Théorie du calcul - Théorème de Cook-Levin

SAT est NP-complet

Théorème (Cook-Levin, 1971)

SAT est NP-complet.

SAT est NP-complet

Théorème (Cook-Levin, 1971)

SAT est NP-complet.

- 1) SAT est dans NP: sur instance ϕ de SAT, une assignation A satisfaisant ϕ peut servir de certificat. Il est facile de vérifier que A satisfait ϕ en temps polynomial.

SAT est NP-complet

- 2) SAT est NP-difficile.
  Aucun problème duquel réduire. Il faut montrer que $\forall L \in \text{NP}, L \prec_P SAT$.

SAT est NP-complet

- 2) SAT est NP-difficile.
  Aucun problème duquel réduire. Il faut montrer que
  $\forall L \in \text{NP}, L \prec_P SAT$.
- Soit $L \in \text{NP}$. Alors $\exists$ MT non-déterministe M dont le langage accepté est L, en temps $O(n^k)$, avec $k \in \mathbb{N}$.

SAT est NP-complet

- 2) SAT est NP-difficile.
  Aucun problème duquel réduire. Il faut montrer que $\forall L \in \text{NP}, L \prec_P SAT$.
- Soit $L \in \text{NP}$. Alors $\exists$ MT non-déterministe M dont le langage accepté est L, en temps $O(n^k)$, avec $k \in \mathbb{N}$.
- Pour simplifier, on pose ce temps comme n^k.

Configurations

Définition

Configuration de M (sur entrée w de taille n):
chaîne de longueur $O(n^k)$ qui représente l'état de la mémoire, la position de la tête et l'état courant.

$$\# \quad s_1 \quad s_2 \quad s_3 \quad \ldots \quad s_{i-1} \quad q \quad s_i \quad \ldots \quad s_{n^k} \quad \#$$

$\#$: début et fin.
s_i: symboles de la mémoire.
q: état courant, placé à la position de la MT M (symbole vu: s_i).

Tableaux

Définition

Tableau de configuration: séquence de configurations de M.
Rangée 1: était initial sur entrée w.
Rangée $i \to i+1$ est *possible* selon les spécifications de M.

$$
\begin{array}{cccccccccc}
\# & q_0 & w_1 & w_2 & \dots & w_n & \sqcup & \sqcup & \dots & \sqcup & \# \\
\# & w_1 & q_1 & w_2 & \dots & w_n & \sqcup & \sqcup & \dots & \sqcup & \# \\
\# & w_1 & x & q_2 & \dots & w_n & \sqcup & \sqcup & \dots & \sqcup & \# \\
 & \dots & & & & & & & & & \\
\# & s_1 & s_2 & s_3 & \dots & s_{i-1} & q & s_i & \dots & s_{n^k} & \# \\
\# & s_1 & s_2 & s_3 & \dots & s_{i-1} & s_i' & q' & \dots & s_{n^k} & \# \\
 & \dots & & & & & & & & & \\
\# & s_1' & s_2' & s_3' & \dots & s_{i-1}' & s_i' & s_{i+1}' & \dots & s_{n^k}' & \#
\end{array}
$$

Tableaux

Un mot w est accepté $\Leftrightarrow$ il existe un tableau de M avec $\leq n^k$ rangées, dans lequel une rangée contient un état acceptant.

$\#$	q_0	w_1	w_2	$\ldots$	w_n	$\sqcup$	$\sqcup$	$\ldots$	$\sqcup$	$\#$
$\#$	w_1	q_1	w_2	$\ldots$	w_n	$\sqcup$	$\sqcup$	$\ldots$	$\sqcup$	$\#$
$\#$	w_1	x	q_2	$\ldots$	w_n	$\sqcup$	$\sqcup$	$\ldots$	$\sqcup$	$\#$
	$\ldots$									
$\#$	s_1	s_2	s_3	$\ldots$	s_{i-1}	q	s_i	$\ldots$	s_{n^k}	$\#$
$\#$	s_1	s_2	s_3	$\ldots$	s_{i-1}	s'_i	q'	$\ldots$	s_{n^k}	$\#$
	$\ldots$									
$\#$	s'_1	s'_2	s'_3	$\ldots$	s'_{i-1}	s'_i	s'_{i+1}	$\ldots$	s'_{n^k}	$\#$

Pour savoir si $w \in L$, il "suffit" de regarder tous les tableaux possibles de dimension $n^k \times n^k$ sur entrée w.

Réduction

Réduction d'un langage arbitraire $L \in$ NP vers SAT.

- Sur entrée w instance de L, générer une instance transformée $f(w) = \phi$ de SAT en temps polynomial, telle que:

Réduction

Réduction d'un langage arbitraire $L \in$ NP vers SAT.

- Sur entrée w instance de L, générer une instance transformée $f(w) = \phi$ de SAT en temps polynomial, telle que:
- il existe un tableau de M sur entrée w contenant une rangée acceptante

  $\Leftrightarrow$

  ϕ est satisfaisable.

