Caractérisation de PSPACE par équations différentielles

Yassine Hamoudi
encadré par Olivier Bournez

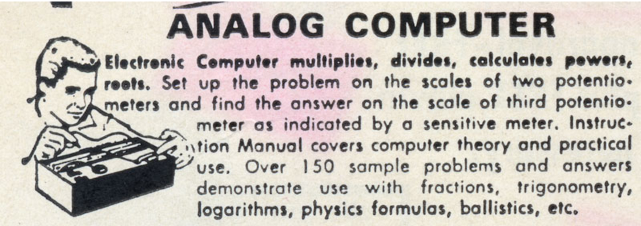
2 septembre 2014

Introduction

Modèles de calcul continus :

- Analyse récursive
- Machines de Blum-Shub-Smale
- GPAC
- ...

$\longrightarrow$ thèse de Church-Turing analogique ?



Plan

Modèle GPAC

GPAC (General Purpose Analog Computer)

Définition

Un langage $\mathcal{L}$ est reconnu par un GPAC si et seulement si il existe un vecteur de polynômes p, une fonction d'encodage ψ tel que pour tout mot w, la solution y au système :

$$\begin{cases} y'(t) = p(y(t)) \\ y(0) = \psi(w) \end{cases}$$

satisfait :

GPAC (General Purpose Analog Computer)

Définition

Un langage $\mathcal{L}$ est reconnu par un GPAC si et seulement si il existe un vecteur de polynômes p, une fonction d'encodage ψ tel que pour tout mot w, la solution y au système :

$$\begin{cases} y'(t) = p(y(t)) \\ y(0) = \psi(w) \end{cases}$$

satisfait :

- y est définie sur $\mathbb{R}$

GPAC (General Purpose Analog Computer)

Définition

Un langage $\mathcal{L}$ est reconnu par un GPAC si et seulement si il existe un vecteur de polynômes p, une fonction d'encodage ψ tel que pour tout mot w, la solution y au système :

$$\begin{cases} y'(t) = p(y(t)) \\ y(0) = \psi(w) \end{cases}$$

satisfait :

- y est définie sur $\mathbb{R}$
- si $w \in \mathcal{L}$ (resp. $w \notin \mathcal{L}$) alors $\exists t > 0$ t.q. $y_1(t) \geqslant 1$ (resp. $y_1(t) \leqslant -1$)

GPAC (General Purpose Analog Computer)

Définition

Un langage $\mathcal{L}$ est reconnu par un GPAC si et seulement si il existe un vecteur de polynômes p, une fonction d'encodage ψ tel que pour tout mot w, la solution y au système :

$$\begin{cases} y'(t) = p(y(t)) \\ y(0) = \psi(w) \end{cases}$$

satisfait :

- y est définie sur $\mathbb{R}$
- si $w \in \mathcal{L}$ (resp. $w \notin \mathcal{L}$) alors $\exists t > 0$ t.q. $y_1(t) \geqslant 1$ (resp. $y_1(t) \leqslant -1$)
- si $y_1(t) \geqslant 1$ (resp. $y_1(t) \leqslant -1$) alors $\forall u \geqslant t$, $y_1(u) \geqslant 1$ (resp. $y_1(u) \leqslant -1$)

GPAC (General Purpose Analog Computer)

Définition

Un langage $\mathcal{L}$ est reconnu en temps polynomial par un GPAC si et seulement si il existe un vecteur de polynômes p, une fonction d'encodage ψ et un polynôme q tel que pour tout mot w, la solution y au système :

$$\begin{cases} y'(t) = p(y(t)) \\ y(0) = \psi(w) \end{cases}$$

satisfait :

- y est définie sur $\mathbb{R}$
- si $w \in \mathcal{L}$ (resp. $w \notin \mathcal{L}$) alors $\exists t > 0$ t.q. $y_1(t) \geqslant 1$ (resp. $y_1(t) \leqslant -1$) et $\mathrm{length}(y)([0,t]) < q(|w|)$
- si $y_1(t) \geqslant 1$ (resp. $y_1(t) \leqslant -1$) alors $\forall u \geqslant t$, $y_1(u) \geqslant 1$ (resp. $y_1(u) \leqslant -1$)
- $\forall t \geqslant 0$, $\mathrm{length}(y)([0,t]) \geqslant t$

