A Calculus for Modular Loop Acceleration

paper written by Florian Frohn

Vincent Mihalkovic
456247@mail.muni.cz

Faculty of Informatics, Masaryk University

December 1, 2021

Intruduction
Acceleration techniques

Where? static analyses for programs operating on integers.

How? extract a quantifier-free first-order formula ψ from a single-path loop $\mathcal{T}$.

Why? proving safety, reachability, deducing bounds, proving (non-)termination.

This paper:

- existing acceleration techniques only apply if certain prerequisites are in place.
- introduce a calculus which allows for combining several acceleration techniques modularly.
- two novel acceleration techniques .

Preliminaries

- $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \ldots$ for vectors
  - $\boldsymbol{a} := \begin{pmatrix} a_1 \\ \ldots \\ a_d \end{pmatrix}$
- Let $\mathscr{C}(\boldsymbol{z})$ be the set of *closed-form expressions* ofver the variables $\boldsymbol{z}$ containing, e.g., all arithemtic expressions built from $\boldsymbol{z}$, integer constants, addition, subtraction, multipication, division and exponentiation.
- We identify $\mathcal{T}_{\text{loop}}$ (the set of all such loops) and the pair $\langle \varphi, \boldsymbol{a} \rangle$.
  - **while** φ **do** $\boldsymbol{x} \leftarrow \boldsymbol{a}$
    - $\varphi \in \text{Prop}(\mathscr{C}(\boldsymbol{x}))$ is a finite propositional formula over the atoms $\{p > 0 \mid p \in \mathscr{C}(\boldsymbol{x})\}$. We identify the formula $\varphi(\boldsymbol{x})$ and the predicate $\boldsymbol{x} \mapsto \varphi$
    - $\boldsymbol{a} \in \mathscr{C}(\boldsymbol{x})^d$
    - function $\boldsymbol{x} \mapsto \boldsymbol{a}$ maps integers to integers. We write $\boldsymbol{a}(\boldsymbol{x})$ to make the variables $\boldsymbol{x}$ explicit.
    - e.g.: while $\underbrace{x_1 > 0}_{\varphi}$ do $\underbrace{\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}}_{\boldsymbol{x}} \leftarrow \underbrace{\begin{pmatrix} x_1 - 1 \\ 2 \cdot x_2 \end{pmatrix}}_{\boldsymbol{a}}$ $\qquad (\mathcal{T}_{\text{loop}})$

Preliminaries

Throughout this presentation, let n be a designated variable and let:

$$\boldsymbol{a} := \begin{pmatrix} a_1 \\ \cdots \\ a_d \end{pmatrix} \quad \boldsymbol{x} := \begin{pmatrix} x_1 \\ \cdots \\ x_d \end{pmatrix} \quad \boldsymbol{x}' := \begin{pmatrix} x'_1 \\ \cdots \\ x'_d \end{pmatrix} \quad \boldsymbol{y} := \begin{pmatrix} \boldsymbol{x} \\ n \\ \boldsymbol{x}' \end{pmatrix}$$

Intuitively, the variable n represents the number of loop iterations and $\boldsymbol{x}'$ corresponds to the values of the program variables $\boldsymbol{x}$ after n iterations.

- $\mathcal{T}_{\text{loop}}$ induces a relation $\longrightarrow_{\mathcal{T}_{\text{loop}}}$ on $\mathbb{Z}^d$:

$$\varphi(x) \wedge x' = a(x) \iff x \longrightarrow_{\mathcal{T}_{\text{loop}}} x'$$

- Our goal is to find a formula $\psi \in \text{Prop}(\mathscr{C}(\boldsymbol{y}))$ such that

$$\psi \iff x \longrightarrow^{n}_{\mathcal{T}_{\text{loop}}} \boldsymbol{x}' \quad \text{for all } n > 0$$

- Some acceleration techniques cannot guarantee (equiv), but the resulting formula is an under-approximation of $\mathcal{T}_{\text{loop}}$ i.e., we have

$$\psi \Longrightarrow x \longrightarrow^{n}_{\mathcal{T}_{\text{loop}}} \boldsymbol{x}' \quad \text{for all } n > 0$$

If (equiv) resp. (approx) holds, then ψ is equivalent to resp. approximates $\mathcal{T}_{\text{loop}}$.

