Stratified Bounded Affine Logic for Logarithmic Space

Ulrich Schöpp
University of Munich

Programming with Logarithmic Space

Goal: *Find a programming language for the functions computable in logarithmic space.*

Programming with Logarithmic Space

Goal: *Find a programming language for the functions computable in logarithmic space.*

Stratified Bounded Affine Logic (SBAL)

- Type system for a functional language with
  - higher-order functions
  - polymorphism
  - inductive datatypes
- Well-typed programs represent the functions computable in logarithmic space

Linear Logic and Complexity

Capture complexity classes as sub-systems of System F
by restricting Intuitionistic Affine Logic ($\otimes$, $\multimap$, $!$, $\forall$).

Polynomial time

Bounded Linear Logic — ! bounded by polynomial

Light Linear Logic — !,§

Soft Linear Logic — multiplexing only

Elementary functions

Elementary Affine Logic — no digging

...

Logarithmic space

Stratified Bounded Affine Logic — restrict ! and $\forall$

Bounded Affine Logic

- Bounded Linear Logic [Girard, Scedrov & Scott 1989]
- Affine variant [Hofmann & Scott 2000]
- BAL = Affine logic with bounded modality

Affine Logic

$!A$ — arbitrarily many copies of A

Bounded Affine Logic

$!_{x<p}A(x)$ — p many copies of A

$$!_{x<p}A(x) \;=\; A(0) \otimes A(1) \otimes \cdots \otimes A(p-1)$$

Bounded Affine Logic

- Formulae

$$A ::= \alpha(p_1, \ldots, p_n) \mid A \otimes A \mid A \multimap A \mid !_{x<p} A \mid \forall \alpha.\, A$$

- Rules for $\otimes$, $\multimap$, $\forall$ essentially as in Affine Logic

Affine Logic	Bounded Affine Logic
$\dfrac{}{\Gamma,\, \alpha \vdash \alpha}$	$\dfrac{p \leq q}{\Gamma,\, \alpha(p) \vdash \alpha(q)}$
$\dfrac{\Gamma,\, A[B/\alpha] \vdash C}{\Gamma,\, \forall \alpha.\, A \vdash C}$	$\dfrac{\Gamma,\, A[\lambda \vec{x}.\, B/\alpha] \vdash C}{\Gamma,\, \forall \alpha.\, A \vdash C}\ \star$

Bounded Affine Logic – Modality

$$\frac{\Gamma, A[0/x] \vdash B}{\Gamma, !_{x<1+w} A \vdash B} \qquad \frac{\Gamma, !_{x<p} A, !_{y<q} A[p+y/x] \vdash B}{\Gamma, !_{x<p+q+w} A \vdash B}$$

$$\frac{\Gamma \vdash A}{!_{x<p}\Gamma \vdash !_{x<p} A}$$

$$\frac{\Gamma,\ !_{x<p} !_{z<q(x)} A[z + \sum_{u<x} q(u)/y] \vdash B}{\Gamma,\ !_{y<\sum_{x<p} q(x)} A \vdash B}$$

Bounded Affine Logic

Natural Numbers

$$\mathbf{N}_x \;=\; \forall\alpha.\,!_{y<x}(\alpha(y) \multimap \alpha(y+1)) \multimap \alpha(0) \multimap \alpha(x)$$

Binary Lists

$$\mathbf{L}_x \;=\; \begin{array}{l}\forall\alpha.\;\; !_{y<x}(\alpha(y) \multimap \alpha(y+1)) \multimap \\ \qquad\;\; !_{y<x}(\alpha(y) \multimap \alpha(y+1)) \multimap \alpha(0) \multimap \alpha(x)\end{array}$$

Theorem: The proofs of sequents of the form

$$\mathbf{L}_x \vdash \mathbf{L}_{p(x)}$$

represent exactly the functions in PTIME.
[Girard, Scedrov & Scott '91],[Hofmann & Scott 2000]

Stratified Bounded Affine Logic

- BAL = Affine logic with bounded modality
- SBAL = BAL with bounded universal quantifier

