A type system for stack-based memory management

wo4mei3

January 6, 2026

Abstract

tofte-talpin system [3] は個々の値に対してどのリージョンに属するかアノテーションすることによって静的メモリ管理を実現する理論であるが、実際のアノテーションは難しいことで知られている。そこで自分はこの難しさの原因の一つとして tofte-talpin system の自由度の高さが一因となってると考えた。tofte-talpin system は region が作られた順序に関わらず、生存してさえいれば任意の region に参照を作ることができる。しかし、それとは違い往来の stack 領域でのメモリ管理には、新たに作られる参照はその時点での stack の一番上を伸長することによって作られるという参照の作られる場所に一意性がある。よって、この制約を tofte-talpin system でも仮定すれば、いつも参照は一番最後に確保された region に確保されるようになり、アノテーションの必要もなくなると考えた。今現在、この論文で導入した言語は region 多相や effect 多相を実現していない。自分の手には追えないものであると見做してる。この論文では一番最後に束縛された region 変数のみに対して参照を作成できるようにしてある。region 抽象によって束縛された region 変数に対しても同様である。region 抽象に対して任意の region name を適用できるようにすれば、この言語は heap 領域と同等のメモリ管理能力を持つと想定している。

1 Introduction

c 言語においては、必ずしも heap 領域を用いてなくても、stack 領域を扱っているだけで dangling pointer を発生させることができる。そのような例として以下のようなものがある。

Listing 1: dangling pointer on stack area

```
{
   char *dp = NULL;
   /* ... */
   {
       char c;
       dp = &c;
   }
     /* c falls out of scope */
     /* dp is now a dangling pointer */
}
```

この論文ではこの問題を防ぐための [3] で提案された tofte-talpin 型システムを流用することを考えた。この論文では stack 領域のみを扱った言語を考え、それが tofte-talpin 型システムに対して型安全性を満たすことを証明する。c 言語でも新しい自動変数は stack 領域の一番上を伸長してメモリ確保されるが、この言語でも参照はすべて stack 領域に保存されるが、stack の一番上の store の一番新しい要素として束縛するようにしている。この論文内での言語の文法や操作的意味論、型システムは [1] のこの論文に似通っているが、この論文内での言語は stack 領域上の参照しか扱わないため、参照をブロックの外へエスケープすることができず、[1] の言語に比べて書けるプログラムに制限がある。この論文内の言語に [1] の論文内で定義された言語のような能力を持たせるためには、region 多相や effect 多相を導入する必要がある。これは今後の研究の発展に任せる。記法は TAPL に準じる。

Syntax

$t ::=$		terms:
	x	variable
	$\lambda x : T.\ t$	lambda abstraction
	$t\ t$	application
	$unit$	constant unit
	$ref\ t\ at\ p$	reference creation
	$!t$	dereference
	$t := t$	assignment
	(i, l)	location
	$begin\ p\ t\ end$	block
$v ::=$		values:
	$\lambda x : T.\ t$	lambda abstraction
	$unit$	constant unit
	(i, l)	location
l		store location
$p ::=$		places:
	ρ	block variable
	i	stack location
	$\bullet$	deallocated
$\phi ::=$		effects:
	$\{p,\ ...,\ p\}$	effect
$T ::=$		types:
	$T \rightarrow^{\phi} T$	type of functions
	$Unit$	unit type
	$(T\ Ref,\ p)$	type of reference cells
$\Gamma ::=$		contexts:
	$\varnothing$	empty context
	$\Gamma,\ x : T$	term variable binding
$\Delta ::=$		block contexts:
	$\varnothing$	empty block context
	$\Delta,\ p$	block context

$\mu ::=$		stores:
	$\varnothing$	empty store
	$\mu,\ l = v$	location binding
$M ::=$		stacks:
	$\varnothing$	empty stack
	$M,\ i = \mu$	store binding
$\sigma ::=$		store typings:
	$\varnothing$	empty store typing
	$\sigma,\ l : T$	location typing
$\Sigma ::=$		stack typings:
	$\varnothing$	empty stack typing
	$\Sigma,\ i : \sigma$	store typing