Acceleration techniques

An **acceleration technique** is a partial function

$$\text{accel} : \text{Loop} \rightharpoonup \text{Prop}(\mathscr{C}(\boldsymbol{y})).$$

sound if accel $(\mathcal{T})$ approximates $\mathcal{T}$ for all $\mathcal{T} \in \text{dom}(\text{ accel })$.

exact if accel $(\mathcal{T})$ is equivalent to $\mathcal{T}$ for all $\mathcal{T} \in \text{dom}(\text{ accel })$.

- All these techniques first compute a *closed form* $c \in \mathscr{C}(\boldsymbol{x}, n)^d$ for the values of the program variables after n iterations.
  - We call $c \in \mathscr{C}(\boldsymbol{x}, n)^d$ a closed form of $\mathcal{T}_{\text{loop}}$ if

$$\forall x \in \mathbb{Z}^d, n \in \mathbb{N}. c = a^n(x)$$

  - Here, a^n is the n-fold application of a, i.e., $a^0(x) = x$ and $a^{n+1}(x) = a(a^n(x))$.

Acceleration via Monotonic Decrease

If $\varphi(\boldsymbol{a}(\boldsymbol{x}))$ implies $\varphi(\boldsymbol{x})$ and $\varphi\left(\boldsymbol{a}^{n-1}(\boldsymbol{x})\right)$ holds, then $\mathcal{T}_{\text{loop}}$ is applicable at least n times.
So in other words: $I_\varphi : \mathbb{Z}^d \to \{0,1\}$ of φ with $I_\varphi(\boldsymbol{x}) = 1 \iff \varphi(\boldsymbol{x})$ is monotonically decreasing w.r.t. $\boldsymbol{a}$, i.e., $I_\varphi(\boldsymbol{x}) \geqslant I_\varphi(\boldsymbol{a}(\boldsymbol{x}))$.
Theorem 1: If

$$\varphi(\boldsymbol{a}(\boldsymbol{x})) \Longrightarrow \varphi(\boldsymbol{x})$$

then the following acceleration technique is exact:

$$\mathcal{T}_{\text{loop}} \mapsto \boldsymbol{x}' = \boldsymbol{a}^n(\boldsymbol{x}) \wedge \varphi\left(\boldsymbol{a}^{n-1}(\boldsymbol{x})\right)$$

Limitations:

$$\text{while } x_1 > 0 \wedge x_2 > 0 \text{ do} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leftarrow \begin{pmatrix} x_1 - 1 \\ x_2 + 1 \end{pmatrix} \qquad (\mathcal{T}_{\text{non-dec}})$$

It cannot be accelerated with Thm. 1 as

$$x_1 - 1 > 0 \wedge x_2 + 1 > 0 \nRightarrow x_1 > 0 \wedge x_2 > 0$$

Example recalled

So for example, Thm. 1 accelerates $\mathcal{T}_{\text{exp}}$ to ψ_{exp} .

$$\text{while } x_1 > 0 \text{ do } \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leftarrow \begin{pmatrix} x_1 - 1 \\ 2 \cdot x_2 \end{pmatrix} \qquad (\mathcal{T}_{\text{exp}})$$

Since

$$\underbrace{x_1 - 1 > 0}_{\varphi(\boldsymbol{a}(\boldsymbol{x}))} \Rightarrow \underbrace{x_1 > 0}_{\varphi(\boldsymbol{x})}$$

an acceleration technique synthesizes, e.g., the formula

$$\underbrace{\begin{pmatrix} x_1' \\ x_2' \end{pmatrix}}_{\boldsymbol{x}'} = \underbrace{\begin{pmatrix} x_1 - n \\ 2^n \cdot x_2 \end{pmatrix}}_{\boldsymbol{a}^n(\boldsymbol{x})} \wedge \underbrace{x_1 - n + 1 > 0 \qquad (\psi_{\text{exp}})}_{\varphi\left(\boldsymbol{a}^{n-1}(\boldsymbol{x})\right)}$$