Bounded Linear Logic

$\forall\alpha.\, A$ — unbounded quantifier

Stratified Bounded Linear Logic

$\forall\alpha \leq p.\, A$ — quantifier bounded by polynomial p

Quantifier Rules

Bounded Affine Logic

$$\frac{\Gamma,\ A[\lambda\vec{x}.\,B/\alpha] \vdash C}{\Gamma,\ \forall\alpha.\,A \vdash C} \qquad\qquad \frac{\Gamma \vdash A}{\Gamma \vdash \forall\alpha.\,A}\ \alpha \notin \Gamma$$

Stratified Bounded Affine Logic

$$\frac{\begin{array}{l}\Sigma \vdash B \leq p(\vec{x}) \\ \Sigma \mid \Gamma,\ A[\lambda\vec{x}.\,B/\alpha] \vdash C\end{array}}{\Sigma \mid \Gamma,\ \forall\alpha \leq p.\,A \vdash C}\,\star \qquad \frac{\Sigma,\ \alpha \leq p \mid \Gamma \vdash A}{\Sigma \mid \Gamma \vdash \forall\alpha \leq p.\,A}\ \alpha \notin \Gamma$$

Examples

$$\mathbf{N}_x^p \; = \; \forall \alpha \leq p. \, !_{y<x}(\alpha(y) \multimap \alpha(y+1)) \multimap \alpha(0) \multimap \alpha(x)$$

The following functions are definable as usual:

$$\begin{aligned}
&\vdash \quad zero : \mathbf{N}_x^p \\
&\vdash \quad succ : \mathbf{N}_x^p \multimap \mathbf{N}_{x+1}^p \\
&\vdash \quad add : \mathbf{N}_x^p \multimap \mathbf{N}_y^{q(p)} \multimap \mathbf{N}_{x+y}^p \\
&\vdash \quad pred : \mathbf{N}_x^{q(p)} \multimap \mathbf{N}_x^p
\end{aligned}$$

Coercion:

$$\vdash \quad coerc : \mathbf{N}_x^{r(p,q)} \multimap \, !_p \mathbf{N}_x^q$$

BAL: $\vdash \quad coerc : \mathbf{N}_x \multimap \, !_p \mathbf{N}_x$

SBAL – Complexity I

Corollary: Each SBAL-proof of a sequent

$$\mathbf{L}^r_x \vdash \mathbf{L}^q_{p(x)}$$

for sufficiently large q represents a LOGSPACE function.

$$\mathbf{L}^p_x \;=\; \begin{array}{l} \forall\alpha \leq p.\ \ !_{y<x}(\alpha(y) \multimap \alpha(y+1)) \multimap \\ \qquad\qquad\quad !_{y<x}(\alpha(y) \multimap \alpha(y+1)) \multimap \alpha(0) \multimap \alpha(x) \end{array}$$

Functional Programs in Logarithmic Space

- Pointer-algorithms
- Representation by higher-order functions

$$
\begin{aligned}
cons_0 &\quad : \quad \{0,1\}^* \to \{0,1\}^* \\
cons_0(w) &\quad = \quad 0w
\end{aligned}
$$

```
let cons0 (f:nat- {0,1,blank ) =
    fun (n:nat) -  if n=0 then 0 else f (n-1)
```

- Standard evaluation strategies (e.g. call-by-value) use linear space.

Functional Programs in Logarithmic Space

- Pointer-algorithms
- Representation by higher-order functions

$$\begin{aligned} cons_0 &: \{0,1\}^* \to \{0,1\}^* \\ cons_0(w) &= 0w \end{aligned}$$

```
let cons0 (f:nat- {0,1,blank ) =
      fun (n:nat) -  if n=0 then 0 else f (n-1)
```

- In SBAL:

$$\vdash cons_0 \colon (\mathbf{N}_x^q \multimap 3) \multimap (\mathbf{N}_x^p \multimap 3)$$

SBAL – Complexity II

$$\mathbf{W}_x^{p,q,r} = (!_q \mathbf{N}_x^p \multimap 3^q) \otimes \mathbf{N}_x^r$$

word encoded by a function

upper bound on word length

Theorem: Each SBAL-proof of $!_{y<t}\mathbf{W}_x^{q,r,s} \vdash \mathbf{W}_{p(x)}^{u,v,w}$ represents a LOGSPACE function.

