Computing (the smallest) fixed points of monotone maps arising in static analysis by AI

Joint work, over the last 4 years: Stéphane Gaubert[1], Assalé Adjé[2], Eric Goubault[3], Sylvie Putot, Alexandru Costan, Matthieu Martel, Sarah Zennou, Ankur Taly

5th of February 2009, Toulouse

[1]INRIA Saclay, CMAP Ecole Polytechnique
[2]CEA LIST and LIX, MeASI
[3]CEA LIST, MeASI

Static analysis by abstract interpretation (Cousot 77 etc.)

A simple C program

```
int x=0                  //1
    while(x<=99){        //2
          x=1+x;         //3
    }                    //4
```

Find the smallest overapproximation of the values of x at each breakpoint.

Static analysis by abstract interpretation (Cousot 77 etc.)

A simple C program/collecting semantics

```
int x=0                  //1
    while(x<=99){        //2
          x=1+x;         //3
    }                    //4
```

$$
\begin{aligned}
x_1 &= \{0\} \\
x_2 &= (x_1 \cup x_3) \cap]-\infty, 99] \\
x_3 &= 1 + x_2 \\
x_4 &= (x_1 \cup x_3) \cap [100, +\infty[
\end{aligned}
$$

[First part] Find a (smallest?) vector x of intervals which satisfies the fixed point equation. By writing an interval $I = [-i^-, i^+]$ and after reductions:

Static analysis by abstract interpretation (Cousot 77 etc.)

A simple C program/collecting semantics

```
int x=0                    //1
    while(x<=99){          //2
          x=1+x;           //3
    }                      //4
```

$$
\begin{aligned}
x_1 &= \{0\} \\
x_2 &= (x_1 \cup x_3) \cap]-\infty, 99] \\
x_3 &= 1 + x_2 \\
x_4 &= (x_1 \cup x_3) \cap [100, +\infty[
\end{aligned}
$$

Nonlinear fixed point equation given by line 2

$$
\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0, (x_2^- - 1)) \\ \min(99, \max(0, (x_2^+ + 1)) \end{pmatrix}.
$$

Static analysis by abstract interpretation (Cousot 77 etc.)

A simple C program/collecting semantics

```
int x=0                  //1
    while(x<=99){        //2
          x=1+x;         //3
    }                    //4
```

$$
\begin{aligned}
x_1 &= \{0\} \\
x_2 &= (x_1 \cup x_3) \cap]-\infty, 99] \\
x_3 &= 1 + x_2 \\
x_4 &= (x_1 \cup x_3) \cap [100, +\infty[
\end{aligned}
$$

Nonlinear fixed point equation given by line 2

$$
\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0, (x_2^- - 1)) \\ \min(99, \max(0, (x_2^+ + 1)) \end{pmatrix}.
$$

$\Longrightarrow (x_2^-, x_2^+) = (0, 99)$ (How to find it efficiently?)

Static analysis by abstract interpretation (Cousot 77 etc.)

A simple C program/collecting semantics

```
int x,
x=0                  //1
while(x<=99){        //2
      x=1+x;         //3
}                    //4
```

$$
\begin{aligned}
x_1 &= \{0\} \\
x_2 &= (x_1 \cup x_3) \cap]-\infty, 99] \\
x_3 &= 1 + x_2 \\
x_4 &= (x_1 \cup x_3) \cap [100, +\infty[
\end{aligned}
$$

Static analysis by abstract interpretation (Cousot 77 etc.)

A simple C program/collecting semantics

```
int x,
x=0                  //1
while(x<=99){        //2
      x=1+x;         //3
}                    //4
```

$$
\begin{aligned}
x_1 &= \{0\} \\
x_2 &= (x_1 \cup x_3) \cap]-\infty, 99] \\
x_3 &= 1 + x_2 \\
x_4 &= (x_1 \cup x_3) \cap [100, +\infty[
\end{aligned}
$$

[Second part] Extension of these methods to relational domains such as zones, templates, and non-linear templates

Static analysis by abstract interpretation (Cousot 77 etc.)

