Overview

- Motivation
- Interval arithmetic;
- The notions of approximation and narrowing;
- Narrowing-based local consistency
- Interval solvers

Interval Constraints

Frédéric Benhamou and William J. Older.
Applying Interval Arithmetic to Real, Integer and Boolean Constraints
Journal of Logic Programming 32(1), 1-24,1997

Real Intervals

$$\mathcal{E} = \mathcal{R} \cup \{-\infty, +\infty\}$$

- (Closed) Interval: $[a, b] = \{x \in \mathcal{R} | a \leq x \leq b\}$ where $a, b \in \mathcal{E}$
- Similar definition for open/mixed intervals.

F-intervals:
Choose instead a finite subset F of $\mathcal{E}$, **including** $-\infty, +\infty$! e.g. floating-point numbers

F-boxes:
Cartesian products of F-intervals; inclusion ordering

Motivation

- Nonlinear constraints on real
- Integrating Real, Integer and Boolean constraints

Properties of Approximations

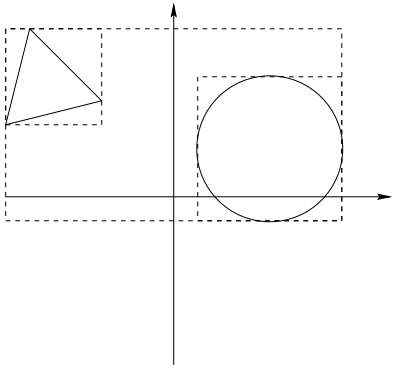
Approximation

- can be done separately on projections.
  $apx(\rho) = (apx(\pi_1(\rho)), ..., apx(\pi_n(\rho)))$
- is monotonic
  $(\rho \subseteq \rho') \Rightarrow (apx(\rho) \subseteq apx(\rho'))$
- is idempotent
  $apx(apx(\rho)) = apx(\rho)$
- $apx(\rho \cup \rho') = apx(apx(\rho) \cup apx(\rho'))$
- $apx(\rho \cap \rho') \subseteq apx(\rho) \cap apx(\rho')$

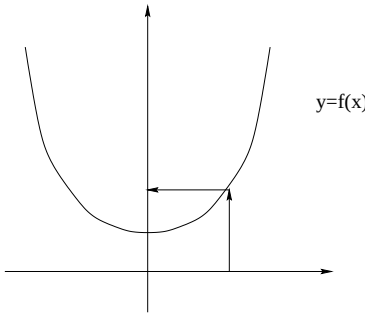
Approximations

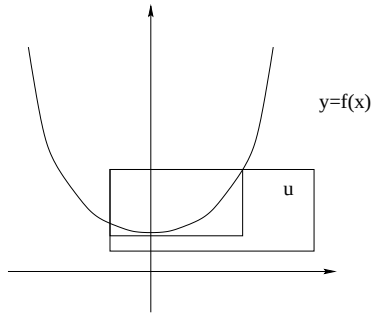
The *approximation* $apx(\rho)$ of $\rho \subseteq \mathcal{R}^n$
is the least F-box including ρ

Notation: For $r \in \mathcal{R}$ $apx(r)$ denoted $\vec{r}$.



Narrowing





Narrowing

A basis for interval computations.

Given:

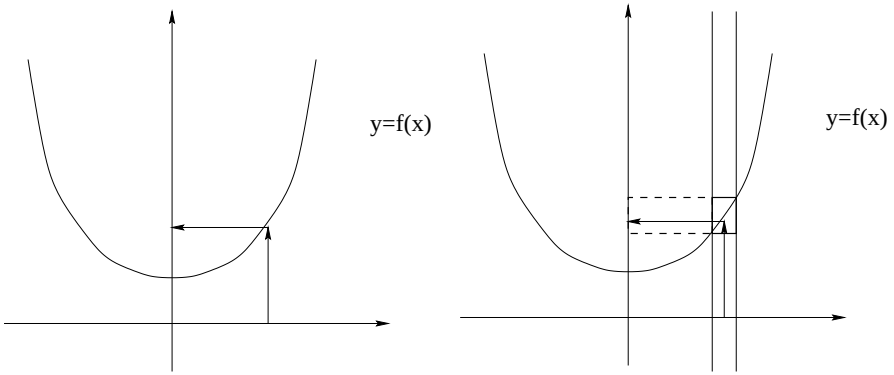
- ρ a relation on $\mathcal{R}$
- an F-box u

find the least F-box u' s.that
u and u' include the same elements of ρ.

$$\vec{\rho}(u) = apx(\rho \cap u)$$

Interval computations

Using narrowing for computing approximated values:



Properties of Narrowing

- Contractance $\vec{\rho}(u) \subseteq u$
- Correctness $u \cap \rho = \vec{\rho}(u) \cap \rho$
- Monotonicity $u \subseteq v \Rightarrow \vec{\rho}(u) \subseteq \vec{\rho}(v)$
- Idempotence $\vec{\rho}(\vec{\rho}(u)) = \vec{\rho}(u)$

Interval Extensions of Arithmetic Operations

$$< a, b > * < c, d >= \{< x * y > | x \in < a, b >, y \in < c, d >\}$$

Thus:

- $[a, b] + [c, d] = [a + c, , b + d]$
- $[a, b] - [c, d] = [a - d, b - c]$
- $[a, b] \times [c, d] = [min(ac, ad, bc, bd), max(ad, ad, bc, bd)]$
- $[a, b]/[c, d] = [a, b] \times [1/c, 1/d],\ if\ 0 \notin [c, d]$

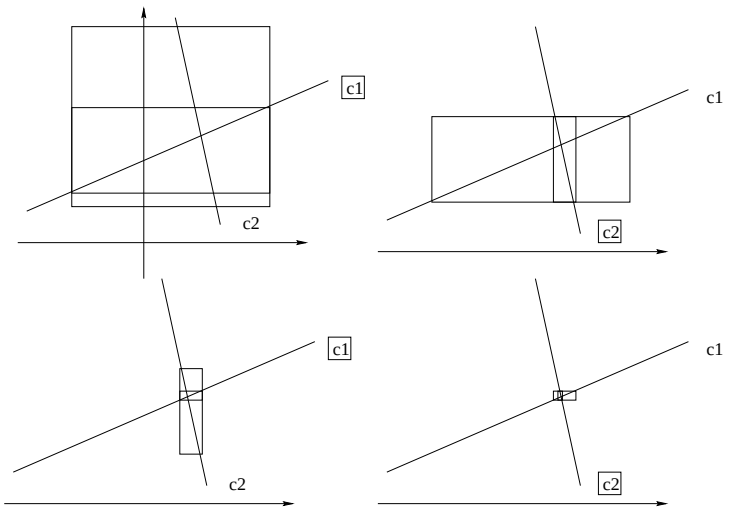
Narrowing in Practice

How to compute $\rho \cap u$?

- An n-ary interval function F is an interval extension of f on $\mathcal{R}^n$ iff
$$f(a_1, ..., a_n) \in F(\vec{a_1}, ..., \vec{a_n})$$
- An n-ary interval relation R is an interval extension of $r \subseteq \mathcal{R}^n$ iff
$$(a_1, ..., a_n) \in r \Rightarrow (\vec{a_1}, ..., \vec{a_n}) \in R$$

Notation: $\vec{r}$ denotes an interval extension of r.

Narrowing constraint systems



Example Relation

$p(x, y, z)$ a polynomial over R;

$$p(x, y, z) = 0$$

is a ternary relation ρ over R;

P: interval extension of p (in the interval arithmetic).

Interval extension $\vec{\rho}$.

$$\vec{\rho}(X, Y, Z) = \{< X, Y, Z > | P(X, Y, Z) \cap [0, 0] \neq \emptyset\}$$

Approximation Revisited

We will call apx **Hull approximation** (hap).

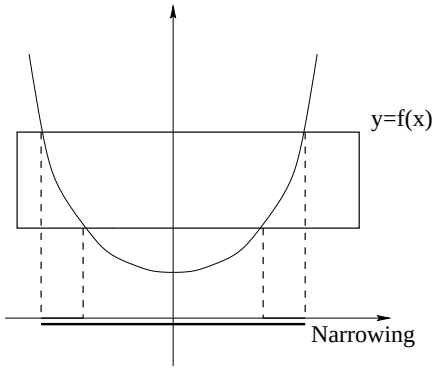
A more precise way : **Finite Unions of F-boxes** (uap).

- Preserves information coming from non-convex relations.
- Computationally very expensive

Interval Convexity

A relation ρ is **interval-convex** iff
for every box u $\pi_i(\rho \cap u)$ is an interval.

Counterexample:



Arc-consistency

C arc consistent wrt $(D_1, ..., D_n)$ iff for each $i = 1, ..., n$:

$$D_i = D_i \cap \{r_i | (\exists r_1 \in D_1, ...r_{i-1} \in D_{i-1}, \\ r_{i+1} \in D_{i+1}, ..., r_n \in D_n) : \overline{r} \in C\}$$

D_i a set of reals: in practice has to be approximated

Relation to clp(FD)

- Variable domain: a finite number of F-intervals
- Domain reduction by narrowing
- Hull-based narrowing similar to bound consistency
- Union-of-interval mode similar to hyper-arc consistency
- Backtracking: a systematic choice of boxes
- Generic: narrowing technique applies to any function/relation
- Efficiency may be a problem.