Theorem: Each SBAL-proof of $\mathbf{N}_x^r \vdash \mathbf{N}_{p(x)}^q$ represents a LINSPACE function.

Skewed Iteration

$$\mathbf{N}^p_x \;=\; \forall\alpha \leq p.\, !_{y<x}(\alpha(y) \multimap \alpha(y+1)) \multimap \alpha(0) \multimap \alpha(x)$$

The following functions are definable as usual:

$$\begin{array}{ll} \vdash & \mathit{zero}\colon \mathbf{N}^p_x \\ \vdash & \mathit{succ}\colon \mathbf{N}^p_x \multimap \mathbf{N}^p_{x+1} \\ \vdash & \mathit{add}\colon \mathbf{N}^p_x \multimap \mathbf{N}^{q(p)}_y \multimap \mathbf{N}^p_{x+y} \\ \vdash & \mathit{pred}\colon \mathbf{N}^{q(p)}_x \multimap \mathbf{N}^p_x \end{array}$$

Subtraction is not!

Skewed Iteration

- Iteration principle must be implemented without storage of intermediate values, e.g. for ®$=\mathbf{L}_x$

$$n\colon \forall\alpha \leq p.\, !_{y<x}(\alpha(y) \multimap \alpha(y+1)) \multimap \alpha(0) \multimap \alpha(x)$$

$\Rightarrow$ Tail-call optimisation not possible

$$n\ f\ g \quad \longrightarrow$$

```
x:=g;
while n 0 do (x:=f(x); n:=n-1)
return x;
```

- Without this optimisation, subtraction (defined by iteration of *pred*) uses more than LOGSPACE.

Skewed Iteration

- Tail-call optimisation possible for small types:

$$A, B ::= \mathbf{B}^p \mid \mathbf{N}^p_x \mid A \otimes B$$

- Skewed Iteration
  For small types, size-polynomials can be ignored:

$$\mathbf{N}^p_x \multimap !_{y<x}(\mathbf{N}^q_x \multimap \mathbf{N}^r_x) \multimap \mathbf{N}^s_x \multimap \mathbf{N}^t_x$$

$$\mathbf{N}^p_x \multimap !_{y<x}(\mathbf{N}^q_x \otimes \mathbf{N}^r_x \multimap \mathbf{N}^s_x \otimes \mathbf{N}^t_x) \multimap \mathbf{N}^u_x \otimes \mathbf{N}^v_x \multimap \ldots$$

...

SBAL – Complexity III

Theorem: Each proof in SBAL+(Skew) of

$$!_{y<t}\mathbf{W}_x^{q,r,s} \vdash \mathbf{W}_{p(x)}^{u,v,w}$$

represents a LOGSPACE function.

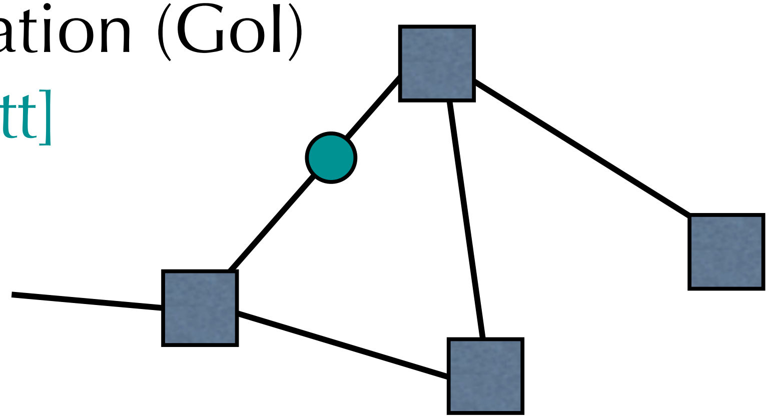
Theorem: Each LOGSPACE function can be represented in SBAL+(Skew) in this way.

