Two Type-Theoretic Approaches to Probabilistic Termination

Ugo dal Lago   Charles Grellois

FOCUS Team – INRIA & University of Bologna

DICE-FOPARA 2017

Motivations

- Probabilistic programming languages are more and more pervasive in computer science: modeling uncertainty, robotics, cryptography, machine learning, AI...
- Quantitative notion of termination: almost-sure termination (AST)
- AST has been studied for imperative programs in the last years...
- ...but what about the probabilistic functional languages?

We introduce a monadic, affine sized type system sound for AST (our result at ESOP 2017), and sketch a dependent, affine type system for AST (work in progress).

Sized Types: the Deterministic Case

Simply-typed λ-calculus is strongly normalizing (SN).

No longer true with the letrec construction...

Sized types: a decidable extension of the simple type system ensuring SN for λ-terms with letrec.

See notably:

- Hughes-Pareto-Sabry 1996, *Proving the correctness of reactive systems using sized types*,
- Barthe-Frade-Giménez-Pinto-Uustalu 2004, *Type-based termination of recursive definitions*.

Sized Types: the Deterministic Case

Sizes: $\mathfrak{s}, \mathfrak{r} \quad ::= \quad \mathfrak{i} \;\mid\; \infty \;\mid\; \widehat{\mathfrak{s}}$

$+$ size comparison underlying subtyping. Notably $\widehat{\infty} \equiv \infty$.

Idea: k successors = at most k constructors.

- $\mathsf{Nat}^{\widehat{\mathfrak{i}}}$ is 0,
- $\mathsf{Nat}^{\widehat{\widehat{\mathfrak{i}}}}$ is 0 or S 0,
- ...
- Nat^{∞} is any natural number. Often denoted simply Nat.

The same for lists,. . .

Sized Types: the Deterministic Case

Sizes: $\mathfrak{s}, \mathfrak{r} \quad ::= \quad \mathfrak{i} \;\Big|\; \infty \;\Big|\; \widehat{\mathfrak{s}}$

$+$ size comparison underlying subtyping. Notably $\widehat{\infty} \equiv \infty$.

Fixpoint rule:

$$\frac{\Gamma, f \,:\, \mathsf{Nat}^{\mathfrak{i}} \to \sigma \vdash M \,:\, \mathsf{Nat}^{\widehat{\mathfrak{i}}} \to \sigma[\mathfrak{i}/\widehat{\mathfrak{i}}] \qquad \mathfrak{i} \text{ pos } \sigma}{\Gamma \vdash \mathsf{letrec}\ f \ = \ M \,:\, \mathsf{Nat}^{\mathfrak{s}} \to \sigma[\mathfrak{i}/\mathfrak{s}]}$$

"To define the action of f on size $n+1$,
we only call recursively f on size at most n"

Sized Types: the Deterministic Case

Sizes: $\quad \mathfrak{s}, \mathfrak{r} \quad ::= \quad \mathfrak{i} \mid \infty \mid \widehat{\mathfrak{s}}$

+ size comparison underlying subtyping. Notably $\widehat{\infty} \equiv \infty$.

Fixpoint rule:

$$\frac{\Gamma, f : \mathsf{Nat}^{\mathfrak{i}} \to \sigma \vdash M : \mathsf{Nat}^{\widehat{\mathfrak{i}}} \to \sigma[\mathfrak{i}/\widehat{\mathfrak{i}}] \qquad \mathfrak{i} \text{ pos } \sigma}{\Gamma \vdash \mathsf{letrec}\ f = M : \mathsf{Nat}^{\mathfrak{s}} \to \sigma[\mathfrak{i}/\mathfrak{s}]}$$

Typable $\implies$ SN. Proof using reducibility candidates.

Decidable type inference.