Hull-consistency

Restricting domains to F-intervals

C hull-consistent wrt $(I_1, ..., I_n)$ iff for each $i = 1, ..., n$:

$$I_i = hap(I_i \cap \{r_i | (\exists r_1 \in I_1, ...r_{i-1} \in I_{i-1}, \\ r_{i+1} \in I_{i+1}, ..., r_n \in I_n) : \overline{r} \in C\})$$

$x = y^2$ is interval consistent wrt $[-1, 1], [0.25, 1]$

How to check this in practice?

Interval-consistency

Restricting domains to finite unions of F-intervals

C interval-consistent wrt $(U_1, ..., U_n)$ iff for each $i = 1, ..., n$:

$$U_i = uap(U_i \cap \{r_i | (\exists r_1 \in U_1, ...r_{i-1} \in U_{i-1}, \\ r_{i+1} \in U_{i+1}, ..., r_n \in U_n) : \overline{r} \in C\})$$

$x = y^2$ is interval-consistent wrt $\{[-1, -0.5], [0.5, 1]\}, \{[0.25, 1]\}$

How to check this in practice?

Hull-consistency vs. Box-consistency

Check $x_1 - x_1 + x_2 = 0$ wrt $[-1,1][0,2]$

Hull- consistency:
$X_2 = \{x_2 | \exists x_1 \in [-1,1] : x_1 - x_1 + x_2 = 0\} = \{0\}$
$\{0\} \cap [0,2] \neq [0,2]$ hence C **not** hull-consistent.

Box-consistency: **yes**
$\vec{C}(X,Y) = \{< X,Y > | (X - X + Y) \cap [0,0] \neq \emptyset\}$
Checking bounds:
$P = left([-1,1] - left([-1,1]) + [0,2] = [0,0] + [0,2]$:
$P \cap [0,0] \neq \emptyset$ hence $< [left([-1,1]), [0,2] > \in \vec{C}$
....

Box-consistency

Use hull-approximation of the relation C:

C box-consistent wrt $(I_1, ..., I_n)$ iff for each $i = 1, ..., n$:

$I_i = hap(I_i \cap \{r_i | (I_1, ..., I_{i-1}, \vec{r_i}, I_{i+1}, ..., I_n) \in \vec{C}\}$ (*)

If (*) doesn't hold then:

$< I_1, ..., left(I_i), ...I_n >$ or $< I_1, ..., right(I_i), ...I_n >$ is not in$\vec{C}$

Restricting Intervals

Integer intervals: If $u = [a,b]$ then $\vec{int}(u) = [\lceil a \rceil, \lfloor b \rfloor]$

```
>>   U ~ cc(-0.3,2.4) n int,  V ~ cc(1.5,2.5) n int.

V = 2,
U ~ cc(0,2).

>>  U ~ cc(-0.3,2.4) n int,  V ~ cc(1.5,2.5) n int,
    dif(U,V).

V = 2,
U ~ cc(0,1).
```


Combining Booleans, Integers and Reals

```
>> B = ble(X,0).

   X ~ real,
   B ~ cc(0,1).
>> B = ble(X,0), B=1, X ~ cc(-5.2, 3) n int.

   B = 1,
   X ~ cc(-5,0).
>> Z = ble(X,0).+.4.

   X ~ real,
   Z ~ cc(4,5).
```

Prolog IV

```
>> square(X) ~ 4.

X ~ cc(-2,2).

>> set_prolog_flag(interval_mode, union).

true.

>> square(X) ~ 4.

X ~ -2 u 2.
```


Prolog IV

```
>> Y~X, Y ~ square(X).+.2.*.X.+.1.

Y = X,
X ~ real.

>> Y~cc(0,10), X~cc(0,10),  Y~X, Y ~ square(X).+.2.*.X.+.1.

false.
>> Y~cc(0,10), X~cc(0,10),  Y~X, Y ~ square(X).-.X.+.1.

Y = X,
X ~ cc(0,‘>1.0004882‘).
```


Prolog IV

```
>> X ~ cc(1,10),X~int, Y ~ cc(1,10) n int,
   Z ~cc(1,100) n int,
   square(Z) ~ square(X).+.square(Y),
   intsplit([X,Y,Z]), gt(X,Y).

   Z = 5,
   Y = 3,
   X = 4;

   Z = 10,
   Y = 6,
   X = 8.
```


Prolog IV

```
>> X ~ cc(1,10),X~int, Y ~ cc(1,10) n int,
   Z ~cc(1,10) n int,
   square(Z) ~ square(X).+.square(Y).

   Z ~ 5 u 10,
   Y ~ 3 u 4 u 6 u 8,
   X ~ 3 u 4 u 6 u 8.
```