From SBAL to Logarithmic Space

- BC$^-$ [Murawski & Ong], [Møller-Neergaard & Mairson]
- Model computation as an interaction process
  - Recomputation of intermediate values
  - Question/Answer dialogues

$\Rightarrow$ Game Semantics

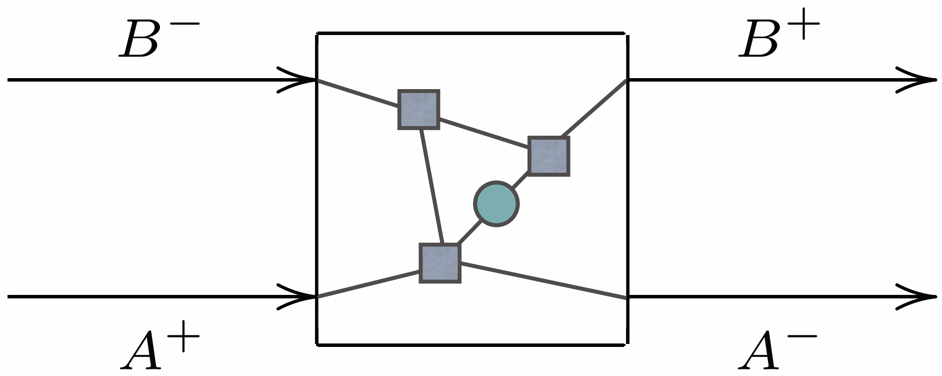
$\Rightarrow$ Geometry of Interaction Situation (GoI) [Abramsky, Haghverdi & Scott]



A Geometry of Interaction Situation

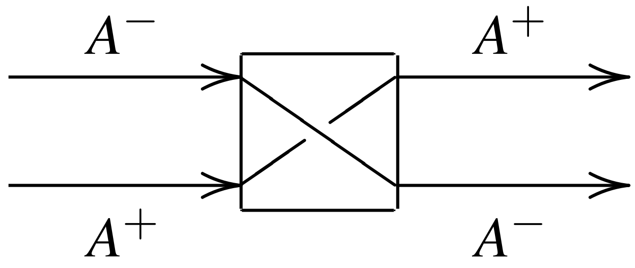
- Objects $A = (A^-, A^+)$
- Morphisms $A \longrightarrow B$