A Probabilistic λ-calculus

$$M, N, \ldots \quad ::= \quad V \mid V\ V \mid \mathsf{let}\ x = M\ \mathsf{in}\ N \mid M \oplus_p N \\ \mid \mathsf{case}\ V\ \mathsf{of}\ \{\, \mathsf{S} \to W \mid 0 \to Z \,\}$$

$$V, W, Z, \ldots \quad ::= \quad x \mid 0 \mid \mathsf{S}\ V \mid \lambda x.M \mid \mathsf{letrec}\ f = V$$

- Formulation equivalent to λ-calculus with $\oplus_p$, but constrained for technical reasons (A-normal form)
- Restriction to base type Nat for simplicity, but can be extended to general inductive datatypes (as in sized types)

A Probabilistic λ-calculus: Operational Semantics

$$\frac{}{\text{let } x \; = \; V \text{ in } M \;\; \to_v \;\; \left\{ (M[x/V])^1 \right\}}$$

$$\frac{}{(\lambda x.M) \;\; V \;\; \to_v \;\; \left\{ (M[x/V])^1 \right\}}$$

$$\frac{}{(\text{letrec } f \; = \; V) \; \left(c \; \overrightarrow{W}\right) \;\; \to_v \;\; \left\{ \left(V[f/\,(\text{letrec } f \; = \; V)] \; \left(c \; \overrightarrow{W}\right)\right)^1 \right\}}$$

A Probabilistic λ-calculus: Operational Semantics

$$\frac{}{\mathsf{case}\ \mathsf{S}\ V\ \mathsf{of}\ \{\,\mathsf{S} \rightarrow W \ \mid\ 0 \rightarrow Z\,\} \ \ \rightarrow_v\ \ \left\{ (W\ \ V)^1 \right\}}$$

$$\frac{}{\mathsf{case}\ 0\ \mathsf{of}\ \{\,\mathsf{S} \rightarrow W \ \mid\ 0 \rightarrow Z\,\} \ \ \rightarrow_v\ \ \left\{ (Z)^1 \right\}}$$

A Probabilistic λ-calculus: Operational Semantics

$$\frac{}{M \;\oplus_p\; N \;\;\to_v\;\; \left\{\, M^p,\, N^{1-p} \,\right\}}$$

$$\frac{M \;\;\to_v\;\; \left\{\, L_i^{p_i} \;\middle|\; i \in I \,\right\}}{\mathsf{let}\ x \;=\; M\ \mathsf{in}\ N \;\;\to_v\;\; \left\{\, (\mathsf{let}\ x \;=\; L_i\ \mathsf{in}\ N)^{p_i} \;\middle|\; i \in I \,\right\}}$$

A Probabilistic λ-calculus: Operational Semantics

$$\frac{\mathscr{D} \stackrel{VD}{=} \left\{ M_j^{p_j} \;\middle|\; j \in J \right\} + \mathscr{D}_V \qquad \forall j \in J, \quad M_j \;\rightarrow_v\; \mathscr{E}_j}{\mathscr{D} \;\rightarrow_v\; \left(\sum_{j \in J} p_j \cdot \mathscr{E}_j \right) + \mathscr{D}_V}$$

For $\mathscr{D}$ a distribution of terms:

$$[\![\, \mathscr{D} \,]\!] \;=\; \sup_{n \in \mathbb{N}} \left(\left\{ \mathscr{E}_n \;\middle|\; \mathscr{D} \Rrightarrow_v^n \mathscr{E}_n \right\} \right)$$

where $\Rrightarrow_v^n$ is $\rightarrow_v^n$ followed by projection on values.

We let $[\![\, M \,]\!] \;=\; [\![\, \{ M^1 \} \,]\!]$.

M is AST iff $\sum [\![\, M \,]\!] = 1$.

Random Walks as Probabilistic Terms

- Biased random walk:

$$M_{bias} \;=\; \Big(\text{letrec } f \;=\; \lambda x.\text{case } x \text{ of } \Big\{ \mathsf{S} \to \lambda y.f(y) \oplus_{\frac{2}{3}} (f(\mathsf{S}\,\mathsf{S}\,y))) \;\Big|\; 0 \to 0 \Big\}\Big) \;\underline{n}$$

- Unbiased random walk:

$$M_{unb} \;=\; \Big(\text{letrec } f \;=\; \lambda x.\text{case } x \text{ of } \Big\{ \mathsf{S} \to \lambda y.f(y) \oplus_{\frac{1}{2}} (f(\mathsf{S}\,\mathsf{S}\,y))) \;\Big|\; 0 \to 0 \Big\}\Big) \;\underline{n}$$

$$\sum [\![M_{bias}]\!] \;=\; \sum [\![M_{unb}]\!] \;=\; 1$$

Capture this in a sized type system?