Evalution $\boxed{t, M \to t', M'}$

$$\frac{t_1, M \to t_1', M'}{t_1\ t_2, M \to t_1'\ t_2, M'}\ \text{E-App1}$$

$$\frac{t_2, M \to t_2', M'}{v\ t_2, M \to v\ t_2', M'}\ \text{E-App2}$$

$$\frac{}{(\lambda x : T.\ t)\ v, M \to [x \to v]t, M}\ \text{E-AppAbs}$$

$$\frac{l \notin dom(\mu)}{\begin{array}{l} ref\ v\ at\ i, (M, i = \mu) \\ \to (i, l), (M, i = (\mu, l = v)) \end{array}}\ \text{E-Refv}$$

$$\frac{t, M \to t', M'}{ref\ t\ at\ i, M \to ref\ t'\ at\ i, M'}\ \text{E-Ref}$$

$$\frac{M(i)(l) = v}{!(i, l), M \to v, M}\ \text{E-DerefLoc}$$

$$\frac{t, M \to t', M'}{!t, M \to !t', M'}\ \text{E-Deref}$$

$$\frac{}{\begin{array}{l} (i, l) := v, M \\ \to unit, [i \to [l \to v]M(i)]M \end{array}}\ \text{E-Assign}$$

$$\frac{t_1, M \to t_1', M'}{t_1 := t_2, M \to t_1' := t_2', M'}\ \text{E-Assign1}$$

$$\frac{t_2, M \to t_2', M'}{v := t_2, M \to v' := t_2', M'}\ \text{E-Assign2}$$

$$\frac{i \notin dom(M)}{\begin{array}{l} begin\ \rho\ t\ end, M \\ \to [\rho \to i]begin\ \rho\ t\ end, (M, i = \emptyset) \end{array}}\ \text{E-BlockBegin}$$

$$\frac{}{begin\ i\ v\ end, (M, i = \mu) \to [i \to \bullet]v, M}\ \text{E-BlockEnd}$$

$$\frac{t, M \to t', M'}{begin\ i\ t\ end, M \to begin\ i\ t'\ end, M'}\ \text{E-Block}$$

Typing $\boxed{\Gamma; \Delta; \Sigma_1; \Sigma_2 \vdash t : T \& \phi}$

$$\frac{x : T \in \Gamma}{\Gamma; \Delta; \Sigma_1; \emptyset \vdash x : T \& \emptyset}\ \text{T-Var}$$

$$\frac{\Gamma, x : T_1; \Delta; \Sigma_1; \emptyset \vdash t : T_2 \& \phi_2}{\Gamma; \Delta; \Sigma_1; \emptyset \vdash \lambda x : T_1.\ t : T_1 \rightarrow^{\phi_2} T_2 \& \emptyset}\ \text{T-Abs}$$

$$\frac{\begin{array}{c}\Gamma; \Delta; \Sigma_1; \Sigma_2 \vdash t_1 : T_1 \rightarrow^{\phi} T_2 \& \phi_1 \\ \Gamma; \Delta; \Sigma_1, \Sigma_2; \Sigma_3 \vdash t_2 : T_1 \& \phi_2\end{array}}{\Gamma; \Delta; \Sigma_1; \Sigma_2, \Sigma_3 \vdash t_1\ t_2 : T_2 \& \phi \cup \phi_1 \cup \phi_2}\ \text{T-App}$$

$$\frac{}{\Gamma; \Delta; \Sigma_1; \emptyset \vdash unit : Unit \& \emptyset}\ \text{T-Unit}$$

$$\frac{\Sigma_1(i)(l) = T}{\Gamma; \Delta; \Sigma_1; \emptyset \vdash (i, l) : (T\ ref,\ i) \& \emptyset}\ \text{T-Loc}$$

$$\frac{}{\Gamma; \Delta; \Sigma_1; \emptyset \vdash (\bullet, l) : (T\ ref,\ \bullet) \& \emptyset}\ \text{T-Dead}$$

