POPL 2014 - STUDENT POSTER SESSION

AUTOMATIC INFERENCE OF RANKING FUNCTIONS BY ABSTRACT INTERPRETATION

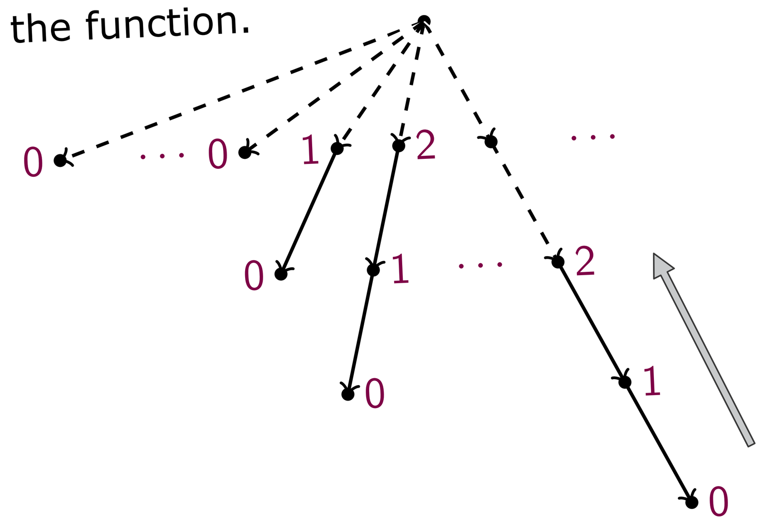
Caterina Urban

École Normale Supérieure, Paris, France

We can define a ranking function in an incremental way: We start from the program final states, where the function has value 0. Then, we add states to the domain of the function, retracing the program backwards and **counting the** maximum **number of program steps** as value of the function.