Another Term

We also want to capture terms as:

$$M_{nat} \;=\; \left(\text{letrec } f \;=\; \lambda x.x \oplus_{\frac{1}{2}} \mathsf{S}\;(f\;x)\right)\;0$$

of semantics

$$[\![\, M_{nat} \,]\!] \;=\; \left\{(0)^{\frac{1}{2}}, (\mathsf{S}\;0)^{\frac{1}{4}}, (\mathsf{S}\;\mathsf{S}\;0)^{\frac{1}{8}}, \ldots\right\}$$

summing to 1.

(This is the geometric distribution.)

Beyond SN Terms, Towards Distribution Types

First idea: extend the sized type system with:

$$\text{Choice} \qquad \frac{\Gamma \vdash M : \sigma \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash M \oplus_p N : \sigma}$$

and "unify" types of M and N by subtyping.

Kind of product interpretation of $\oplus$: we can't capture more than SN...

Beyond SN Terms, Towards Distribution Types

First idea: extend the sized type system with:

$$\text{Choice} \quad \frac{\Gamma \vdash M : \sigma \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash M \oplus_p N : \sigma}$$

and "unify" types of M and N by subtyping.

We get at best

$$f \ : \ \mathsf{Nat}^{\widehat{\widehat{\mathfrak{i}}}} \to \mathsf{Nat}^{\infty} \ \vdash \ \lambda y. f(y) \oplus_{\frac{1}{2}} (f(\mathsf{S}\,\mathsf{S}\,y))) \ : \ \mathsf{Nat}^{\widehat{\mathfrak{i}}} \to \mathsf{Nat}^{\infty}$$

and can't use a variation of the letrec rule on that.

Beyond SN Terms, Towards Distribution Types

We will use distribution types, built as follows:

$$\text{Choice} \quad \frac{\Gamma \,|\, \Theta \;\vdash\; M : \mu \qquad \Gamma \,|\, \Psi \;\vdash\; N : \nu \qquad \{\!|\, \mu \,|\!\} = \{\!|\, \nu \,|\!\}}{\Gamma \,|\, \Theta \oplus_p \Psi \;\vdash\; M \oplus_p N : \mu \oplus_p \nu}$$

Now

$$f \;:\; \left\{ \left(\mathrm{Nat}^{\mathfrak{i}} \to \mathrm{Nat}^{\infty}\right)^{\frac{1}{2}},\; \left(\mathrm{Nat}^{\widehat{\widehat{\mathfrak{i}}}} \to \mathrm{Nat}^{\infty}\right)^{\frac{1}{2}} \right\}$$

$$\vdash$$

$$\lambda y.f(y) \oplus_{\frac{1}{2}} (f(\mathsf{S}\,\mathsf{S}\,y))) \;:\; \mathrm{Nat}^{\widehat{\mathfrak{i}}} \to \mathrm{Nat}^{\infty}$$

Designing the Fixpoint Rule

$$f \ : \ \left\{ \left(\mathsf{Nat}^{\mathfrak{i}} \to \mathsf{Nat}^{\infty}\right)^{\frac{1}{2}},\ \left(\mathsf{Nat}^{\widehat{\widehat{\mathfrak{i}}}} \to \mathsf{Nat}^{\infty}\right)^{\frac{1}{2}} \right\}$$

$$\vdash$$

$$\lambda y.f(y) \oplus_{\frac{1}{2}} (f(\mathsf{S}\,\mathsf{S}\,y))) \quad : \quad \mathsf{Nat}^{\widehat{\mathfrak{i}}} \to \mathsf{Nat}^{\infty}$$

induces a random walk on $\mathbb{N}$:

- on $n+1$, move to n with probability $\frac{1}{2}$, on $n+2$ with probability $\frac{1}{2}$,
- on 0, loop.

The type system ensures that there is no recursive call from size 0.

Random walk AST (= reaches 0 with proba 1) $\Rightarrow$ termination.