Acceleration via Monotonic Increase

Theorem 2: If

$$\varphi(\boldsymbol{x}) \implies \varphi(\boldsymbol{a}(\boldsymbol{x}))$$

then the following acceleration technique is exact:

$$\mathcal{T}_{\text{loop}} \mapsto \boldsymbol{x}' = \boldsymbol{a}^n(\boldsymbol{x}) \wedge \varphi(\boldsymbol{x})$$

As a minimal example, Thm. 2 accelerates

$$\text{while } x > 0 \text{ do } x \leftarrow x + 1$$

to

$$x' = x + n \wedge x > 0$$

Acceleration via Decrease and Increase

Theorem 3: If

$$\begin{aligned}
\varphi(\boldsymbol{x}) &\iff \varphi_1(\boldsymbol{x}) \land \varphi_2(\boldsymbol{x}) \land \varphi_3(\boldsymbol{x}) \\
\varphi_1(\boldsymbol{x}) &\implies \varphi_1(\boldsymbol{a}(\boldsymbol{x})) \\
\varphi_1(\boldsymbol{x}) \land \varphi_2(\boldsymbol{a}(\boldsymbol{x})) &\implies \varphi_2(\boldsymbol{x}) \\
\varphi_1(\boldsymbol{x}) \land \varphi_2(\boldsymbol{x}) \land \varphi_3(\boldsymbol{x}) &\implies \varphi_3(\boldsymbol{a}(\boldsymbol{x}))
\end{aligned}$$

then the following acceleration technique is exact:

$$\mathcal{T}_{\text{loop}} \mapsto \boldsymbol{x}' = \boldsymbol{a}^n(\boldsymbol{x}) \land \varphi_1(\boldsymbol{x}) \land \varphi_2\left(\boldsymbol{a}^{n-1}(\boldsymbol{x})\right) \land \varphi_3(\boldsymbol{x})$$

Here, φ_1 and φ_3 are again invariants of the loop. Thus, as in Thm. 2 it suffices to require that they hold before entering the loop. On the other hand, φ_2 needs to satisfy a similar condition as in Thm. 1 and thus it suffices to require that φ_2 holds before the last iteration. We also say that φ_2 is a converse invariant (w.r.t. φ_1). It is easy to see that Thm. 3 is equivalent to Thm. 1 if $\varphi_1 \equiv \varphi_3 \equiv \top$ (where T denotes logical truth) and it is equivalent to Thm. 2 if $\varphi_2 \equiv \varphi_3 \equiv \top$.

Calculus for Modular Loop Acceleration

- All acceleration techniques presented so far are monolithic: Either they accelerate a loop successfully or they fail completely.
- In other words, we cannot combine several techniques to accelerate a single loop.
- Calculus that repeatedly applies acceleration techniques to simplify an *acceleration problem* resulting from a loop $\mathcal{T}_{\mathsf{loop}}$ until it is *solved* and hence gives rise to a suitable $\psi \in \mathrm{Prop}(\mathscr{C}(\boldsymbol{y}))$ which approximates resp. is equivalent to $\mathcal{T}_{\mathsf{loop}}$.

Acceleration Problem

Definition 3 A tuple

$$[\![\psi \mid \check{\varphi} \mid \hat{\varphi} \mid \boldsymbol{a}]\!]$$

where

- $\psi \in \mathrm{Prop}(\mathscr{C}(\boldsymbol{y}))$,the partial result that has been computed so far
- $\hat{\varphi} \in \mathrm{Prop}(\mathscr{C}(\boldsymbol{x}))$,the part of the loop condition that remains to be processed (ψ always approximates $\langle\check{\varphi}, \boldsymbol{a}\rangle$)
- $\check{\varphi} \in \mathrm{Prop}(\mathscr{C}(\boldsymbol{x}))$,the part of the loop condition that has already been processed successfully (loop $\langle\hat{\varphi}, \boldsymbol{a}\rangle$ still needs to be accelerated)
- $\boldsymbol{a} : \mathbb{Z}^d \to \mathbb{Z}^d$