Réduction

Réduction d'un langage arbitraire $L \in$ NP vers SAT.

- Sur entrée w instance de L, générer une instance transformée $f(w) = \phi$ de SAT en temps polynomial, telle que:
- il existe un tableau de M sur entrée w contenant une rangée acceptante

  $\Leftrightarrow$

  ϕ est satisfaisable.
- La formule ϕ "encode" l'existence d'un tableau acceptant.

Réduction

Il existe un tableau de M sur entrée w contenant une rangée acceptante

$\Leftrightarrow$

ϕ est satisfaisable.

ϕ doit encoder 4 conditions d'un tableau valide de M:

1. le tableau démarre dans l'état initial sur entrée w;
2. chaque case du tableau a une valeur bien définie;
3. il y a une rangée acceptante dans le tableau;
4. chaque rangée i peut mener à la rangée $i+1$ selon M.

Réduction

Il existe un tableau de M sur entrée w contenant une rangée acceptante

$\Leftrightarrow$

ϕ est satisfaisable.

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$, où:

1. ϕ_{init}: le tableau démarre dans l'état initial sur entrée w;
2. ϕ_{case}: chaque case du tableau a une valeur bien définie;
3. ϕ_{accept}: il y a une rangée acceptante dans le tableau;
4. ϕ_{move}: chaque rangée i peut mener à la rangée $i+1$ selon M.

Réduction

Variables de ϕ: pour $i, j \in O(n^k)$ et $s \in \Sigma$,

$$x_{i,j,s}$$

est une variable de ϕ.

Réduction

Variables de ϕ: pour $i, j \in O(n^k)$ et $s \in \Sigma$,

$$x_{i,j,s}$$

est une variable de ϕ.

Interprétation:
$x_{i,j,s} = V$ veut dire "on met s à la case i, j du tableau".
$x_{i,j,s} = F$ veut dire "on ne met **pas** s à la case i, j du tableau".

Réduction

Variables de ϕ: pour $i, j \in O(n^k)$ et $s \in \Sigma$,

$$x_{i,j,s}$$

est une variable de ϕ.

Interprétation:
$x_{i,j,s} = V$ veut dire "on met s à la case i, j du tableau".
$x_{i,j,s} = F$ veut dire "on ne met **pas** s à la case i, j du tableau".

Demande $O(n^k \cdot n^k \cdot |\Sigma|) = O(n^{2k})$ variables.

Réduction

$$
\begin{array}{cccccccccc}
\# & q_0 & w_1 & w_2 & \dots & w_n & \sqcup & \sqcup & \dots & \sqcup & \# \\
\# & w_1 & q_1 & w_2 & \dots & w_n & \sqcup & \sqcup & \dots & \sqcup & \# \\
\# & w_1 & x & q_2 & \dots & w_n & \sqcup & \sqcup & \dots & \sqcup & \# \\
 & \dots & & & & & & & & & \\
\# & s_1 & s_2 & s_3 & \dots & s_{i-1} & q & s_i & \dots & s_{n^k} & \# \\
\# & s_1 & s_2 & s_3 & \dots & s_{i-1} & s_i' & q' & \dots & s_{n^k} & \# \\
 & \dots & & & & & & & & & \\
\# & s_1' & s_2' & s_3' & \dots & s_{i-1}' & s_i' & s_{i+1}' & \dots & s_{n^k}' & \#
\end{array}
$$

$$
x_{1,2,q_0} = V \qquad x_{1,2,w_1} = F \qquad x_{1,2,w_2} = F \qquad \text{etc.}
$$

Réduction

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$.

ϕ_{init}: le tableau démarre dans l'état initial sur entrée w.

$$\begin{aligned}\phi_{init} = {} & x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \\ & \ldots \wedge \\ & x_{1,n+2,w_n} \wedge x_{1,n+3,\sqcup} \wedge \ldots \wedge x_{1,n^k+3,\#}\end{aligned}$$

Réduction

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$.

ϕ_{case}: chaque case du tableau a une valeur bien définie.

Toute case i, j contient s_1 et rien d'autre, ou s_2 et rien d'autre, etc
Soit $S = \Sigma \cup Q \cup \{\#\}$ les symboles possibles.

Réduction

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$.

ϕ_{case}: chaque case du tableau a une valeur bien définie.

Toute case i, j contient s_1 et rien d'autre, ou s_2 et rien d'autre, etc
Soit $S = \Sigma \cup Q \cup \{\#\}$ les symboles possibles.

$$\phi_{case} = \bigwedge_{i,j} \left(\bigvee_{s \in S} x_{i,j,s} \wedge \neg \left(\bigvee_{t \in S \setminus \{s\}} x_{i,j,t} \right) \right)$$

Réduction

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$.

ϕ_{accept}: le tableau contient une rangée acceptante. Soit A l'ensemble des états acceptants.

$$\phi_{accept} = \bigvee_{i,j} \bigvee_{q \in A} x_{i,j,q}$$

Réduction

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$.