A simple C program/collecting semantics

```
int x,
x=0                  //1
while(x<=99){        //2
      x=1+x;         //3
}                    //4
```

$$
\begin{aligned}
x_1 &= \{0\} \\
x_2 &= (x_1 \cup x_3) \cap]-\infty, 99] \\
x_3 &= 1 + x_2 \\
x_4 &= (x_1 \cup x_3) \cap [100, +\infty[
\end{aligned}
$$

[Second part] Extension of these methods to relational domains such as zones, templates, and non-linear templates

[Third part] Find the smallest fixpoint of the abstract semantic equations

Method to compute the optimal invariant

- Classical Value Iteration (Kleene):
  - Theoretically guarantees the minimality of the fixed point found.
  - Slow (99 iterations for the previous simple case) and may not be convergent in finite time.
  - Often needs widening/narrowing techniques: minimality no longer guaranteed.
- An alternative: Policy Iteration Algorithm: Howard (60) (stochastic control), extended by Hoffman and Karp (66) (stochastic games). Fast method (akin to Newton's method but complexity unknown in our context).

Extended by Costan, Gaubert, Goubault, Martel and Putot (CAV'05) to fixed point problems in static analysis.
See also Gaubert, Goubault, Taly, Zennou (ESOP'2007), Adjé, Gaubert, Goubault (MTNS'2008) and Seidl, Gawlitza (CSL'2007, ESOP'2007)

Game interpretation

Equation

$$x_2^+ = \min(99, \max(0, (x_2^+ + 1))$$

is interpreted as a simple zero sum repeated game with stopping option:

- (Min) can decide to stop and pays 99 (the game ends) or to let (Max) play.
- Then (Max) can stop and wins 0 (the game ends) or continue to play and wins 1.

Policies on (Min) informally:

(1) if (Min) gives its strategy, max can optimize its gain

(2) (Min) finds its optimal "loss" by minimizing on all previous (Max) gains as in (1)

Example 1

```
x=0                  //1
while(x<=99){        //2
      x=1+x;         //3
}                    //4
```

$$f\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0,(x_2^- - 1)) \\ \min(99, \max(0,(x_2^+ + 1))) \end{pmatrix}.$$

$$\text{i.e. } x_2 =]-\infty, 99] \cap ([0,0] \cup (x_2 + 1))$$

We start with the (simplest?) initial policy:

$$f^{\pi^0}\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0,(x_2^- - 1)) \\ 99 \end{pmatrix}.$$

$$\text{i.e. } x_2 = [min\,([0,0] \cup (x_2 + 1)), 99] \text{ policy rl}$$

The smallest fixed point of this policy is $(0, 99)$ (by LP...).

Example 1

```
x=0                  //1
while(x<=99){        //2
      x=1+x;         //3
}                    //4
```

$$f\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0,(x_2^- - 1)) \\ \min(99, \max(0,(x_2^+ + 1))) \end{pmatrix}.$$

$$\text{i.e. } x_2 =]-\infty, 99] \cap ([0,0] \cup (x_2 + 1))$$

We start with the (simplest?) initial policy:

$$f^{\pi^0}\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0,(x_2^- - 1)) \\ 99 \end{pmatrix}.$$

$$\text{i.e. } x_2 = [min\,([0,0] \cup (x_2 + 1)), 99] \text{ policy rl}$$

The smallest fixed point of this policy is $(0, 99)$ (by LP...).
This fixed point is also a fixed point for f so PI Algorithm stops.
(1 iteration)

Linear programming for "simplified" equations

- We can use Kleene+widening/narrowing like in CAV'2005 for finding the lfp of f^{π^0}
- We can also use linear programming (if the assignments are all linear, or linearized); for instance, finding the lfp of f^{π^0} is equivalent to solving the LP problem (one has to be careful with infinities though...):

$$\begin{gathered}\min x_2^- + x_2^+ \\ x_2^+ \geq 99 \\ x_2^- \geq x_2^- - 1 \\ x_2^- \geq 0\end{gathered}$$