$$\frac{\Gamma; \Delta, \rho; \Sigma_1; \emptyset \vdash t : T \& \phi}{\Gamma; \Delta, \rho; \Sigma_1; \emptyset \vdash ref\ t\ at\ \rho : (T\ ref,\ \rho) \& \phi \cup \{\rho\}}\ \text{T-Ref1}$$

$$\frac{\Gamma; \Delta, i; \Sigma_1, i : \sigma; \Sigma_2 \vdash t : T \& \phi}{\Gamma; \Delta, i; \Sigma_1, i : \sigma; \Sigma_2 \vdash ref\ t\ at\ i : (T\ ref,\ i) \& \phi \cup \{i\}}\ \text{T-Ref2}$$

$$\frac{\Gamma; \Delta; \Sigma_1; \Sigma_2 \vdash t : (T\ ref,\ p) \& \phi}{\Gamma; \Delta; \Sigma_1; \Sigma_2 \vdash !t : T \& \phi \cup \{p\}}\ \text{T-DeRef}$$

$$\frac{\begin{array}{c}\Gamma; \Delta; \Sigma_1; \Sigma_2 \vdash t_1 : (T\ ref,\ p) \& \phi_1 \\ \Gamma; \Delta; \Sigma_1, \Sigma_2; \Sigma_3 \vdash t_2 : T \& \phi_2\end{array}}{\Gamma; \Delta; \Sigma_1; \Sigma_2, \Sigma_3 \vdash t_1 := t_2 : Unit \& \phi_1 \cup \phi_2 \cup \{p\}}\ \text{T-Assign}$$

$$\frac{\Gamma; \Delta, \rho; \Sigma_1; \emptyset \vdash t : T \& \phi \quad \rho \notin FPV(\Gamma, T) \quad \rho \notin \Delta}{\Gamma; \Delta; \Sigma_1; \emptyset \vdash begin\ \rho\ t\ end : T \& \phi \setminus \{\rho\}}\ \text{T-Block1}$$

$$\frac{\Gamma; \Delta, i; \Sigma_1, i : \sigma; \Sigma_2 \vdash t : T \& \phi \quad i \notin FPV(\Gamma, T) \quad i \notin \Delta}{\Gamma; \Delta; \Sigma_1; i : \sigma, \Sigma_2 \vdash begin\ i\ t\ end : T \& \phi \setminus \{i\}}\ \text{T-Block2}$$

Store Typing $\boxed{\Sigma \vdash \mu : \sigma}$

$$\frac{dom(\sigma) = dom(\mu) \quad (\forall l \in dom(\sigma)) \emptyset; \emptyset; \Sigma; \emptyset \vdash \mu(l) : \sigma(l) \& \emptyset}{\Sigma \vdash \mu : \sigma}\ \text{T-Store}$$

Stack Typing $\boxed{\Sigma \vdash M}$

$$\frac{}{\emptyset \vdash \emptyset}\ \text{T-Stack1}$$

$$\frac{\Sigma \vdash M \quad \Sigma \vdash \mu : \sigma \quad FPV(\Sigma) \subseteq dom(\Sigma)}{\Sigma, i : \sigma \vdash M, i = \mu}\ \text{T-Stack2}$$

Configuration Typing $\boxed{\Sigma \vdash t, M : T \& \phi}$

$$\frac{\emptyset;\emptyset;\Sigma_1;\Sigma_2 \vdash t : T\&\phi \quad \Sigma_1, \Sigma_2 \vdash M}{\Sigma_1;\Sigma_2 \vdash t, M : T\&\phi}\ \text{T-Conf}$$

FPV(t) を次のように定義する。

$$FPV(x) = \emptyset$$
$$FPV(\lambda x : T.t) = FPV(T) \cup FPV(t)$$
$$FPV(t_1\ t_2) = FPV(t_1) \cup FPV(t_2)$$
$$FPV(unit) = \emptyset$$
$$FPV(ref\ t\ at\ p) = FPV(t) \cup \{p\}$$
$$FPV(!t) = FPV(t)$$
$$FPV(t_1\ :=\ t_2) = FPV(t_1) \cup FPV(t_2)$$
$$FPV((i, l)) = \{i\}$$
$$FPV((\bullet, l)) = \{\bullet\}$$
$$FPV(begin\ p\ t\ end) = FPV(t) \setminus \{p\}$$