ϕ_{move}: chaque transition de rangée $i \rightarrow i+1$ est possible selon M.

- On aimerait bien faire un gros OU sur toutes les paires de configurations consécutives valides.

Réduction

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$.

ϕ_{move}: chaque transition de rangée $i \rightarrow i+1$ est possible selon M.

- On aimerait bien faire un gros OU sur toutes les paires de configurations consécutives valides.
- Mais il y a un nombre exponentiel de configurations. On ne pourrait pas générer ϕ en temps polynomial!

Réduction

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$.

ϕ_{move}: chaque transition de rangée $i \rightarrow i+1$ est possible selon M.

- On aimerait bien faire un gros OU sur toutes les paires de configurations consécutives valides.
- Mais il y a un nombre exponentiel de configurations. On ne pourrait pas générer ϕ en temps polynomial!
- On regarde seulement les sous-tableaux 2×3 et on vérifie que eux sont tous possibles.

Réduction

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$.

ϕ_{move}: chaque transition de rangée $i \rightarrow i + 1$ est possible selon M.

- On aimerait bien faire un gros OU sur toutes les paires de configurations consécutives valides.
- Mais il y a un nombre exponentiel de configurations. On ne pourrait pas générer ϕ en temps polynomial!
- On regarde seulement les sous-tableaux 2×3 et on vérifie que eux sont tous possibles.
- Un sous-tableau 2×3 est une *fenêtre légale*.

Fenêtres légales

#	q_0	w_1	w_2	$\dots$	w_n	$\sqcup$	$\sqcup$	$\dots$	$\sqcup$	#
#	w_1	q_1	w_2	$\dots$	w_n	$\sqcup$	$\sqcup$	$\dots$	$\sqcup$	#
#	w_1	x	q_2	$\dots$	w_n	$\sqcup$	$\sqcup$	$\dots$	$\sqcup$	#
	$\dots$									
#	s_1	s_2	s_3	$\dots$	s_{i-1}	q	s_i	$\dots$	s_{n^k}	#
#	s_1	s_2	s_3	$\dots$	s_{i-1}	s'_i	q'	$\dots$	s_{n^k}	#
	$\dots$									
#	s'_1	s'_2	s'_3	$\dots$	s'_{i-1}	s'_i	s'_{i+1}	$\dots$	s'_{n^k}	#

Fenêtres légales

#	q_0	w_1	w_2	$\dots$	w_n	$\sqcup$	$\sqcup$	$\dots$	$\sqcup$	#
#	w_1	q_1	w_2	$\dots$	w_n	$\sqcup$	$\sqcup$	$\dots$	$\sqcup$	#
#	w_1	x	q_2	$\dots$	w_n	$\sqcup$	$\sqcup$	$\dots$	$\sqcup$	#
	$\dots$									
#	s_1	s_2	s_3	$\dots$	s_{i-1}	q	s_i	$\dots$	s_{n^k}	#
#	s_1	s_2	s_3	$\dots$	s_{i-1}	s'_i	q'	$\dots$	s_{n^k}	#
	$\dots$									
#	s'_1	s'_2	s'_3	$\dots$	s'_{i-1}	s'_i	s'_{i+1}	$\dots$	s'_{n^k}	#

Fenêtres légales

ex: M : $\delta(q_1, a) = \{(q_1, b, R)\}$ $\delta(q_1, b) = \{(q_2, c, L), (q_2, a, R)\}$

Fenêtres légales :

a	a	b
a	a	b

a	q_1	b
a	a	q_2

a	a	q_1
a	a	b

illégale

a	b	a
a	a	a

a	q_1	b
q_1	a	a

Réduction

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$.

ϕ_{move}: chaque transition de rangée $i \rightarrow i + 1$ est possible selon M.

Lemme

Un tableau est valide si et seulement si sa première rangée est l'état initial correct et chaque fenêtre 2×3 *est légale.*

Réduction

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$.

ϕ_{move}: chaque transition de rangée $i \rightarrow i+1$ est possible selon M.

Lemme

Un tableau est valide si et seulement si sa première rangée est l'état initial correct et chaque fenêtre 2×3 *est légale.*

- Nombre constant de valeurs possibles dans une case. Le nombre de fenêtres possibles est constant: $(|\Sigma| + |Q| + 1)^6$
- Pour chaque fenêtre possible, on vérifie avec M si la fenêtre est légale (temps polynomial).

Réduction

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$.

ϕ_{move}: chaque transition de rangée $i \rightarrow i+1$ est possible selon M.

$\phi_{move} = \bigwedge_{i,j}$ "les six cases autour de i, j forment une fenêtre légale"

Réduction

$\phi = \phi_{init} \wedge \phi_{case} \wedge \phi_{accept} \wedge \phi_{move}$ est satisfaisable.

$\Leftrightarrow$

Il existe un tableau valide de M sur entrée w.

$\Leftrightarrow$

$w \in L$.

f s'exécute en temps polynomial et $w \in L \Leftrightarrow f(w) \in SAT$ et donc SAT est NP-difficile!