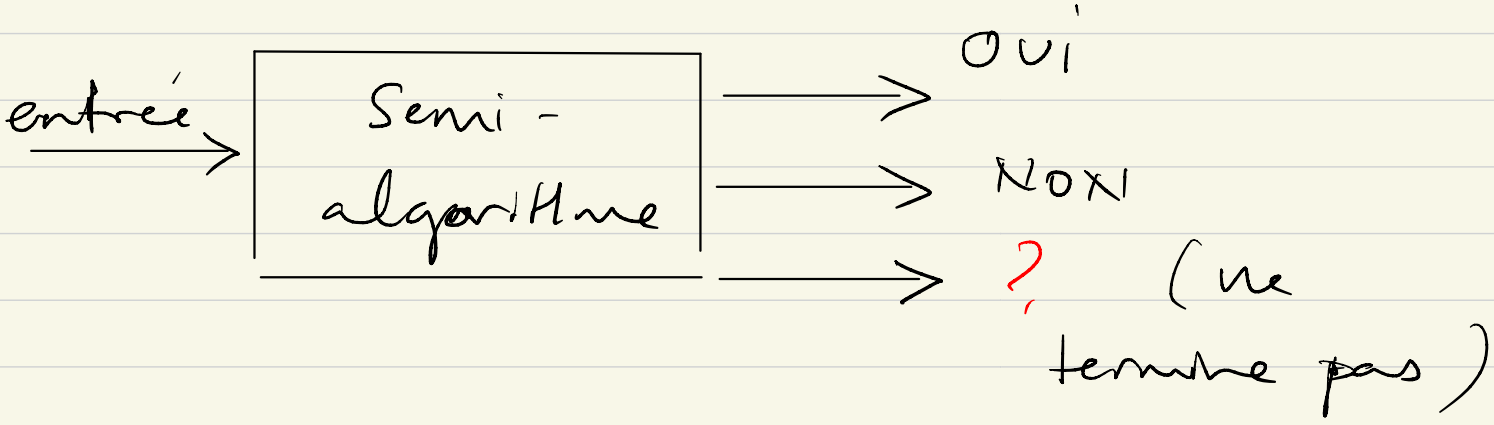
Mais comment peut-on savoir si $P$ ne termine pas sur $n$ ?

Ben, on ne peut pas le savoir !

On appelle un programme WHILE qui s'arrête sur toutes ses entrées, un algorithme.

Tandis que un programme WHILE arbitraire (qui ne s'arrête pas forcément sur toutes ses entrées) est appelé semi-algorithme.

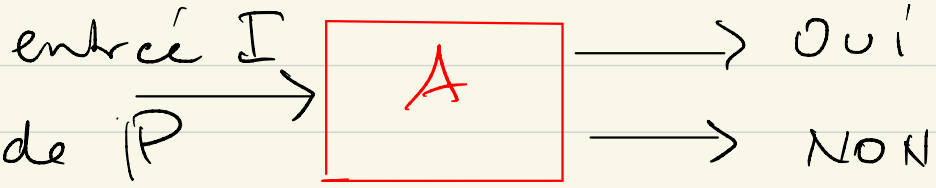



---

On appelle un problème $\mathbb{P}$ décidable s'il existe un algorithme $A$ qui résout $\mathbb{P}$.

---




I : entrée de $\mathbb{P}$ (instance)

$A$ doit répondre "OUI" si $I$ est positive, et "NON" sinon.

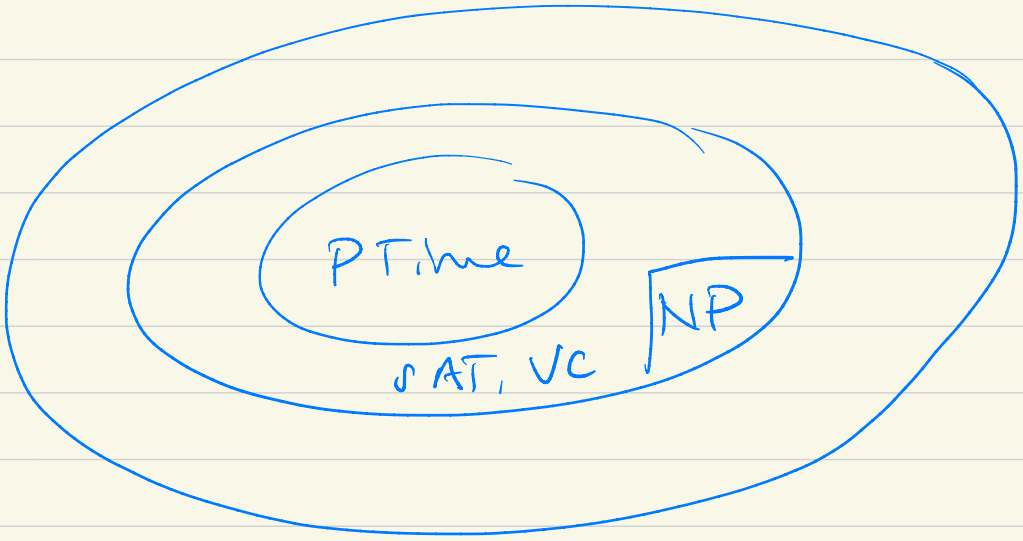
Le problème de l'arrêt (de programmes WHILE) n'est pas décidable.