FPV(T) を次のように定義する。

$$FPV(T_1 \rightarrow^{\phi} T_2) = FPV(T_1) \cup \phi \cup FPV(T_2)$$
$$FPV(Unit) = \emptyset$$
$$FPV((T\ ref,\ p)) = FPV(T) \cup \{p\}$$

FPV(Γ) を次のように定義する。

$$FPV(\Gamma) = \bigcup_{(x:T)\in\Gamma} FPV(T)$$

FPV(σ) を次のように定義する。

$$FPV(\sigma) = \bigcup_{(l:T)\in\sigma} FPV(T)$$

FPV(Σ) を次のように定義する。

$$FPV(\Sigma) = \bigcup_{(i:\sigma)\in\Sigma} FPV(\sigma)$$

最後に、PFPV(t) を次のように定義する。

$$PFPV(x) = \emptyset$$
$$PFPV(\lambda x : T.t) = FPV(T) \cup PFPV(t)$$
$$PFPV(t_1\ t_2) = PFPV(t_1) \cup PFPV(t_2)$$
$$PFPV(unit) = \emptyset$$
$$PFPV(ref\ t\ at\ p) = PFPV(t) \cup \{p\}$$
$$PFPV(!t) = PFPV(t)$$
$$PFPV(t_1\ :=\ t_2) = PFPV(t_1) \cup PFPV(t_2)$$
$$PFPV((i, l)) = \emptyset$$
$$PFPV((\bullet, l)) = \{\bullet\}$$
$$PFPV(begin\ p\ t\ end) = PFPV(t) \setminus \{p\}$$

2 健全性の証明

$\Gamma;\Delta;\Sigma_1;\Sigma_2 \vdash t : T\&\phi$ は、型環境 Γ、ブロック環境 Δ、事前の stack 型環境 Σ_1、項が確保した stack 型環境 Σ_2 のもとで、項 t が型 T を持ち、効果 ϕ を持つことを表す。ここで、Σ_1 は項の実行前に確保された stack で、Σ_2 は項自身が確保した stack である。Σ_2 は項が値にまで簡約しきる間にすべて解放されてないとならない。型システムではこれら 2 つを区別する。

T-App や T-Assign では二つの部分導出を仮定に必要とするが、これらの項は実行時にはそれぞれ別々の stack を確保する。評価規則によって先に評価される項が確保した stack から先に積まれるので、これらの演算子は左辺が確保した stack から先に積まれることが型付け規則に反映されてある。

$FPV(t)$ や $PFPV(t)$ は単調性が成り立つ。つまり型付け規則の導出の各部分項でいえたら、元々の項でも言える。この性質があるので帰納法の証明ではこちらを使う。よって $FPV(T)$ などは直接は使わない

$PFPV(t)$ は $FPV(t)$ とは違いすべての要素が $FPV(T)$ または ϕ に含まれる。よって帰納法では使えない仮定である $FPV(T)$ や ϕ を直接使うのではなく、$PFPV(t)$ に直してから使う。

Lemma 2.1 (store 型付けに対する代入補題)**.**
$\Sigma \vdash \mu : \sigma$ かつ $\sigma(l) = T$ かつ $\emptyset;\emptyset;\Sigma;\emptyset \vdash v : T\&\emptyset$ ならば
$\Sigma \vdash [l \to v]\mu : \sigma$

Definition 2.2 (任意の store の拡張)**.**
$dom(\Sigma) = dom(\Sigma')$ かつ $\forall i \in dom(\Sigma).\Sigma(i) \subseteq \Sigma'(i)$
ならば $\forall i.\Sigma(i) \subseteq \Sigma'(i)$ と書く。

Definition 2.3 (stack の拡張)**.**
$dom(\Sigma) \subseteq dom(\Sigma')$ かつ $\forall i \in dom(\Sigma).\Sigma(i) \subseteq \Sigma'(i)$
ならば $\Sigma \subseteq \Sigma'$ と書く。