The canonical acceleration problem of a loop $\mathcal{T}_{\text{loop}}$ is

$$[\![\boldsymbol{x}' = \boldsymbol{a}^n(\boldsymbol{x}) | \top | \varphi(\boldsymbol{x}) \mid \boldsymbol{a}(\boldsymbol{x})]\!]$$

Possible states:

- consistent if ψ approximates $\langle\check{\varphi}, a\rangle$,
- exact if ψ is equivalent to $\langle\check{\varphi}, \boldsymbol{a}\rangle$,
- solved if it is consistent and $\hat{\varphi} \equiv \top$.

The goal of our calculus is to transform a canonical into a solved acceleration problem.

Acceleration problem

When we have simplified a canonical acceleration problem

from $[\![\boldsymbol{x}' = \boldsymbol{a}^n(\boldsymbol{x}) \mid \top \mid \varphi(\boldsymbol{x}) \mid \boldsymbol{a}(\boldsymbol{x})]\!]$

to $[\![\psi_1(\boldsymbol{y}) \mid \check{\varphi}(\boldsymbol{x}) \mid \widehat{\varphi}(\boldsymbol{x}) \mid \boldsymbol{a}(\boldsymbol{x})]\!]$

then

$$\varphi \equiv \check{\varphi} \wedge \widehat{\varphi} \text{ and } \psi_1 \implies x \longrightarrow^n_{\langle \check{\varphi}, a \rangle} x'$$

Thus, it then suffices to find some $\psi_2 \in \mathrm{Prop}(\mathscr{C}(\boldsymbol{y}))$ such that

$$x \longrightarrow^n_{\langle \check{\varphi}, a \rangle} x' \wedge \psi_2 \implies x \longrightarrow^n_{\langle \widehat{\varphi}, a \rangle} x'$$

The reason is that we have

$$\longrightarrow_{\langle \check{\varphi}, a \rangle} \cap \longrightarrow_{\langle \widehat{\varphi}, a \rangle} = \longrightarrow_{\langle \check{\varphi} \wedge \widehat{\varphi}, a \rangle} = \longrightarrow_{\langle \varphi, a \rangle}$$

and thus

$$\psi_1 \wedge \psi_2 \implies x \longrightarrow^n_{\langle \varphi, a \rangle} x'$$

i.e., $\psi_1 \wedge \psi_2$ approximates $\mathcal{T}_{\text{loop}}$.

Conditional acceleration

Definition 4 We call a partial function

$$\text{accel} : \text{Loop} \times \text{Prop}(\mathscr{C}(\boldsymbol{x})) \rightharpoonup \text{Prop}(\mathscr{C}(\boldsymbol{y}))$$

a conditional acceleration technique.

sound if

$$\boldsymbol{x} \longrightarrow^{n}_{\langle \check{\varphi}, \boldsymbol{a} \rangle} \boldsymbol{x}' \wedge \text{accel}(\langle \chi, \boldsymbol{a} \rangle, \check{\varphi}) \quad \text{implies} \quad \boldsymbol{x} \longrightarrow^{n}_{\langle \chi, \boldsymbol{a} \rangle} \boldsymbol{x}'$$

for all $(\langle \chi, \boldsymbol{a} \rangle, \check{\varphi}) \in \text{dom}(\text{ accel }), \boldsymbol{x}, \boldsymbol{x}' \in \mathbb{Z}^d$, and $n > 0$.

exact if additionally

$$\boldsymbol{x} \longrightarrow^{n}_{\langle \chi \wedge \check{\varphi}, a \rangle} \boldsymbol{x}' \quad \text{implies} \quad \text{accel}(\langle \chi, \boldsymbol{a} \rangle, \check{\varphi})$$

for all $(\langle \chi, \boldsymbol{a} \rangle, \check{\varphi}) \in \text{dom}(\text{ accel }), \boldsymbol{x}, \boldsymbol{x}' \in \mathbb{Z}^d$, and $n > 0$

We are now ready to present our acceleration calculus, which combines loop acceleration techniques in a modular way.