Another example

```
void main(){
int i,j;
i=1; // 1
j=10; // 2
while (j >= i) {
// 3

  i = i+2; // 4
  j = -1+j; // 5

} // 6 }
```

$$\begin{cases} i_3 &=]-\infty, \max(j_2, j_5)] \cap (i_2 \cup i_5) \\ j_3 &= [\min(i_2, i_5), +\infty[\cap (j_2 \cup j_5) \end{cases}$$

$$\cdots$$

$$\cdots$$

$$\begin{cases} i_6 &= [\min(j_2, j_5) + 1, +\infty[\cap (i_2 \cup i_5) \\ j_6 &=]-\infty, \max(i_2, i_5) - 1] \cap (j_2 \cup j_5) \end{cases}$$

Initial policy

rl for i in 3, *lr* for j in 3, *lr* for i in 6 and *rl* for j in 6.

$$
\begin{array}{lcl}
i_3 =]-\infty, \max(j_2, j_5)] \cap (i_2 \cup i_5) & \rightarrow & [(i_2 \cup i_5)^-, \max(j_2, j_5)] \\
j_3 = [\min(i_2, i_5), +\infty[\cap (j_2 \cup j_5) & \rightarrow & [(i_2, i_5)^-, \max(j_2 \cup j_5)] \\
i_6 = [\min(j_2, j_5) + 1, +\infty[\cap (i_2 \cup i_5) & \rightarrow & [(j_2, j_5)^- + 1, \max(i_2 \cup i_5)] \\
j_6 =]-\infty, \max(i_2, i_5) - 1] \cap (j_2 \cup j_5) & \rightarrow & [(j_2 \cup j_5)^-, \max(i_2, i_5) - 1]
\end{array}
$$

In one Kleene iteration, or LP, finds:

$$
\begin{array}{lcl}
i_2 & = & [1, 1] \\
j_2 & = & [10, 10] \\
i_5 & = & [3, 12] \\
j_5 & = & [0, 9] \\
i_6 & = & [\min([10, 10], [0, 9]) + 1, \max([1, 1] \cup [3, 12])] = [1, 12] \\
j_6 & = & [\min([10, 10] \cup [0, 9]), \max([1, 1], [3, 12]) - 1] = [0, 11]
\end{array}
$$

Is this the lfp of the complete equations?

$F(I,J) = (I,J)$?

$$
\begin{aligned}
F(I,J)^i_6 &= [\min(j^1_2, j^1_5) + 1, +\infty[\cap (i^1_2 \cup i^1_5) \\
&= [0, +\infty[\cap [1, 12] \\
&= i^1_6 \text{ [local fixed point!]} \\
F(I,J)^j_6, &=]-\infty, \max(i^1_2, i^1_5) - 1] \cap (j^1_2 \cup j^1_5)) \\
&=]-\infty, 11] \cap [0, 10] \\
&= [0, 10] \neq j^1_6([0, 11])!
\end{aligned}
$$

$\Longrightarrow$ Improvement of policy for j at control point 6 (rr instead of rl)

Policy Iteration Algorithm

$f : \mathcal{L} \mapsto \mathcal{L}$, $\mathcal{L}$ is a complete lattice E.g:
$\mathcal{L} = \bar{\mathbb{R}}^d = \{\mathbb{R} \cup \{-\infty\} \cup \{+\infty\}\}^d$.
Assume $f = \inf_{\pi \in \Pi} f^\pi$ (f^π are minimum free).
The set $\{f^\pi \mid \pi \in \Pi\}$ has the lower selection property.

1. *Init*: Select a strategy π^0, $k = 0$.
2. *Value Determination* ($\mathbf{D_k}$): Compute a fixed point u^k of f^{π^k}.
3. Compute $f(u^k)$.
4. If $f(u^k) = u^k$, return u^k
5. *Policy Improvement* ($\mathbf{I_k}$): If $f(u^k) < u^k$ define π^{k+1} s.t $f(u^k) = f^{\pi^{k+1}}(u^k)$ and go to Step (D_{k+1}).