Designing the Fixpoint Rule

$$\text{LetRec}\quad \frac{\begin{array}{c}\{\!|\,\Gamma\,|\!\} \;=\; \mathsf{Nat} \\ \mathfrak{i} \notin \Gamma \text{ and } \mathfrak{i} \text{ positive in } \nu \\ \color{red}{\left\{\, (\mathsf{Nat}^{\mathfrak{s}_j} \to \nu[\mathfrak{i}/\mathfrak{s}_j])^{p_j} \;\middle|\; j \in J \,\right\} \text{ induces an AST sized walk}} \\ \Gamma \,|\, f \,:\, \left\{\, (\mathsf{Nat}^{\mathfrak{s}_j} \to \nu[\mathfrak{i}/\mathfrak{s}_j])^{p_j} \;\middle|\; j \in J \,\right\} \vdash V \,:\, \mathsf{Nat}^{\widehat{\mathfrak{i}}} \to \nu[\mathfrak{i}/\widehat{\mathfrak{i}}]\end{array}}{\Gamma \,|\, \emptyset \vdash \mathsf{letrec}\ f \;=\; V \,:\, \mathsf{Nat}^{\mathfrak{r}} \to \nu[\mathfrak{i}/\mathfrak{r}]}$$

Sized walk: AST is checked by an external PTIME procedure.

Generalized Random Walks and the Necessity of Affinity

A crucial feature: our type system is affine.

Higher-order symbols occur at most once. Consider:

$$M_{naff} \quad = \quad \text{letrec } f \; = \; \lambda x.\text{case } x \text{ of } \left\{ \mathsf{S} \to \lambda y.f(y) \oplus_{\frac{2}{3}} (f(\mathsf{S}\,\mathsf{S}\,y)\,;\, f(\mathsf{S}\,\mathsf{S}\,y)) \;\middle|\; 0 \to 0 \right\}$$

The induced sized walk is AST, but M_{naff} is not.

Key Property I: Subject Reduction

Main idea: reduction of

$$\emptyset \mid \emptyset \vdash 0 \oplus 0 \; : \; \left\{ \left(\mathsf{Nat}^{\widehat{\mathfrak{s}}}\right)^{\frac{1}{2}}, \left(\mathsf{Nat}^{\widehat{\widehat{\mathfrak{r}}}}\right)^{\frac{1}{2}} \right\}$$

is to

$$\left\{ \left(0 \; : \; \mathsf{Nat}^{\widehat{\mathfrak{s}}}\right)^{\frac{1}{2}}, \left(0 \; : \; \mathsf{Nat}^{\widehat{\widehat{\mathfrak{r}}}}\right)^{\frac{1}{2}} \right\}$$

1. Same expectation type: $\frac{1}{2} \cdot \mathsf{Nat}^{\widehat{\mathfrak{s}}} + \frac{1}{2} \cdot \mathsf{Nat}^{\widehat{\widehat{\mathfrak{r}}}}$
2. Splitting of $[\![\, 0 \oplus 0 \,]\!]$ in a typed representation $\rightarrow$ notion of pseudo-representation

Key Property I: Subject Reduction

Theorem

Let $M \in \Lambda_\oplus$ be such that $\emptyset \mid \emptyset \vdash M : \mu$. Then there exists a closed typed distribution $\left\{ (W_j : \sigma_j)^{p'_j} \;\middle|\; j \in J \right\}$ such that

- $\mathbb{E}\left((W_j : \sigma_j)^{p'_j} \right) \preccurlyeq \mu$,
- *and that $\left[(W_j)^{p'_j} \;\middle|\; j \in J \right]$ is a pseudo-representation of $[\![M]\!]$.*

By the soundness theorem of next slide, this inequality is in fact an equality.

Key Property II: Typing Soundness

Theorem (Typing soundness)

If $\Gamma \mid \Theta \vdash M : \mu$, then M is AST.

Proof by reducibility, using set of candidates parametrized by probabilities.