Partial functions $A^+ + B^- \longrightarrow A^- + B^+$



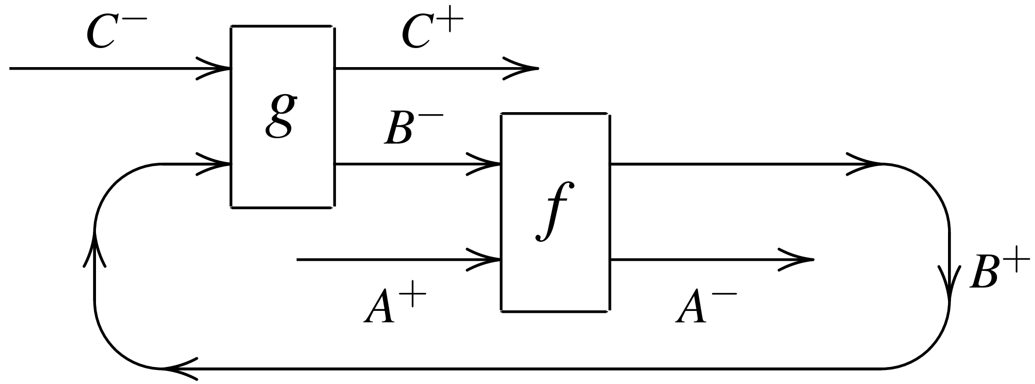
A Geometry of Interaction Situation

- Identity $id : A \to A$



- Composition of $f : A \to B$ and $g : B \to C$

$g \cdot f =$



GoI Structure

- Tensor product

$$A \otimes B = (A^- + B^-, A^+ + B^+)$$

- Linear function space

$$A \multimap B = (A^+ + B^-, A^- + B^+)$$

- Modality

$$!A = (\mathbb{N} \times A^-, \mathbb{N} \times A^+)$$

SBAL and GoI

- Identify LOGSPACE-fragment of this GoI Situation
  - Size bound on messages

$$!_p A = \big(\{n \mid n < p\} \times A^-, \{n \mid n < p\} \times A^+\big)$$

instead of

$$!A = (\mathbb{N} \times A^-, \mathbb{N} \times A^+)$$

  - Restriction of morphisms to LINSPACE
- Translation of SBAL-typed terms in GoI-programs
- Realisability model based on the BAL-model of [Hofmann & Scott 2000]

Realisability

Interpret SBAL-formula with resource variables in X by

$$(|A|, \|A\|, \Vdash)$$

System F type

Realising Object

Realisation Relation

Realisability

Interpret SBAL-formula with resource variables in X by

$$(|A|, \|A\|, \Vdash)$$

System F type

GoI-Object $\|A\|_\eta$ for
each X-environment η

Relation $\eta, e \Vdash a$

$$\eta \in V(X), \quad e\colon \|A\|^-_\eta \to \|A\|^+_\eta, \quad a \in |A|$$

Realisability

Interpret SBAL-formula with resource variables in X by

$$(|A|, \|A\|, \Vdash)$$

Size condition:

$$\forall \boldsymbol{\eta} \in V(X).\ \forall x \in \|A\|^{-}_{\boldsymbol{\eta}} \cup \|A\|^{+}_{\boldsymbol{\eta}}.\ |x| \leq c \cdot |\boldsymbol{\eta}| + d.$$

Realisability – Morphisms

Interpret SBAL sequent $\Sigma \mid A \vdash B$ by a System F-term $M\colon |A| \Rightarrow |B|$ that has a realiser:

$$r: \prod_{\eta \in V(X)} \|A\|_\eta \to \|B\|_\eta$$

1. If $\eta, e \Vdash a$ then $\eta,\ r_\eta e \Vdash M a$.
2. The function $\langle \eta, q \rangle \mapsto r_\eta(q)$ is computable in linear space.

Example – Small Numbers

$$
\begin{array}{rcl}
|S_x| & = & \mathbb{N} \\
\|S_x\|^-_\eta & = & \{*\} \\
\|S_x\|^+_\eta & = & \{n \in \mathbb{N} \mid n \leq \eta(x)\} \\
\eta, e \Vdash n & \iff & n \leq \eta(x) \ \wedge \ e(*) = n
\end{array}
$$

$$
succ\colon S_x \to S_{x+1}
$$

$$
\begin{array}{llcl}
r_\eta\colon & \|S_x\|^+_\eta + \ \|S_{x+1}\|^-_\eta & \longrightarrow & \|S_x\|^-_\eta + \ \|S_{x+1}\|^+_\eta \\
 & \qquad\qquad * & \longmapsto & * \\
 & n & \longmapsto & \qquad\qquad n+1
\end{array}
$$

Modality

$$
\begin{array}{rcl}
|!_{x<p}A| & = & |A| \\
\|!_{x<p}A\|_\eta & = & \left(\sum\limits_{n<p[\eta]} \|A\|^{-}_{\eta[n/x]}, \sum\limits_{n<p[\eta]} \|A\|^{+}_{\eta[n/x]}\right)
\end{array}
$$

Contraction

$$!_{x<p+q}A \multimap (!_{x<p}A) \otimes (!_{y<q}A[p+y/x])$$

Domain of realiser:

$$\sum_{n<p[\eta]+q[\eta]} \|A\|^{+}_{\eta[n/x]} + \left(\sum_{n<p[\eta]} \|A\|^{-}_{\eta[n/x]} + \sum_{n<q[\eta]} \|A\|^{-}_{\eta[n+p[\eta]/x]} \right)$$