Lemma 2.4 (stack の拡張補題)**.**
$\Gamma;\Delta;\Sigma_1;\emptyset \vdash t : T\&\phi$ かつ $\Sigma_1 \subseteq \Sigma_1'$ ならば $\Gamma;\Delta;\Sigma_1';\emptyset \vdash t : T\&\phi$

Lemma 2.5.
$\Gamma;\Delta;\Sigma_1;\emptyset \vdash t : T\&\phi$ かつ $FPV(\Sigma_1) \subseteq dom(\Sigma_1)$ ならば $FPV(T) \subseteq dom(\Sigma_1)$

Corollary 2.6.
$\Sigma \vdash \mu : \sigma$ かつ $FPV(\sigma) \subseteq dom(\Sigma)$ ならば $FPV(\sigma) \subseteq dom(\Sigma)$

Corollary 2.7.
$\Sigma \vdash M$ ならば $FPV(\Sigma) \subseteq dom(\Sigma)$

Lemma 2.8 (stack push 補題)**.**
$\Gamma;\Delta;\Sigma_1;\emptyset \vdash t : T\&\phi$ かつ $\rho \in \Delta$ ならば $[\rho \to i]\Gamma;[\rho \to i]\Delta;[\rho \to i]\Sigma_1, i : \emptyset;\emptyset \vdash [\rho \to i]t : [\rho \to i]T\&[\rho \to i]\phi$

証明.
$\Gamma;\Delta;\Sigma_1;\emptyset \vdash t : T\&\phi$ に stack の拡張補題を適用して、 $\Gamma;\Delta;\Sigma_1, i : \emptyset;\emptyset \vdash t : T\&\phi$ を得た後、
$\Gamma;\Delta;\Sigma_1, i : \emptyset;\emptyset \vdash t : T\&\phi$ かつ $\rho \in \Delta$ ならば $[\rho \to i]\Gamma;[\rho \to i]\Delta;[\rho \to i]\Sigma_1, i : \emptyset;\emptyset \vdash [\rho \to i]t : [\rho \to i]T\&[\rho \to i]\phi$ を、$\Gamma;\Delta;\Sigma_1, i : \emptyset;\emptyset \vdash t : T\&\phi$ の導出に関する帰納法によって証明する。
最後に使われた規則によって場合分けする。
興味深いのは、T-Ref1、T-Block1、T-Block2 のときである。
最後の規則が T-Block2 のときは、Σ_2 が $\emptyset$ にはならないので、このとき、空虚な真になる。
最後の規則が T-Ref1 のとき、 $\Gamma;\Delta,\rho_1;\Sigma_1, i : \emptyset;\emptyset \vdash ref\ t\ at\ \rho_1 : (T\ ref,\ \rho_1)\&\phi \cup \{\rho_1\}$ である。T-Ref1 の定義より、$\Gamma;\Delta,\rho_1;\Sigma_1, i : \emptyset;\emptyset \vdash t : T\&\phi$ である。これに、帰納法の仮定を適用して、$[\rho \to i]\Gamma;[\rho \to i]\Delta,\rho_1;[\rho \to i]\Sigma_1, i : \emptyset;\emptyset \vdash [\rho \to i]t : [\rho \to i]T\&[\rho \to i]\phi$ を得る。ここで、$\rho \neq \rho_1$ のとき、T-Ref1 を適用して、$[\rho \to i]\Gamma;[\rho \to i]\Delta,\rho_1;[\rho \to i]\Sigma_1, i : \emptyset;\emptyset \vdash ref\ [\rho \to i]t\ at\ \rho_1 : ([\rho \to i]T\ ref,\ \rho_1)\&[\rho \to i]\phi \cup \{\rho_1\}$ すなわち、$[\rho \to i]\Gamma;[\rho \to i](\Delta,\rho_1);[\rho \to i]\Sigma_1, i : \emptyset;\emptyset \vdash [\rho \to i](ref\ t\ at\ \rho_1) :$