Simulation des machines de Turing ($P \subset P_{GPAC}$) :

Simulation des machines de Turing ($P \subset P_{GPAC}$) :

Configuration : $c = (x, s, y, q, z)$

Simulation des machines de Turing ($P \subset P_{GPAC}$) :

Configuration : $c = (x, s, y, q, z)$

$$\downarrow$$

Fonction d'itération : step : $\mathbb{R}^5 \to \mathbb{R}^5$

- Suffisamment simple (fonctions analytiques...)
- Peu sensible aux erreurs

Simulation des machines de Turing ($P \subset P_{GPAC}$) :

Configuration : $c = (x, s, y, q, z)$

$$\downarrow$$

Fonction d'itération : step : $\mathbb{R}^5 \to \mathbb{R}^5$

- Suffisamment simple (fonctions analytiques...)
- Peu sensible aux erreurs

$$\downarrow$$

Itération par ODE : $y'(t) = p(y(t))$

- $y(n)$ désigne step$^{[n]}(c)$

Objectif

Produire une caractérisation de PSPACE par GPAC/équations différentielles.

Caractérisation 1 : oracles

Simulation des machines de Turing avec oracle $\mathcal{O}$

Simulation des machines de Turing avec oracle $\mathcal{O}$

Configuration : $c = (x, s, y, q, z)$ (inchangée)

Simulation des machines de Turing avec oracle $\mathcal{O}$

Configuration : $c = (x, s, y, q, z)$ (inchangée)

$$\downarrow$$

Fonction d'itération : $\text{step}_{\mathcal{O}} : \mathbb{R}^5 \to \mathbb{R}^5$

- Quasiment inchangée
- Injection de la fonction d'oracle $\mathcal{O}$ à l'intérieur

Simulation des machines de Turing avec oracle $\mathcal{O}$

Configuration : $c = (x, s, y, q, z)$ (inchangée)

$$\downarrow$$

Fonction d'itération : $\text{step}_{\mathcal{O}} : \mathbb{R}^5 \to \mathbb{R}^5$

- Quasiment inchangée
- Injection de la fonction d'oracle $\mathcal{O}$ à l'intérieur

$$\downarrow$$

Itération par ODE : $y' = p(y, \mathcal{O}(y))$

- Injection de $\mathcal{O}$ dans l'équation

Caractérisation 1

PSPACE est inclue dans l'ensemble des langages reconnus en temps polynomial par une équation différentielle de la forme :

$$y'(t) = p(y, o(y))$$

où p est un vecteur de polynômes et o une fonction associant à tout encodage réel d'une QBF la valeur 1 si la QBF est vraie, 0 sinon.

Caractérisation 2 : machines à Vecteurs

Définition

Une *machines à Vecteurs* est un ensemble de registres R_1, R_2... (entiers *relatifs* de taille *arbitraire*) et un programme P construit à partir de :

Instruction	Description
$R_a \leftarrow x$	Chargement direct
$if\ \ R_a = 0\ \ goto\ \ j$	Conditionnelle
$accept/reject$	Etats d'arrêt
$R_a \leftarrow \neg R_a$	Négation bit à bit
$R_a \leftarrow R_b \wedge R_c$	Conjonction bit à bit
$R_a \leftarrow R_b \uparrow R_c$	Shift gauche
$R_a \leftarrow R_b \downarrow R_c$	Shift droite

Définition

Une *machines à Vecteurs* est un ensemble de registres R_1, R_2... (entiers *relatifs* de taille *arbitraire*) et un programme P construit à partir de :

Instruction	Description
$R_a \leftarrow x$	Chargement direct
$if \;\; R_a = 0 \;\; goto \;\; j$	Conditionnelle
$accept/reject$	Etats d'arrêt
$R_a \leftarrow \neg R_a$	Négation bit à bit
$R_a \leftarrow R_b \wedge R_c$	Conjonction bit à bit
$R_a \leftarrow R_b \uparrow R_c$	Shift gauche
$R_a \leftarrow R_b \downarrow R_c$	Shift droite

Thèse du calcul parallèle et machines à Vecteurs

L'ensemble des langages reconnus en temps polynomial par machines à Vecteurs est PSPACE.