Acceleration Calculus

Definition 5 The relation $\rightsquigarrow$ on acceleration problems is defined by the following rule:

$$\frac{\emptyset \neq \chi \subseteq \widehat{\varphi} \qquad \overbrace{\mathsf{accel}(\langle \chi, \boldsymbol{a} \rangle, \check{\varphi})}^{\text{sound CondAccelTechn}} = \psi_2}{[\![\psi_1 \mid \check{\varphi} \mid \widehat{\varphi} \mid \boldsymbol{a}]\!] \rightsquigarrow_{(e)} [\![\psi_1 \cup \psi_2 \mid \check{\varphi} \cup \chi \mid \widehat{\varphi} \setminus \chi \mid \boldsymbol{a}]\!]}$$

- $\rightsquigarrow$ step is exact (written $\rightsquigarrow_e$) if accel is exact.
- our calculus allows us to pick a subset χ (of clauses) from the yet unprocessed condition $\widehat{\varphi}$ and
- "move" it to $\check{\varphi}$, which has already been processed successfully.
- To this end, $\langle \chi, \boldsymbol{a} \rangle$ needs to be accelerated by a conditional acceleration technique, i.e., when accelerating $\langle \chi, \boldsymbol{a} \rangle$ we may assume $\boldsymbol{x} \longrightarrow^{n}_{\langle \check{\varphi}, \boldsymbol{a} \rangle} \boldsymbol{x}'$.

Note that every acceleration technique trivially gives rise to a conditional acceleration technique (by disregarding the second argument $\check{\varphi}$ of accel in Def. 4). Thus, our calculus allows for combining arbitrary existing acceleration techniques without adapting them.

Example

$$\text{while } x_1 > 0 \wedge x_2 > 0 \text{ do } \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leftarrow \begin{pmatrix} x_1 - 1 \\ x_2 + 1 \end{pmatrix} \qquad (\mathcal{T}_{\text{non-dec}})$$

$$\left[\!\!\left[\begin{pmatrix} x_1' \\ x_2' \end{pmatrix} = \begin{pmatrix} x_1 - n \\ x_2 + n \end{pmatrix} \middle| \top \middle| x_1 > 0 \wedge x_2 > 0 \,\middle|\, \begin{pmatrix} x_1 - 1 \\ x_2 + 1 \end{pmatrix} \right]\!\!\right]$$

Acceleration calculus, properties

Lemma 1. $\rightsquigarrow$ preserves consistency and $\rightsquigarrow_e$ preserves exactness.
Then the correctness of our calculus follows immediately. The reason is that

$$[\![\boldsymbol{x}' = \boldsymbol{a}^n(\boldsymbol{x}) | \top | \varphi(\boldsymbol{x}) \,|\boldsymbol{a}(\boldsymbol{x})]\!] \rightsquigarrow^*_{(e)} [\![\psi(\boldsymbol{y})|\, \check{\varphi}(\boldsymbol{x}) | \top | \boldsymbol{a}(\boldsymbol{x})]\!] \text{ implies } \varphi \equiv \check{\varphi}$$

Theorem 5 (Correctness of $\rightsquigarrow$).
If

$$[\![\boldsymbol{x}' = \boldsymbol{a}^n(\boldsymbol{x}) | \top | \varphi(\boldsymbol{x}) \,|\boldsymbol{a}(\boldsymbol{x})]\!] \rightsquigarrow^* [\![\psi(\boldsymbol{y})|\, \check{\varphi}(\boldsymbol{x}) | \top | \boldsymbol{a}(\boldsymbol{x})]\!]$$

then ψ approximates $\mathcal{T}_{\text{loop}}$.