$[\rho \to i](T\ ref,\ \rho_1)\&[\rho \to i](\phi \cup \{\rho_1\})$ を得る。ここで、$\rho = \rho_1$ のとき、T-Ref2 を適用して、$[\rho \to i]\Gamma; [\rho \to i]\Delta, i; [\rho \to i]\Sigma_1, i : \emptyset; \emptyset \vdash ref\ [\rho \to i]t\ at\ i : ([\rho \to i]T\ ref,\ i)\&[\rho \to i]\phi \cup \{i\}$ すなわち、$[\rho \to i]\Gamma; [\rho \to i](\Delta, \rho_1); [\rho \to i]\Sigma_1, i : \emptyset; \emptyset \vdash [\rho \to i](ref\ t\ at\ \rho_1) : [\rho \to i](T\ ref,\ \rho_1)\&[\rho \to i](\phi \cup \{\rho_1\})$ を得る。

最後の規則が T-Block1 のときは、$\Gamma; \Delta; \Sigma_1, i : \emptyset; \emptyset \vdash begin\ \rho_1\ t\ end : T\&\phi \setminus \{\rho_1\}$ である。ここで、T-Block1 の定義より、$\Gamma; \Delta, \rho_1; \Sigma_1, emptyset; \emptyset \vdash t : T\&\phi$、$\rho_1 \notin FPV(\Gamma, T)$、$\rho_1 \notin \Delta$ である。$\Gamma; \Delta, \rho_1; \Sigma_1, \emptyset; \emptyset \vdash t : T\&\phi$ に帰納法の仮定を適用して、$[\rho \to i]\Gamma; [\rho \to i]\Delta, \rho_1; [\rho \to i]\Sigma_1, i : \emptyset; \emptyset \vdash [\rho \to i]t : [\rho \to i]T\&[\rho \to i]\phi$ を得る。よって、T-Block1 を適用して、$[\rho \to i]\Gamma; [\rho \to i]\Delta; [\rho \to i]\Sigma_1, i : \emptyset; \emptyset \vdash begin\ \rho_1\ [\rho \to i]t\ end : [\rho \to i]T\&[\rho \to i]\phi \setminus \{\rho_1\}$ を得る。すなわち、$[\rho \to i]\Gamma; [\rho \to i]\Delta; [\rho \to i]\Sigma_1, i : \emptyset; \emptyset \vdash [\rho \to i](begin\ \rho_1\ t\ end) : [\rho \to i]T\&[\rho \to i]\phi \setminus \{\rho_1\}$ を得る。

その他の規則は自明であるので、省略する。 □

Lemma 2.9.
$\Gamma; \Delta; \Sigma_1; \emptyset \vdash t : T\&\phi$ ならば $PFPV(t) \subseteq FPV(T) \cup \phi$

証明.
これはたぶんあってると思う。自分では証明確認してない。これが間違えていたら stack pop 補題成り立たないし、型保存定理も証明できない。 □

Lemma 2.10.
$\Gamma; \Delta; \Sigma_1; \emptyset \vdash t : T\&\phi$ かつ $i \notin PFPV(t)$ ならば $[i \to \bullet]\Gamma; [i \to \bullet]\Delta; [i \to \bullet]\Sigma_1; \emptyset \vdash [i \to \bullet]t : [i \to \bullet]T\&[i \to \bullet]\phi$