Programme P avec s instructions et p registres :

Programme P avec s instructions et p registres :

Configuration : $c = (R_1, R_2 ... R_p, i)$

- i : numéro de l'instruction en cours

Programme P avec s instructions et p registres :

Configuration : $c = (R_1, R_2 ... R_p, i)$

- i : numéro de l'instruction en cours

$$\downarrow$$

Fonction d'itération :

$$step(c) = step\begin{pmatrix} R_1 \\ R_2 \\ ... \\ R_p \\ i \end{pmatrix} = \sum_{j=1}^{s} \mathbb{1}(i-j) \cdot step_j(c)$$

- $step_j(c)$ applique la $j^{ème}$ instruction à c
- $\mathbb{1}(i-j) = 0$ sauf lorsque $i = j$

$$\begin{pmatrix} R_1 \\ R_2 \\ \dots \\ R_p \\ i \end{pmatrix} \longrightarrow \begin{pmatrix} R_1 \\ R_5 \cdot 10^{R_4} \\ \dots \\ R_p \\ i+1 \end{pmatrix}$$

$$R_2 \leftarrow R_5 \uparrow R_4$$

$$\begin{pmatrix} R_1 \\ R_2 \\ \dots \\ R_p \\ i \end{pmatrix} \longrightarrow \begin{pmatrix} R_1 \\ R_5 \cdot 10^{R_4} \\ \dots \\ R_p \\ i+1 \end{pmatrix} \qquad \begin{pmatrix} R_1 \\ R_2 \\ \dots \\ R_p \\ i \end{pmatrix} \longrightarrow \begin{pmatrix} x \\ R_2 \\ \dots \\ R_p \\ i+1 \end{pmatrix}$$

$$R_2 \leftarrow R_5 \uparrow R_4 \qquad R_1 \leftarrow x$$

$$\begin{pmatrix} R_1 \\ R_2 \\ \dots \\ R_p \\ i \end{pmatrix} \longrightarrow \begin{pmatrix} R_1 \\ R_5 \cdot 10^{R_4} \\ \dots \\ R_p \\ i+1 \end{pmatrix} \qquad \begin{pmatrix} R_1 \\ R_2 \\ \dots \\ R_p \\ i \end{pmatrix} \longrightarrow \begin{pmatrix} x \\ R_2 \\ \dots \\ R_p \\ i+1 \end{pmatrix}$$

$$R_2 \leftarrow R_5 \uparrow R_4 \qquad R_1 \leftarrow x$$

Et la conjonction $conj : \mathbb{R}^2 \longrightarrow \mathbb{R}$?

$$\begin{pmatrix} R_1 \\ R_2 \\ \dots \\ R_p \\ i \end{pmatrix} \longrightarrow \begin{pmatrix} R_1 \\ R_5 \cdot 10^{R_4} \\ \dots \\ R_p \\ i+1 \end{pmatrix} \qquad \begin{pmatrix} R_1 \\ R_2 \\ \dots \\ R_p \\ i \end{pmatrix} \longrightarrow \begin{pmatrix} x \\ R_2 \\ \dots \\ R_p \\ i+1 \end{pmatrix}$$

$$R_2 \leftarrow R_5 \uparrow R_4 \qquad R_1 \leftarrow x$$

Et la conjonction $conj : \mathbb{R}^2 \longrightarrow \mathbb{R}$?
$\longrightarrow$ obliger de l'injecter directement dans l'équation différentielle