If

$$[\![\boldsymbol{x}' = \boldsymbol{a}^n(\boldsymbol{x}) | \top | \varphi(\boldsymbol{x}) \,|\boldsymbol{a}(\boldsymbol{x})]\!] \rightsquigarrow^*_e [\![\psi(\boldsymbol{y})|\, \check{\varphi}(\boldsymbol{x}) | \top | \boldsymbol{a}(\boldsymbol{x})]\!]$$

then ψ is equivalent to $\mathcal{T}_{\text{loop}}$.

Termination of our calculus is trivial, as the size of the third component $\hat{\varphi}$ of the acceleration problem is decreasing.
Theorem 6 (Termination of $\rightsquigarrow$). $\rightsquigarrow$ terminates.

Conditional Acceleration via Monotonic Decrease

Theorem 7 If

$$\breve{\varphi}(\boldsymbol{x}) \wedge \chi(\boldsymbol{a}(\boldsymbol{x})) \Longrightarrow \chi(\boldsymbol{x})$$

then the following conditional acceleration technique is exact:

$$(\langle \chi, \boldsymbol{a} \rangle, \breve{\varphi}) \mapsto \boldsymbol{x}' = \boldsymbol{a}^n(\boldsymbol{x}) \wedge \chi\left(\boldsymbol{a}^{n-1}(\boldsymbol{x})\right)$$

So we just add $\breve{\varphi}$ to the premise of the implication that needs to be checked to apply acceleration via monotonic decrease. Thm. 2 can be adapted analogously.

Acceleration via Eventual Monotonicity

The combination of the calculus and the conditional acceleration techniques still fails to handle certain interesting classes of loops ...

Acceleration via Eventual Decrease All (combinations of) techniques presented so far fail for the following example.

$$\text{while } x_1 > 0 \text{ do } \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leftarrow \begin{pmatrix} x_1 + x_2 \\ x_2 - 1 \end{pmatrix} \quad (\mathcal{T}_{ev-dec})$$

The reason is that x_1 does not behave monotonically ...

Theorem 10 (*Acceleration via Eventual Decrease*). If $\varphi(x) \equiv \bigwedge_{i=1}^{k} C_i$ where each C_i contains an inequation $\text{expr}_i(\boldsymbol{x}) > 0$ such that

$$\text{expr}_i(\boldsymbol{x}) \geqslant \text{expr}_i(\boldsymbol{a}(\boldsymbol{x})) \implies \text{expr}_i(\boldsymbol{a}(\boldsymbol{x})) \geqslant \text{expr}_i\left(\boldsymbol{a}^2(\boldsymbol{x})\right)$$

then the following acceleration technique is sound:

$$\mathcal{T}_{\text{loop}} \mapsto \boldsymbol{x}' = \boldsymbol{a}^n(\boldsymbol{x}) \wedge \bigwedge_{i=1}^{k} \left(\text{expr}_i(\boldsymbol{x}) > 0 \wedge \text{expr}_i\left(\boldsymbol{a}^{n-1}(\boldsymbol{x})\right) > 0\right)$$

If $C_i \equiv \text{expr}_i > 0$ for all $i \in [1, k]$, then it is exact.

Experiments

- Loop Acceleration Tool - LoAT
- LoAT uses Z3 to check implications and PURRS to compute closed forms.
- To evaluate our approach, they extracted 1511 loops with conjunctive guards from the category *Termination of Integer Transition Systems of the Termination Problems Database* which is used at the annual *Termination and Complexity Competition*
- `Flata`, which implements the techniques to accelerate FMATs and octagonal relations

Experiment tables

	LoAT	Monot.	Meter	Flata
exact	1444	845	0[3]	1231
approx	38	0	733	0
fail	29	666	778	280
avg rt	0.16 s	0.11 s	0.09 s	0.47 s

	~~Ev-Inc~~	~~Ev-Dec~~	~~Ev-Mon~~
exact	1444	845	845
approx	0	493	0
fail	67	173	666
avg rt	0.15 s	0.14 s	0.09 s

MUNI
FACULTY
OF INFORMATICS