If we compute, at ($\mathbf{D_k}$) the smallest u^k and if $\{f^\pi \mid \pi$ strategy$\}$ is finite and f^π are order preserving, the algorithm stops.

Second and last policy

- Use equation: $j_6 = j_2 \cup j_5$
- Using K' we find the lfp:

$$\begin{aligned} i_6^2 &= [1, 12] \\ j_6^2 &= [0, 10] \end{aligned}$$

- This is reached by several policies (none of which improves the fixed point here)
- PI: 2 iterations on policies, 5 iterations on values (Kleene K') and 272 elementary operations (addition, comparison etc.)
- Kleene (K) takes 10 iterations (and 476 operations) for the same result.

Policy iteration might even be faster AND more precise than Kleene+widening/narrowing (see CAV'2005, ESOP'2007)

The case of zones

Consider the domain of zones (see A. Miné 2004-)

- Zones on variables $\{v_0, \ldots, v_n\}$ are polyhedra of a special kind: only constraints of the form $v_i - v_j \leq c_{i,j}$ are allowed
- Can be representated by Difference Bound Matrices:
  - A DBM is a $(n+1) \times (n+1)$ matrix whose (i,j) entry is $c_{j,i}$ or $+\infty$ if there is no bound on $v_j - v_i$
  - Example: $v_1 - v_0 \leq 10$, $v_0 - v_1 \leq 0$ (i.e. "morally" v_1 in [0,10] for a start), $v_1 - v_2 \leq 3$, $v_2 - v_0 \leq 2$
  - The corresponding DBM is:

$$\begin{pmatrix} 0 & 10 & 2 \\ 0 & 0 & +\infty \\ +\infty & 3 & 0 \end{pmatrix}$$

Closure and Lattice operations

- Closure
  - Taking "consequences" of inequalities:

$$\begin{pmatrix} 0 & 10 & 2 \\ 0 & 0 & +\infty \\ +\infty & 3 & 0 \end{pmatrix} \Longrightarrow \begin{pmatrix} 0 & 5 & 2 \\ 0 & 0 & +\infty \\ +\infty & 3 & 0 \end{pmatrix}$$

  - $$c^*_{i,j} = \min_{1 \leq k \leq n-1} \left\{ c_{i,i_1} + \ldots + c_{i_{k-1},j} | \forall i_1, \ldots, i_{k-1} \in \{1, \ldots, n\} \right\}$$

    (i.e. $v_i - v_{i_1} \leq c_{i,i_1}, \ldots, v_{i_{k-1}} - v_j \leq c_{i_{k-1},j} \Longrightarrow$
    $v_i - v_j \leq c_{i,i_1} + \ldots + c_{i_{k-1},j}$)
- The $\sqcup$ operation is a pointwise maximum between the entries of the corresponding DBMs; Preserves closure.
- The $\sqcap$ operation is a pointwise minimum operation between the entries of the corresponding DBMs; Does not preserve closure.

Policies for zones (ESOP 2007)

- Intersection policies: as with intervals, one policy left/right for each entry of the DBMs
- Closure policies:
  - new with respect to intervals
  - a policy for closure corresponds a choice of "partial consequences" of the set of inequalities:

$$c^*_{i,j} = \min_{1 \leq k \leq n-1} \left\{ c_{i,i_1} + \ldots + c_{i_{k-1},j} | \forall i_1, \ldots, i_{k-1} \in \{1, \ldots, n\} \right\}$$

$$\Longrightarrow$$

$$\pi(c^*)_{i,j} = c_{i,i_1} + \ldots + c_{i_{k-1},j} \text{ for some } k, \text{ and some } i_1, \ldots, i_{k-1}$$