Reducibility, the Probabilistic Case

Usual reducibility proof:

$$M \text{ closed of type } \sigma \quad \Rightarrow \quad M \in Red_\sigma \quad \Rightarrow \quad M \text{ is SN}$$

In our setting:

Reducibility, the Probabilistic Case

Usual reducibility proof:

M closed of type $\sigma \quad \Rightarrow \quad M \in Red_\sigma \quad \Rightarrow \quad M$ is SN

In our setting:

$$M \in TRed_\sigma^p \quad \Rightarrow \qquad \sum [\![M]\!] \geq p$$

Reducibility, the Probabilistic Case

Usual reducibility proof:

$$M \text{ closed of type } \sigma \quad \Rightarrow \quad M \in Red_\sigma \quad \Rightarrow \quad M \text{ is SN}$$

In our setting:

$$M \text{ closed of type } \sigma \quad \Rightarrow \quad \forall p < 1,\ M \in TRed_\sigma^p \quad \Rightarrow \quad \forall p < 1,\ \sum [\![M]\!] \geq p$$

p increases with the number of fixpoint unfoldings we do, and we prove that M is in $TRed_\sigma^p$ iff its n-unfolding is.

Reducibility, the Probabilistic Case

Usual reducibility proof:

M closed of type $\sigma \quad \Rightarrow \quad M \in Red_\sigma \quad \Rightarrow \quad M$ is SN

In our setting:

M closed of type $\sigma \quad \Rightarrow \quad M \in TRed_\sigma^1 \quad \Rightarrow \quad \sum [\![M]\!] = 1$ i.e. M AST

by a continuity lemma.

Reducibility, the Probabilistic Case – Open Terms

Usual case: $\overrightarrow{x} : \overrightarrow{\sigma} \vdash M : \tau \quad \Rightarrow \quad \forall \overrightarrow{V} \in \overrightarrow{VRed_\sigma}, \;\; M[\overrightarrow{x}/\overrightarrow{V}] \in Red_\tau$

Reducibility, the Probabilistic Case – Open Terms

Usual case: $\overrightarrow{x} : \overrightarrow{\sigma} \vdash M : \tau \quad \Rightarrow \quad \forall \overrightarrow{V} \in \overrightarrow{VRed_{\sigma}}, \;\; M[\overrightarrow{x} / \overrightarrow{V}] \in Red_{\tau}$

In our setting: if $\Gamma \mid y : \{\tau_j^{p_j}\}_{j \in J} \vdash M : \mu$ then

- $\forall (q_i)_i \in [0,1]^n, \;\; \forall \overrightarrow{V} \in \prod_{i=1}^{n} \; \mathsf{VRed}_{\sigma_i}^{q_i},$
- $\forall \left(q_j'\right)_j \in [0,1]^J, \;\; \forall W \in \bigcap_{j \in J} \; \mathsf{VRed}_{\tau_j}^{q_j'},$
- we have $M[\overrightarrow{x}, y / \overrightarrow{V}, W] \in \mathsf{TRed}_{\mu}^{\alpha}$

where $\alpha \;\; = \;\; \left(\prod_{i=1}^{n} q_i\right) \left(\left(\sum_{j \in J} p_j q_j' \right) + 1 - \left(\sum_{j \in J} p_j \right) \right).$

Another Approach Using Dependent Types

Alternative approach to sized types: dependent types.

See Xi (2002), *Dependent Types for Program Termination Verification*.

Examples of dependent types à la Xi:

- $\varphi \;\middle|\; \Gamma \vdash \underline{2} : int\,(2)$
- $\varphi \;\middle|\; \Gamma \vdash \left\langle 2 \;\middle|\; \underline{2} \right\rangle : \Sigma a : int.\, int\,(a)$

Terms of base type: annotated with size information which can be packed in the term (annotation by a size expression). Produces a sum type (existential).

φ: context of constraints on free size variables, like $a \in \{a \in int \;\mid\; a > 2\}$.

Another Approach Using Dependent Types

Alternative approach to sized types: dependent types.

See Xi (2002), *Dependent Types for Program Termination Verification*.

Examples of dependent types à la Xi:

- $\varphi \mid \Gamma \vdash + \; : \; \Pi\{a : \mathit{int},\, b : \mathit{int}\}.\, \mathit{int}(a) \times \mathit{int}(b) \rightarrow \mathit{int}(a + b)$
- $\varphi \mid \Gamma \vdash \times \; : \; \Pi\{a : \mathit{int},\, b : \mathit{int}\}.\, \mathit{int}(a) \times \mathit{int}(b) \rightarrow \mathit{int}(a \times b)$

Functions typically have universally quantified arguments (product type). Note that we could derive terms from $+$ and $\times$ which use sum types for return types.