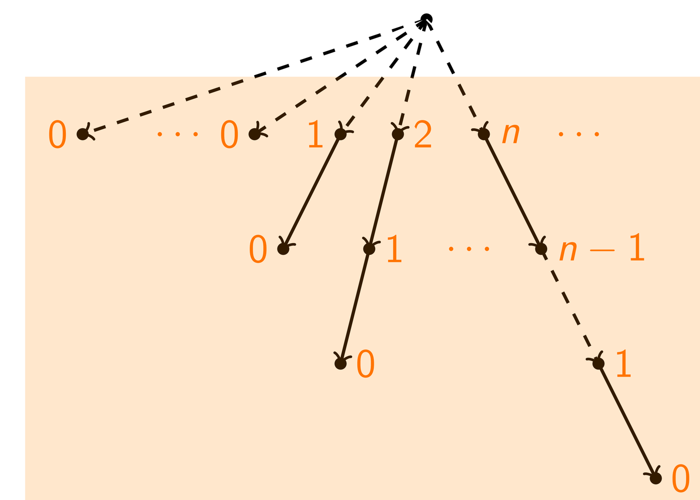
Example

int : x
$x := ?$
while $(x \geq 0)$ do
 $x := x - 1$
od

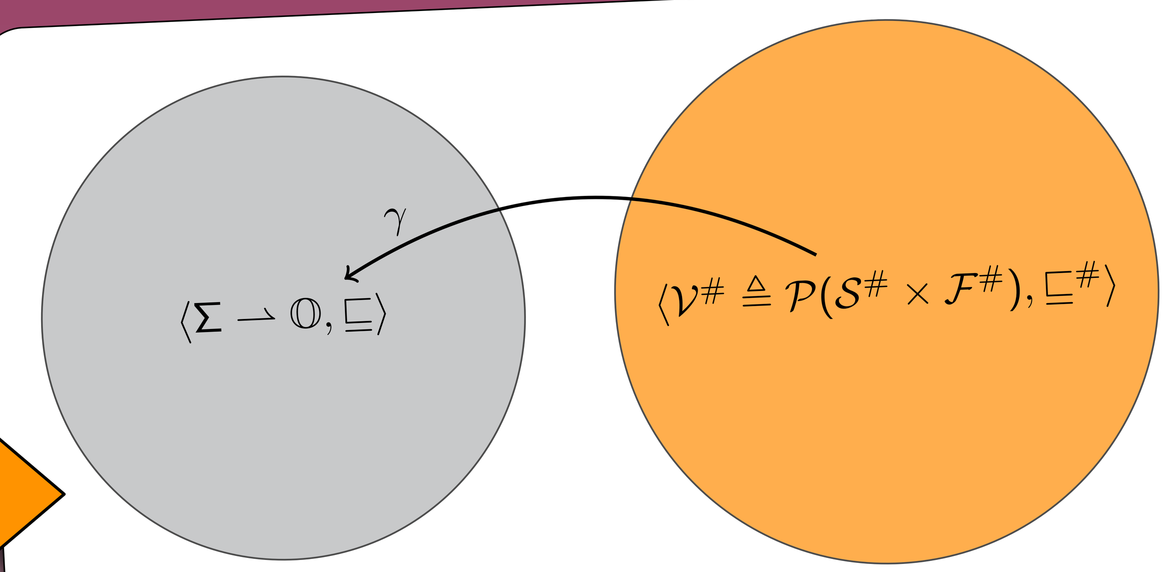


Example

int : x
$x := ?$
while $(x \geq 0)$ do
 $x := x - 1$
od



It is not computable!



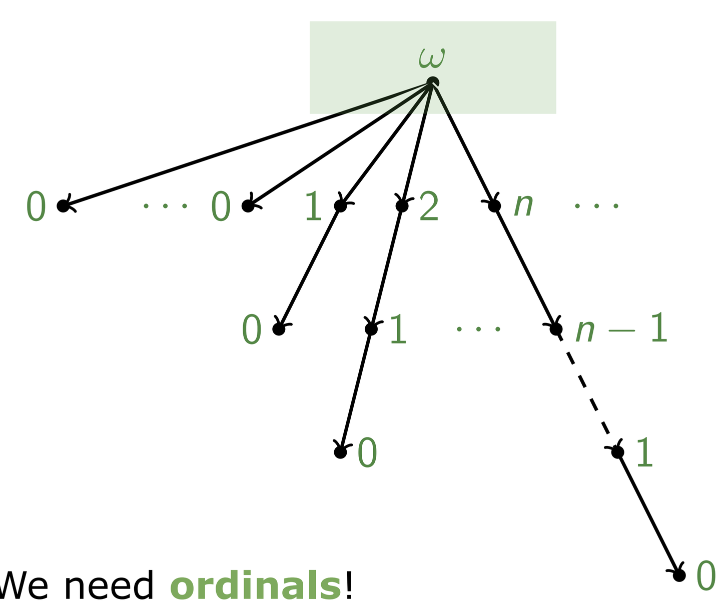
Example

$$v^\#(x) \triangleq \begin{cases} x \in [-\infty, 5] \mapsto \bot_F \\ x \in [6, 8] \mapsto -3x + 38 \\ x \in [9, 10] \mapsto 4 \\ x \in [11, +\infty] \mapsto 1 \end{cases}$$

- $\mathcal{S}^\# \triangleq$ numerical abstract domain
- $\mathcal{F}^\# \triangleq \{\bot_F\} \cup \{f^\# \mid f^\# \in \mathbb{Z}^n \to \mathbb{N}\} \cup \{\top_F\}$ where $f^\# \equiv f(x_1, \ldots, x_n) = m_1 x_1 + \cdots + m_n x_n + q$

Example

int : x
$x := ?$
while $(x \geq 0)$ do
 $x := x - 1$
od



We need **ordinals**!