Range of realiser:

$$\sum_{n<p[\eta]+q[\eta]} \|A\|^{-}_{\eta[n/x]} + \left(\sum_{n<p[\eta]} \|A\|^{+}_{\eta[n/x]} + \sum_{n<q[\eta]} \|A\|^{+}_{\eta[n+p[\eta]/x]} \right)$$

Contraction

$$!_{x<p+q} A \multimap (!_{x<p} A) \otimes (!_{y<q} A[p+y/x])$$

$$\sum_{n<p[\eta]+q[\eta]} \|A\|^{+}_{\eta[n/x]} + \left(\sum_{n<p[\eta]} \|A\|^{-}_{\eta[n/x]} + \sum_{n<q[\eta]} \|A\|^{-}_{\eta[n+p[\eta]/x]} \right)$$

$$\sum_{n<p[\eta]+q[\eta]} \|A\|^{-}_{\eta[n/x]} + \left(\sum_{n<p[\eta]} \|A\|^{+}_{\eta[n/x]} + \sum_{n<q[\eta]} \|A\|^{+}_{\eta[n+p[\eta]/x]} \right)$$

Complexity

Theorem: Each morphism

$$!_q S_x \longrightarrow S_{p(x)}$$

has a LINSPACE-function as its underlying function.

Theorem: Each morphism

$$!_q(S_x \multimap S_3) \otimes !_q S_x \longrightarrow (S_{p(x)} \multimap S_3) \otimes S_{p(x)}$$

represents a LOGSPACE-function.

Universal Quantifier

- Universal quantifiers in realisability models

$$r \text{ realises } \; f : \forall\alpha.\, A$$

$$\Longleftrightarrow$$

$$r \text{ realises all instances } \quad f\; B : A[B/\alpha]$$

- Realising object?

$\|\forall\alpha.\, \alpha \multimap \alpha\|_\eta$ must be as good as all instances

$$\|B \multimap B\|_\eta,\; \|(B \multimap B) \multimap (B \multimap B)\|_\eta,$$
$$\|(C \otimes (B \multimap B)) \multimap (C \otimes (B \multimap B))\|_\eta, \ldots$$

Universal Quantifier

- Universal quantifiers in realisability models

$$r \text{ realises } \; f : \forall \alpha.\, A$$

$$\Longleftrightarrow$$

$$r \text{ realises all instances } \quad f\, B : A[B/\alpha]$$

- Realising object?

$\|\forall \alpha.\, \alpha \multimap \alpha\|_\eta$ must be as good as all instances

$$\|\forall \alpha.\, \alpha \multimap \alpha\|_\eta = (\mathbb{N}, \mathbb{N}) \multimap (\mathbb{N}, \mathbb{N}) \quad ?$$

Universal Quantifier

- Universal quantifiers in realisability models

$$r \text{ realises } f : \forall\alpha.\, A$$

$$\Longleftrightarrow$$

Condition:

$$\forall \boldsymbol{\eta} \in V(X).\ \forall x \in \|A\|^{-}_{\eta} \cup \|A\|^{+}_{\eta}.\ |x| \leq c \cdot |\boldsymbol{\eta}| + d.$$

- Re

$$\|\forall\alpha.\, \alpha \multimap \alpha\|_{\eta} \text{ must be as good as all instances}$$

$$\|\forall\alpha.\, \alpha \multimap \alpha\|_{\eta} = (\mathbb{N}, \mathbb{N}) \multimap (\mathbb{N}, \mathbb{N}) \quad ?$$

Universal Quantifier

- Restrict the range of the quantifier to objects whose questions and answers can be encoded within a certain bound.

$$\|\forall \alpha \leq p.\, \alpha \multimap \alpha\|_\eta = (\mathbb{N}_{p[\eta]}, \mathbb{N}_{p[\eta]}) \multimap (\mathbb{N}_{p[\eta]}, \mathbb{N}_{p[\eta]})$$

Conclusion

SBAL is a functional programming for LOGSPACE with higher-order functions, polymorphism and inductive datatypes.

Further Work

- Simplification without resource polynomials, perhaps as in Light Affine Logic
- Logical status of skewed iteration
- Structured data (e.g. graphs)