Another Approach Using Dependent Types

Sum types allow to get a uniform Choice rule:

$$\frac{\varphi \,|\, \Gamma \,|\, \Theta \vdash M \,:\, \sigma \qquad \varphi \,|\, \Gamma \,|\, \Theta \vdash N \,:\, \sigma}{\varphi \,|\, \Gamma \,|\, \Theta \vdash M \oplus_p N \,:\, \sigma}$$

No longer need for distribution types!

Various sizes are annotated in the term.

Another Approach Using Dependent Types

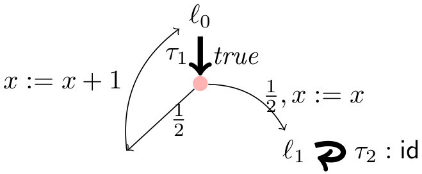
$$\dfrac{\{\!\mid \Gamma \mid\!\} \subseteq \{\mathbf{bool}, \mathbf{int}\} \qquad \varphi, \overrightarrow{a} : \overrightarrow{\gamma} \mid \Gamma \mid f : \prod \overrightarrow{a} : \overrightarrow{\gamma}.\sigma \Vdash_{\mathcal{P}} V : \sigma \qquad \mathsf{letrec}\,(\mathcal{P}, \rho) \text{ is AST for every } \rho \models \varphi}{\varphi \mid \Gamma \mid \Theta \vdash \mathsf{letrec}\ f[\overrightarrow{a} : \overrightarrow{\gamma}] : \sigma \ = \ V : \prod \overrightarrow{a} : \overrightarrow{\gamma}.\sigma}$$

- Rely on PTS analysis (more general than random walks)
- letrec enters a new mode: typing relation $\Vdash_{\mathcal{P}}$ indexed by a PTS

PTS = probabilistic transition system

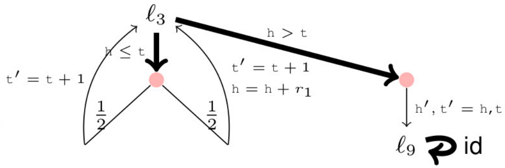
Examples of PTS

```
int x := 0;
while (flip (0.5))
    x ++;
// end
```



Examples from Chakarov and Sankaranarayanan (2013), *Probabilistic Program Analysis with Martingales*

Examples of PTS



Examples from Chakarov and Sankaranarayanan (2013), *Probabilistic Program Analysis with Martingales*

Our point: replaced sized walks by these processes modeling the flow of recursive calls. The process is built on-the-fly by the type system.

Building the PTS

$$\frac{\varphi \vdash \overrightarrow{I} : \overrightarrow{\gamma} \qquad \mathcal{P} = \textbf{leaf}\left(f\left[\overrightarrow{a} \mapsto \overrightarrow{I}\right]\right)}{\varphi \mid \Gamma \mid f : \prod \overrightarrow{a} : \overrightarrow{\gamma}.\sigma \Vdash_{\mathcal{P}} f\left[\overrightarrow{I}\right] : \sigma[\overrightarrow{a}/\overrightarrow{I}]}$$

$\textbf{leaf}\left(f\left[\overrightarrow{a} \mapsto \overrightarrow{I}\right]\right)$ is a PTS with just one node, looping on itself and updating $\overrightarrow{a}$ with $\overrightarrow{[\![I]\!]_\rho}$.

Building the PTS

$$\frac{\varphi \mid \Gamma \mid \emptyset \vdash M : \mathbf{bool}(I) \qquad \varphi, I = 1 \mid \Delta \mid \Theta \Vdash_{\mathcal{P}} N : \sigma \qquad \varphi, I = 0 \mid \Delta \mid \Theta \Vdash_{\mathcal{Q}} L : \sigma}{\varphi \mid \Gamma, \Delta \mid \Theta \Vdash_{\mathrm{if}(I,\mathcal{P},\mathcal{Q})} \text{if } M \text{ then } N \text{ else } L : \sigma}$$

if $(I, \mathcal{P}, \mathcal{Q})$ is a PTS containing $\mathcal{P}$ and $\mathcal{Q}$ and with one new node branching to the root of $\mathcal{P}$ or of $\mathcal{Q}$ depending on $[\![I]\!]_\rho$.