Example

int : x, y, N
[1]$x := N$
while [2]$(x \geq 0)$ do
 [3]$y := N$
 while [4]$(y \geq 0)$ do
  [5]$y := y - 1$
 od
 [6]$x := x - 1$
od[7]

the loop terminates in a **finite number of iterations**

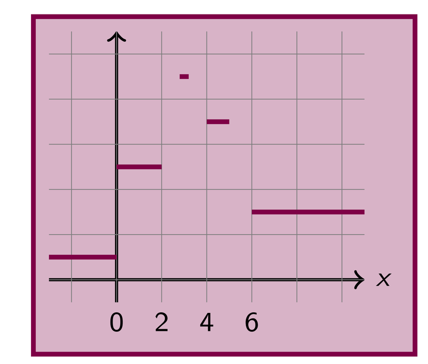
$$v^\#(x, y, N) = \begin{cases} x \in [-\infty, 0] \mapsto 1 \\ x \in [1, +\infty] \mapsto \omega + 2 \end{cases}$$

Handling **ordinals overcomes the limitation of affine functions** in case of programs with non-linear computational complexity.

Why Piecewise-Defined Ranking Functions?

Example

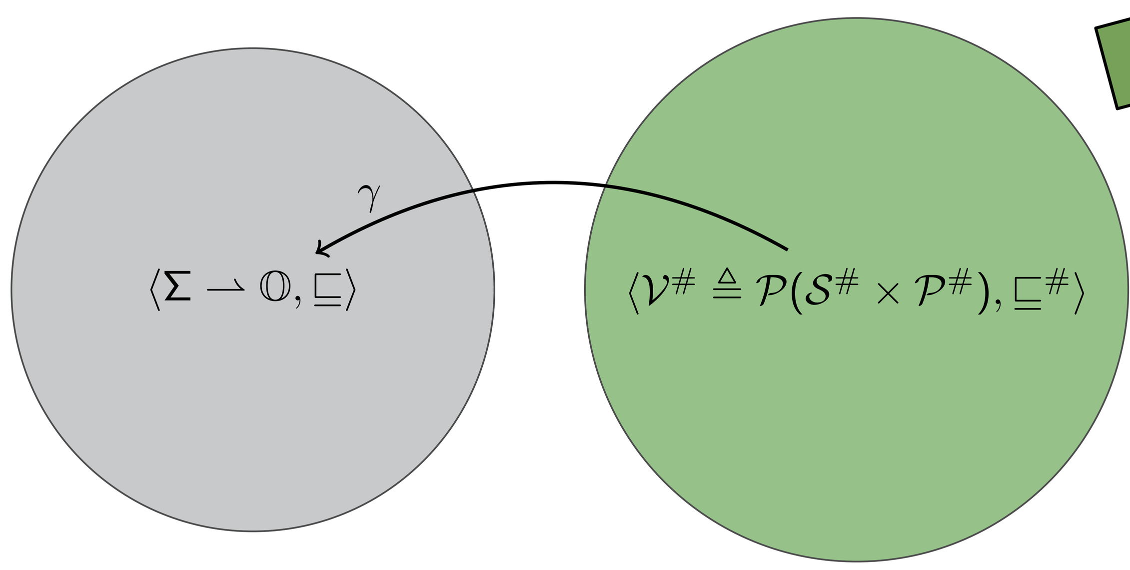
int : x
while [1]$(x \geq 0)$ do
 [2]$x := -2x + 10$
od[3]



$$v^\#(x) \triangleq \begin{cases} 1 & x < 0 \\ 5 & 0 \leq x \leq 2 \\ 9 & x = 3 \\ 7 & 4 \leq x \leq 5 \\ 3 & 5 < x \end{cases}$$

The piecewise-definition of the functions overcomes the non-existence of a linear ranking function for a program. Moreover, it allows the analysis to infer **sufficient preconditions for termination**.

The operators of the abstract domain are combined together, to compute an abstract ranking function for a program, through **backward invariance analysis**. The starting point is the constant function equals to 0 at the program final control point. The ranking function is then propagated backwards towards the program initial control point taking assignments and tests into account with join and widening around loops. **The program states, for which the analysis finds a ranking function, are states from which the program always terminates.**



- $\mathcal{S}^\# \triangleq$ numerical abstract domain
- $\mathcal{P}^\# \triangleq \{\bot_F\} \cup \{\sum_i \omega^i \cdot f_i^\# \mid f_i^\# \in \mathbb{Z}^n \to \mathbb{N}\} \cup \{\top_F\}$ where $f_i^\# \equiv f_i(x_1, \ldots, x_n) = m_{i_1} x_1 + \cdots + m_{i_n} x_n + q_i$