$$\begin{pmatrix} R_1 \\ R_2 \\ \dots \\ R_p \\ i \end{pmatrix} \longrightarrow \begin{pmatrix} R_1 \\ R_5 \cdot 10^{R_4} \\ \dots \\ R_p \\ i+1 \end{pmatrix} \qquad \begin{pmatrix} R_1 \\ R_2 \\ \dots \\ R_p \\ i \end{pmatrix} \longrightarrow \begin{pmatrix} x \\ R_2 \\ \dots \\ R_p \\ i+1 \end{pmatrix}$$

$$R_2 \leftarrow R_5 \uparrow R_4 \qquad R_1 \leftarrow x$$

Et la conjonction *conj* : $\mathbb{R}^2 \longrightarrow \mathbb{R}$?
$\longrightarrow$ obliger de l'injecter directement dans l'équation différentielle

Caractérisation 2 (conjecture)

PSPACE est inclue dans l'ensemble des langages reconnus en temps polynomial par une équation différentielle de la forme :

$$y'(t) = p(y, conj(y))$$

où p est un vecteur de polynômes et *conj* une fonction capable d'effectuer la conjonction bit à bit.

Caractérisation 3 : automates arithmétiques

Définition

Une *RAM arithmétique* est un ensemble de registres R_1, R_2... (entiers *relatifs* de taille *arbitraire*) et un programme P construit à partir de :

Instruction	Description
$R_a \leftarrow x$	Chargement direct
$if \;\; R_a = 0 \;\; goto \;\; j$	Conditionnelle
$accept/reject$	Etats d'arrêt
$R_a \leftarrow R_{R_b}$	Lecture indirecte
$R_{R_a} \leftarrow R_b$	Chargement indirect
$R_a \leftarrow R_b + R_c$	Addition
$R_a \leftarrow R_b - R_c$	Soustraction $(a - b = max(0, a - b))$
$R_a \leftarrow R_b \times R_c$	Multiplication
$R_a \leftarrow R_b \div R_c$	Division entière

PSPACE et RAM arithmétiques

L'ensemble des langages reconnus en temps polynomial par RAM arithmétiques est PSPACE.

PSPACE et RAM arithmétiques

L'ensemble des langages reconnus en temps polynomial par RAM arithmétiques est PSPACE.

Problème

- L'adressage indirect ne permet plus de borner le nombre de registres nécessaires à un programme

PSPACE et RAM arithmétiques

L'ensemble des langages reconnus en temps polynomial par RAM arithmétiques est PSPACE.

Problème

- L'adressage indirect ne permet plus de borner le nombre de registres nécessaires à un programme
- On ne peut pas définir une configuration par l'ensemble des registres utilisés

PSPACE et RAM arithmétiques

L'ensemble des langages reconnus en temps polynomial par RAM arithmétiques est PSPACE.

Problème

- L'adressage indirect ne permet plus de borner le nombre de registres nécessaires à un programme
- On ne peut pas définir une configuration par l'ensemble des registres utilisés

$\longrightarrow$ modifier le modèle des RAM arithmétiques

Automates arithmétiques

Objectifs

- Associer à chaque RAM arithmétique M une fonction $\delta_M : (R_{i-1}, R_i, R_{i+1}) \to R'_i$

Automates arithmétiques

Objectifs

- Associer à chaque RAM arithmétique M une fonction $\delta_M : (R_{i-1}, R_i, R_{i+1}) \to R_i'$
- Définir un pas de calcul comme étant l'application simultanée de δ_M sur tous les registres ($\approx$ automates cellulaires)

Automates arithmétiques

Objectifs

- Associer à chaque RAM arithmétique M une fonction $\delta_M : (R_{i-1}, R_i, R_{i+1}) \rightarrow R_i'$
- Définir un pas de calcul comme étant l'application simultanée de δ_M sur tous les registres ($\approx$ automates cellulaires)