Pas décidable ≠ indécidable

Un problème $\mathbb{P}$ est indécidable, s'il n'existe pas d'algorithme qui le résout.

$\mathbb{P}$ = SAT  décidable

- HALT (pb. de l'arrêt)




complexité $2^{2^{2^{n}}}$

On ne peut pas décider si un programme P quelconque

- s'arrête sur une entrée particulière
- s'arrête sur au moins une entrée
- s'arrête sur toutes ses entrées
- utilise une variable précise

Bien-sûr, on peut y répondre pour des programmes particuliers (while(true)skip) tout comme pour ex. SAT :

si une formule est en 2-CNF, alors on peut savoir en PTime si elle est satisfiable

Pour montrer que HALT (problème de l'arrêt) est indécidable on utilise des réductions.

---

$A, B$ deux problèmes

$X_A, X_B$ : instances de $A, B$ resp.

$f : X_A \to X_B$ est réduction de $A$ vers $B$ si

- $f$ est calculable, et
- $x$ instance positive de $A \iff f(x)$ ——— de $B$

---

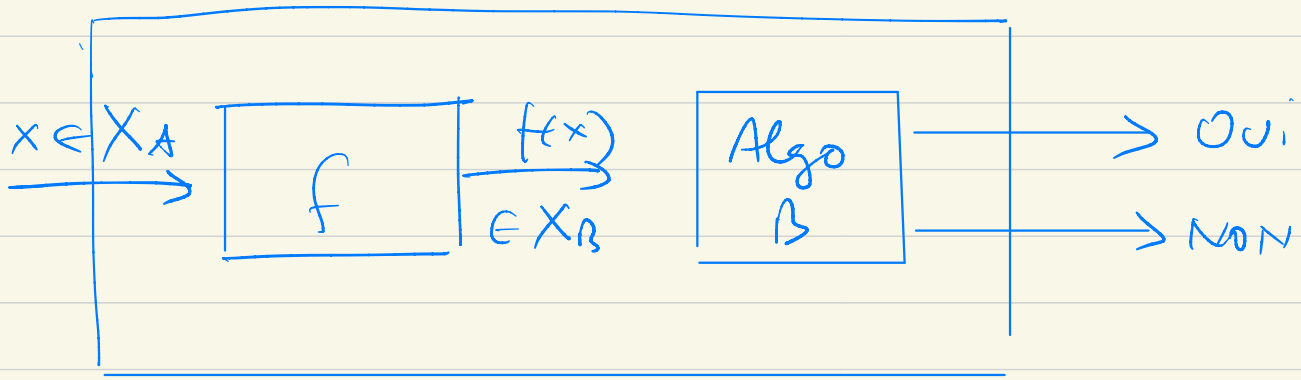
$A \leq B$ s'il ex. une réduction de $A$ vers $B$

① $A \leq B$ , $B$ décidable $\Rightarrow$

$A$ décidable

② $A \leq B$ , $A$ indécidable $\Rightarrow$

$B$ indécidable




Algo pour $A$

décidable —analogie→ PTime / facile

indécidable —→ NP / difficile

On montre que HALT est indécidable en passant par deux autres problèmes, DIAG et UNIV.

UNIV :

Entrée programme P, entrée n

Question est-ce que P termine sur n et retourne 1 ?

Il n'existe pas d'algorithme pour le pb UNIV, ça veut dire que UNIV est indécidable.

UNIV $\leq$ HALT

Tout cela montre que HALT est indécidable.

UNIV

Entrée : P, n

Q est-ce que P termine sur n avec réponse 1 ?

HALT

Entrée : P, n

Q : est-ce que termine sur n ?

UNIV $\leq$ HALT

$P, n \longrightarrow P', n'$

P termine sur n avec résultat 1 $\Leftrightarrow$ P' termine sur n'

$n' = n$

P' = P ; IF (res $\neq$ 1) DO WHILE (true) DO skip OD

Ei