Example

```
0 i = 150;
1 j = 175;
2 while (j >= 100){
3   i++;
4   if (j<= i){
5     i = i - 1;
6     j = j - 2;
7   }
8 }
9
```

$$
\begin{aligned}
M_0 &= \mathit{context_initialization} \\
M_2 &= (\mathit{Assign}(i \leftarrow 150, j \leftarrow 175)(M_0))^* \\
M_3 &= ((M_2 \sqcup M_8) \sqcap (j \geq 100))^* \\
M_4 &= (\mathit{Assign}(i \leftarrow i + 1)(M_3))^* \\
M_5 &= (M_4 \sqcap (j \leq i))^* \\
M_7 &= (\mathit{Assign}(i \leftarrow i - 1, j \leftarrow j - 2)(M_5))^* \\
M_8 &= ((M_4 \sqcap (j > i))^* \sqcup M_7 \\
M_9 &= ((M_2 \sqcup M_8) \sqcap (j < 100))^*
\end{aligned}
$$

With length one closure as initial policy, finds in 1 policy iteration the lfp at (9):

Policy iteration:

$$
\begin{cases}
150 \leq i \leq 174 \\
98 \leq j \leq 99 \\
-76 \leq j - i \leq -51
\end{cases}
$$

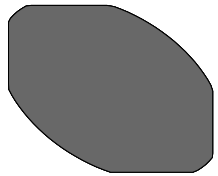
Kleene+widening/narrowing:

$$
\begin{cases}
150 \leq i \\
98 \leq j \leq 99 \\
j - i \leq -51
\end{cases}
$$

The case of (possibly non-linear) templates - currently submitted

Also include the general case of (linear) templates of Manna et al. (VMCAI'2005, 2006)

```
x:=[0,1]; v:=[0,1];
h:=0.01;
while(true){                    //2
w:=v;
v:=v*(1-h)-h*x;
x:=x+h*w;                       }
```



$\{-1.8708 \leq x \leq 1.8708, -1.5275 \leq v \leq 1.5275, 2x^2+3v^2+2xv \leq 7\}$

Policies are in infinite number and correspond to Lagrange multipliers of a relaxed functional...

Are we happy?

- of course not...
- other domains? Alias domains, affine forms domain etc.
- use for incremental verification
- but what is of interest to us here:
  - (1) Policy iteration as defined previously leads efficiently only to a fixed point (see CAV'2005 for a costly workaround for the lfp)
  - (2) Even with this costly workaround, it is only proved to find the lfp (and not just a postfixed point) for non-expansive maps - even though this works very well in practice

We give a method for (1) and hints for (2)
(see also the method of Seidl and Gawlitza, 2007)

Back to intervals: Example 2 ($x = 1+x \rightsquigarrow x = 1-x$)

```
int x=0                  //1
    while(x<=99){        //2
          x=1-x;         //3
    }                    //4
```

The fixed point equation becomes:

$$f\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0,(x_2^+ - 1)) \\ \min(99, \max(0,(x_2^- + 1)) \end{pmatrix}.$$

We start by

$$f^{\pi^0}\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0,(x_2^+ - 1)) \\ 99 \end{pmatrix},$$

PI algorithm returns $(98, 99)$, i.e $x_2 = [-98, 99]$,

Back to intervals: Example 2 ($x = 1+x \rightsquigarrow x = 1-x$)

```
int x=0                  //1
    while(x<=99){        //2
          x=1-x;         //3
    }                    //4
```

The fixed point equation becomes:

$$f\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0,(x_2^+ - 1)) \\ \min(99, \max(0,(x_2^- + 1)) \end{pmatrix}.$$

We start by

$$f^{\pi^0}\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0,(x_2^+ - 1)) \\ 99 \end{pmatrix},$$

PI algorithm returns $(98, 99)$, i.e $x_2 = [-98, 99]$, but $(0, 1)$ is the smallest fixed point i.e $x_2 = [0, 1]$.