Définition

- On associe à chaque registre R : un état q et une valeur v

Automates arithmétiques

Objectifs

- Associer à chaque RAM arithmétique M une fonction $\delta_M : (R_{i-1}, R_i, R_{i+1}) \to R'_i$
- Définir un pas de calcul comme étant l'application simultanée de δ_M sur tous les registres ($\approx$ automates cellulaires)

Définition

- On associe à chaque registre R : un état q et une valeur v
- Fonction $\delta : (Q \times \mathbb{N})^3 \to Q \times \mathbb{N}$ la plus simple possible :
  - n'être fonction que des états q et de tests $=0$ sur les valeurs v
  - produire une combinaison arithmétique $(+, -, \times, \div)$ des valeurs adjacentes

Simulation des RAM arithmétiques par les automates

Tout programme s'exécutant en temps polynomial sur une RAM arithmétique peut être exécuté en temps polynomial sur un automate arithmétique.

Simulation des RAM arithmétiques par les automates

Tout programme s'exécutant en temps polynomial sur une RAM arithmétique peut être exécuté en temps polynomial sur un automate arithmétique.

PSPACE et automates arithmétiques

L'ensemble des langages reconnus en temps polynomial par automates arithmétiques est inclue dans PSPACE.

Simulation par équations différentielles

Simulation par équations différentielles

Configuration : $(x_j)_j \in (\mathbb{R}^2)^{\mathbb{Z}}$

- x_j représente R_j ($j \in \mathbb{Z}$)

Simulation par équations différentielles

Configuration : $(x_j)_j \in (\mathbb{R}^2)^{\mathbb{Z}}$

- x_j représente R_j ($j \in \mathbb{Z}$)

$$\downarrow$$

Fonction d'itération : $\delta : (\mathbb{R}^2)^3 \to \mathbb{R}^2$

- A appliquer simultanément à tous les x_j

Simulation par équations différentielles

Configuration : $(x_j)_j \in (\mathbb{R}^2)^{\mathbb{Z}}$

- x_j représente R_j ($j \in \mathbb{Z}$)

$$\downarrow$$

Fonction d'itération : $\delta : (\mathbb{R}^2)^3 \to \mathbb{R}^2$

- A appliquer simultanément à tous les x_j

$$\downarrow$$

Itération par équation différentielle :

$$\frac{\partial y(t,x)}{\partial t} = p(y(t,x-1), y(t,x), y(t,x+1))$$

- $y(n,v) = R_v$ au temps n

Simulation par équations différentielles

Configuration : $(x_j)_j \in (\mathbb{R}^2)^{\mathbb{Z}}$

- x_j représente R_j $(j \in \mathbb{Z})$

$$\downarrow$$

Fonction d'itération : $\delta : (\mathbb{R}^2)^3 \to \mathbb{R}^2$

- A appliquer simultanément à tous les x_j
- Contrôle des erreurs ?

$$\downarrow$$

Itération par équation différentielle :

$$\frac{\partial y(t,x)}{\partial t} = p(y(t,x-1), y(t,x), y(t,x+1))$$

- $y(n,v) = R_v$ au temps n

Caractérisation 3 (conjecture)

PSPACE est inclue dans l'ensemble des langages reconnus en temps polynomial par une équation différentielle de la forme :

$$\frac{\partial y(t,x)}{\partial t} = p(y(t,x-1), y(t,x), y(t,x+1))$$

où $p : \mathbb{R}^{d^3} \rightarrow \mathbb{R}^d$ est un vecteur de polynômes.

Conclusion

PSPACE est inclue dans l'ensemble des langages reconnus en temps polynomial par une équation différentielle de la forme :

Caractérisation 1 :

$$y'(t) = p(y, o(y))$$

où o est une fonction d'oracle sur QBF.

Caractérisation 2 (conj.) :

$$y'(t) = p(y, conj(y))$$

où *conj* effectue la conjonction bit à bit.

Caractérisation 3 (conj.) :

$$\frac{\partial y(t,x)}{\partial t} = p(y(t,x-1), y(t,x), y(t,x+1))$$