Example

$$v^\#(x) \triangleq \begin{cases} x \in [-\infty, -1], y \in [-\infty, 0] \mapsto 1 \\ x \in [-\infty, -1], y \in [1, +\infty] \mapsto \omega^2 + \omega \cdot (y - 1) - 4x + 9y - 2 \\ x \in [0, 0], y \in [-\infty, +\infty] \mapsto 1 \\ x \in [1, +\infty], y \in [-\infty, 0] \mapsto 1 \\ x \in [1, +\infty), y \in [1, +\infty) \mapsto \omega \cdot (x - 1) + 9x + 4y - 7 \end{cases}$$

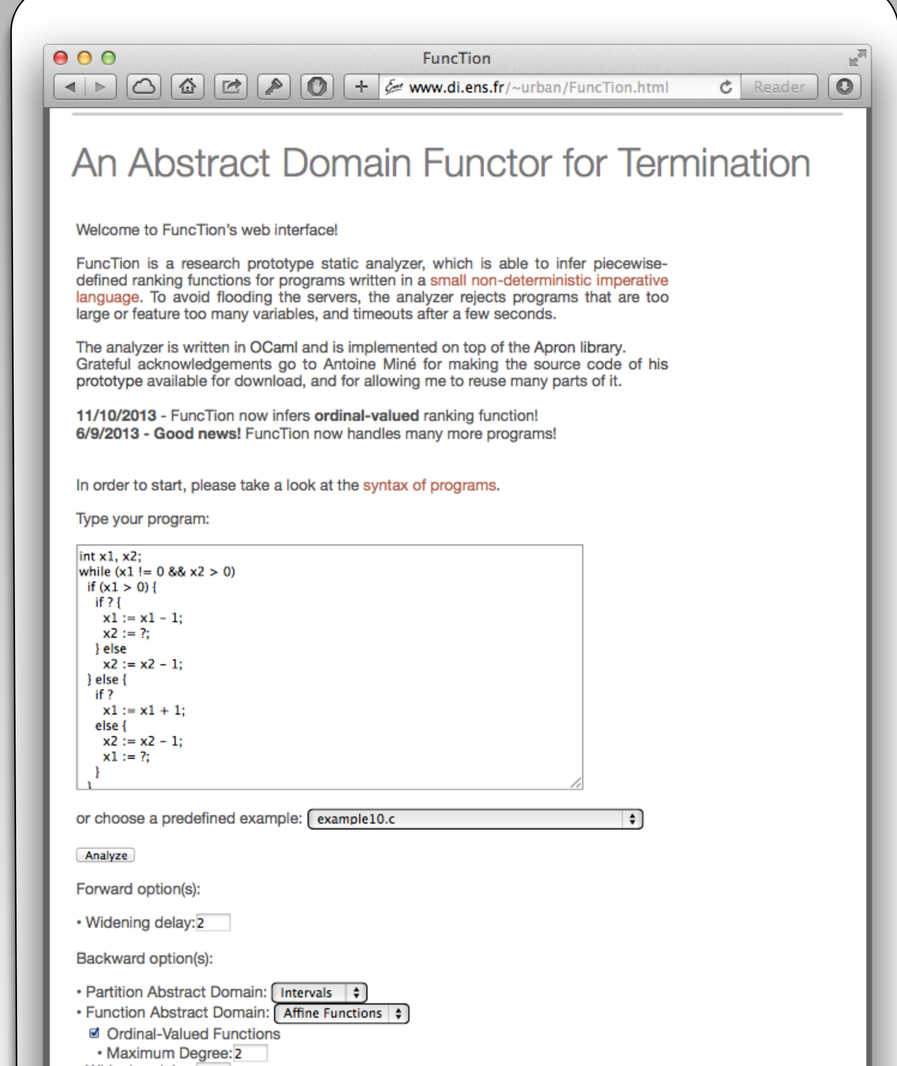
It is also possible to use **lexicographic ranking functions**.