Building the PTS

$$\frac{\varphi \,|\, \Gamma \,|\, \Theta \;\Vdash_{\mathcal{P}}\; M : \sigma \qquad \varphi \,|\, \Gamma \,|\, \Theta \;\Vdash_{\mathcal{Q}}\; N : \sigma}{\varphi \,|\, \Gamma \,|\, \Theta \;\Vdash_{\mathcal{P} \oplus_p \mathcal{Q}}\; M \oplus_q N : \sigma}$$

$\mathcal{P} \oplus_p \mathcal{Q}$ is a PTS containing $\mathcal{P}$ and $\mathcal{Q}$ and with one new node branching to the root of $\mathcal{P}$ or of $\mathcal{Q}$ depending on a biased coin flip of probability q.

Building the PTS

$$\frac{\{\!|\,\Gamma\,|\!\} \subseteq \{\mathbf{bool}, \mathbf{int}\} \qquad \varphi,\ \overrightarrow{a} : \overrightarrow{\gamma} \,|\, \Gamma \,|\, f : \prod \overrightarrow{a} : \overrightarrow{\gamma}.\,\sigma \ \Vdash_{\mathcal{P}} \ V : \sigma \qquad \text{letrec}\,(\mathcal{P}, \rho) \text{ is AST for every } \rho \models \varphi}{\varphi \,|\, \Gamma \,|\, \Theta \vdash \text{letrec } f[\overrightarrow{a} : \overrightarrow{\gamma}] : \sigma \ = \ V : \prod \overrightarrow{a} : \overrightarrow{\gamma}.\,\sigma}$$

letrec $(\mathcal{P}, \rho)$ is a PTS obtained from $\mathcal{P}$ by making the loops on the leaves pointing to the root of $\mathcal{P}$.

Conjecture

We have strong hints that:

$$M \text{ has type } \sigma \quad \Rightarrow \quad M \text{ is AST.}$$

(we have a proof sketch based on the previous realizability argument).

Note that the system is again affine.

Conclusion

First type system:

- Affine type system with distributions of types
- Sized walks induced by the letrec rule and solved by an external PTIME procedure
- Subject reduction + soundness for AST

Second type system:

- Finer analysis: more expressive sizes, modelization by PTS
- No need for distribution types thanks to sum types
- Still affine
- Soundness is work in progress

Thank you for your attention!

Conclusion

First type system:

- Affine type system with distributions of types
- Sized walks induced by the letrec rule and solved by an external PTIME procedure
- Subject reduction + soundness for AST

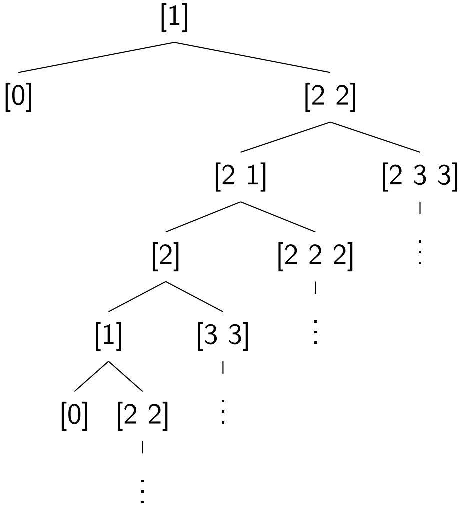
Second type system:

- Finer analysis: more expressive sizes, modelization by PTS
- No need for distribution types thanks to sum types
- Still affine
- Soundness is work in progress

Thank you for your attention!

Generalized Random Walks and the Necessity of Affinity

Tree of recursive calls, starting from 1:



Leftmost edges have probability $\frac{2}{3}$; rightmost ones $\frac{1}{3}$.

This random process is not AST.

Problem: modelisation by sized walk only makes sense for affine programs.