Back to intervals: Example 2 ($x = 1+x \rightsquigarrow x = 1-x$)

```
int x=0                  //1
    while(x<=99){        //2
          x=1-x;         //3
    }                    //4
```

The fixed point equation becomes:

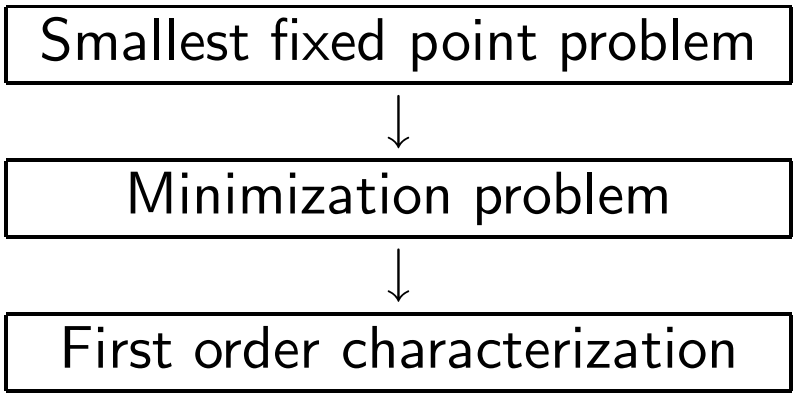
$$f\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0,(x_2^+ - 1)) \\ \min(99, \max(0,(x_2^- + 1)) \end{pmatrix}.$$

We start by

$$f^{\pi^0}\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0,(x_2^+ - 1)) \\ 99 \end{pmatrix},$$

PI algorithm returns $(98, 99)$, i.e $x_2 = [-98, 99]$, but $(0, 1)$ is the smallest fixed point i.e $x_2 = [0, 1]$. How can we refine PI to find the smallest fixed point?

Optimization point of view



Maps are not C^1 neither convex $\dashrightarrow$ weaker differential notion: semidifferential, which exists for our min-max maps.

Implementable characterization $\dashrightarrow$ nonlinear spectral radius.

Example of weak derivative

- Consider

$$f\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0,(x_2^+ - 1)) \\ \min(99,\max(0,(x_2^- + 1))) \end{pmatrix}.$$

- We want the semi-differential at $(98,99)$, we compute first:

$$f\begin{pmatrix} 98 \\ 99 \end{pmatrix} = \begin{pmatrix} max(0, x_2^+ - 1 = 98) = 98 \\ min(99, max(0, x_2^- + 1 = 99)) = 99 \end{pmatrix}$$

- We find

$$f'_{(98,99)}\begin{pmatrix} dx_2^- \\ dx_2^+ \end{pmatrix} = \begin{pmatrix} dx_2^+ \\ \min(0, dx_2^-) \end{pmatrix}$$

Then in a neighbourhood of $(98,99)$ we have

$$f\begin{pmatrix} 98+dx_2^- \\ 99+dx_2^+ \end{pmatrix} = f\begin{pmatrix} 98 \\ 99 \end{pmatrix} + f'_{(98,99)}\begin{pmatrix} dx_2^- \\ dx_2^+ \end{pmatrix}\left[+O\left(\begin{pmatrix} dx_2^- \\ dx_2^+ \end{pmatrix}^2\right)\right]$$

Smallest fixed point equivalent characterization (see MTNS'2008)

Theorem (Characterization of the smallest fixed point)

$f : \mathbb{R}^d \mapsto \mathbb{R}^d$ be an order preserving nonexpansive piecewise affine map. The following assertions are equivalent:

1. *u is the smallest fixed point of f.*
2. *$\mathbf{Fix}_{|\mathbb{R}^d_-}(f'_u) = \{0\}$ (fixpoints in the negative quadrant, of the weak derivative should only be 0)*
3. *$\rho_{\mathbb{R}^d_-}(f'_u) < 1$ (the generalized spectral radius of the weak derivative should be strictly less than 1)*