$$\omega^k \cdot \underbrace{f_k^\#}_{\in \mathbb{N}} + \ldots + \omega^2 \cdot \underbrace{f_2^\#}_{\in \mathbb{N}} + \omega \cdot \underbrace{f_1^\#}_{\in \mathbb{N}} + \underbrace{f_0^\#}_{\in \mathbb{N}} \in \mathbb{O}$$

$$\Updownarrow$$

$$(f_k^\#, \ldots, f_2^\#, f_1^\#, f_0^\#) \in \underbrace{\mathbb{N} \times \ldots \times \mathbb{N}}_{k}$$

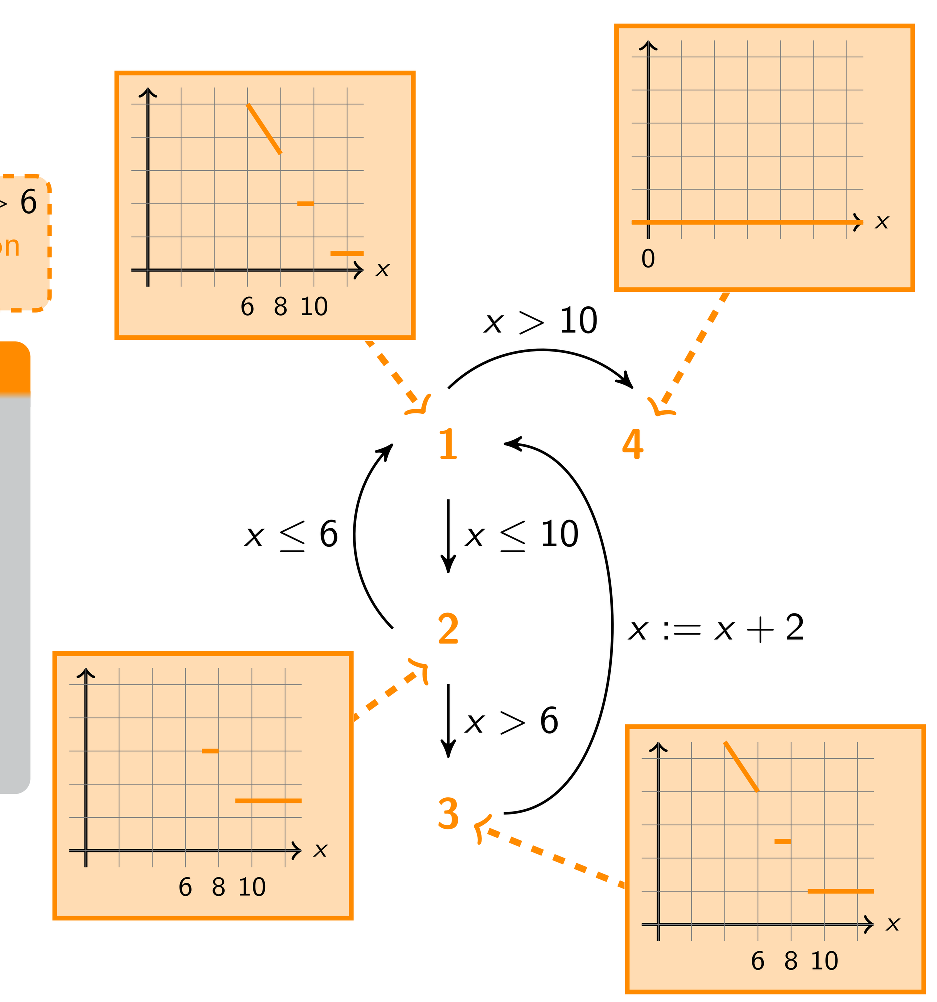
However, reasoning directly with lexicographic ranking functions, poses the additional difficulty of finding an **appropriate lexicographic order**. Instead, with ordinal-valued ranking functions the coefficients and their order are **automatically inferred by the analysis**.



the analysis provides $x > 6$ as sufficient precondition for termination

Example

int : x
while [1]$(x \leq 10)$ do
 if [2]$(x > 6)$ then
  [3]$x := x + 2$
 fi
od[4]



http://www.di.ens.fr/~urban/FuncTion.html