Compute the smallest fixed point by Policy Iteration

1. Compute a fixed point u_k of f by Policy Iteration.
2. *Policy Improvement* ($\mathbf{I}^{\mathbf{2}}_{\mathbf{k}}$) :Compute $\alpha_k := \rho_{\mathbb{R}^d_-}(f'_{u^k})$.
   - If $\alpha_k < 1$, returns u^k.
   - If $\alpha_k = 1$, take $h \in \mathbb{R}^d_- \setminus \{0\}$ s.t $f'_{u^k}(h) = h$.
     Take $\pi^{k+1}(j)$ which attains the min in
     $(f'_{u^k})_j(h) = \min_{a \in \bar{A}_j} (f'_a)_{u^k}(h)$ where $\bar{A}_j = \{a \mid f_a(u^k) = f(u^k)\}$.
     Initialize a new Policy Iteration with $f^{\pi^{k+1}}$.

This algorithm stops.

Computational details

To compute spectral radius...(Olsder (91), Gunawardena (94)):

A homogeneous min-max function of the variables $h_1, \ldots, h_d$ is a term in the grammar: $X \mapsto \min(X, X), \max(X, X), h_1, \cdots, h_d, 0$. E.g, $f(h_1, h_2, h_3) = \min(\max(0, h_2), h_3)$.

Proposition (Spectral radius of homogeneous min-max maps)

Let g be a homogeneous min-max map on $\mathbb{R}^d$, $e \equiv -1$, then

1. $\rho_{\mathbb{R}^d_-}(g) \in \{0, 1\}$.
2. $\rho_{\mathbb{R}^d_-}(g) = 0 \iff \lim_{k \to +\infty} g^k(e) = 0$.
3. *This latter limit is reached in at most d steps.*

Come back to Example 2

- $$f\begin{pmatrix} x_2^- \\ x_2^+ \end{pmatrix} = \begin{pmatrix} \max(0,(x_2^+ - 1)) \\ \min(99, \max(0,(x_2^- + 1))) \end{pmatrix}.$$
- We found the semidifferential at $(98, 99)$:

$$f'_{(98,99)}\begin{pmatrix} dx_2^- \\ dx_2^+ \end{pmatrix} = \begin{pmatrix} dx_2^+ \\ \min(0, dx_2^-) \end{pmatrix}$$

- We iterate $f'_{(98,99)}$ on $e = (-1, -1)$:

$$f'_{(98,99)}\begin{pmatrix} -1 \\ -1 \end{pmatrix} = \begin{pmatrix} -1 \\ \min(0,-1) = -1 \end{pmatrix}!!$$

- Hence $\rho_{\mathbb{R}^2_-}(f'_{(98,99)}) = 1$ and $f'_{(98,99)}(e) = e$.
- This fixed point leads to a new policy:
  $f^{\pi^1}(x_2^-, x_2^+) := (\max(0,(x_2^+ - 1)), \max(0,(x_2^- + 1)))$
- (Policy Iteration) returns $(0, 1)$

Policy improvement

- Now at $(0,1)$ we find:

$$f'_{(0,1)} \begin{pmatrix} h_1 \\ h_2 \end{pmatrix} = \begin{pmatrix} \max(0, h_2) \\ h_1 \end{pmatrix}$$

- We compute successively (power algorithm):

$$f'_{(0,1)} \begin{pmatrix} -1 \\ -1 \end{pmatrix} = \begin{pmatrix} 0 \\ -1 \end{pmatrix}$$

$$f'^2_{(0,1)} \begin{pmatrix} -1 \\ -1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

- Hence $\rho_{\mathbb{R}^2_-}(f'_{(0,1)}) = 0$ so $(0,1)$ is the smallest fixed point.

Conclusion

Conclusion

- Computional method to compute the smallest fixed point of monotone piecewise affine nonexpansive maps .
- Prototype implementation in C.
- Fast method.

Future Work

- Big “jumps“ problem (the map is no longer nonexpansive).
- Improvement of calculus of the spectral radius and the negative fixed point.
- Extension of weak-differential method to zones, templates, non-linear templates...