証明.
$\Gamma; \Delta; \Sigma_1; \emptyset \vdash t : T\&\phi$ の導出に関する帰納法によって証明する。
最後に使われた規則によって場合分けする。
最後の規則が T-Var のとき、$\Gamma; \Delta; \Sigma_1; \emptyset \vdash x : T\&\emptyset$ であり、定義より、$x : T \in \Gamma$ である。よって、$[i \to \bullet]$ を適用して、$[i \to \bullet]x : [i \to \bullet]T \in [i \to]\Gamma$ である。再び T-Var を適用して、$[i \to \bullet]\Gamma; [i \to \bullet]\Delta; [i \to \bullet]\Sigma_1; \emptyset \vdash [i \to \bullet]x : [i \to \bullet]T\&[i \to \bullet]\emptyset$ である。
最後の規則が T-Abs のとき、$\Gamma; \Delta; \Sigma_1; \emptyset \vdash \lambda x : T_1.\ t : T_1 \to^{\phi_2} T_2\&\emptyset$ である。定義より、$\Gamma, x : T_1; \Delta; \Sigma_1; \emptyset \vdash t : T_2\&\phi_2$ である。帰納法の仮定を適用して、$[i \to \bullet]\Gamma, [i \to \bullet]x : T_1; [i \to \bullet]\Delta; [i \to \bullet]\Sigma_1; \emptyset \vdash [i \to \bullet]t : [i \to \bullet]T_2\&[i \to \bullet]\phi_2$ T-Abs を適用して、$[i \to \bullet]\Gamma; [i \to \bullet]\Delta; [i \to \bullet]\Sigma_1; \emptyset \vdash \lambda[i \to \bullet]x : [i \to \bullet]T_1.\ [i \to \bullet]t : [i \to \bullet]T_1 \to^{[i \to \bullet]\phi_2} [i \to \bullet]T_2\&\emptyset$、すなわち、$[i \to \bullet]\Gamma; [i \to \bullet]\Delta; [i \to \bullet]\Sigma_1; \emptyset \vdash [i \to \bullet](\lambda x : T_1.\ t) : [i \to \bullet](T_1 \to^{\phi_2} T_2)\&\emptyset$ を得る。
最後の規則が T-App のとき、$\Gamma; \Delta; \Sigma_1; \emptyset \vdash t_1\ t_2 : T_2\&\phi \cup \phi_1 \cup \phi_2$ である。定義より、$\Gamma; \Delta; \Sigma_1; \Sigma_2 \vdash t_1 : T_1 \to^{\phi} T_2\&\phi_1$、$\Gamma; \Delta; \Sigma_1; \Sigma_3 \vdash t_2 : T_1\&\phi_2$、$\emptyset = \Sigma_2 \veebar \Sigma_3$ である。よって、$\veebar$ の定義より、$\Sigma_2 = \Sigma_3 = \emptyset$ である。よって、それぞれの部分導出に帰納法の仮定を適用して、$[i \to \bullet]\Gamma; [i \to \bullet]\Delta; [i \to \bullet]\Sigma_1; \emptyset \vdash [i \to \bullet]t_1 : [i \to \bullet]T_1 \to^{[i \to \bullet]\phi} [i \to \bullet]T_2\&[i \to \bullet]\phi_1$、$[i \to \bullet]\Gamma; [i \to \bullet]\Delta; [i \to \bullet]\Sigma_1; \emptyset \vdash [i \to \bullet]t_2 : [i \to \bullet]T_1\&[i \to \bullet]\phi_2$ を得る。よって、T-App を適用して、$[i \to \bullet]\Gamma; [i \to \bullet]\Delta; [i \to \bullet]\Sigma_1; \emptyset \vdash [i \to \bullet]t_1\ [i \to \bullet]t_2 : [i \to \bullet]T_2\&[i \to \bullet]\phi \cup [i \to \bullet]\phi_1 \cup [i \to \bullet]\phi_2$、すなわち、$[i \to \bullet]\Gamma; [i \to \bullet]\Delta; [i \to \bullet]\Sigma_1; \emptyset \vdash [i \to \bullet](t_1\ t_2) : [i \to \bullet]T_2\&[i \to \bullet](\phi \cup \phi_1 \cup \phi_2)$ を得る。
最後の規則が T-Unit のとき、自明。
最後の規則が T-Loc のとき、$\Gamma; \Delta; \Sigma_1; \emptyset \vdash (i', l) : (T\ ref,\ i')\&\emptyset$ である。
$i' \neq i$ と $i' = i$ で場合分けする。
$i' \neq i$ のとき、$(i, l) = [i \to \bullet](i, l)$ であり、$[i \to \bullet]\Gamma; [i \to \bullet]\Delta; [i \to \bullet]\Sigma_1; \emptyset \vdash [i \to \bullet](i', l) : [i \to \bullet](T\ ref,\ i')\&\emptyset$ である。$i' = i$ のとき、$(\bullet, l) = [i \to \bullet](i, l)$ であり、T-Dead より、$[i \to \bullet]\Gamma; [i \to \bullet]\Delta; [i \to \bullet]\Sigma_1; \emptyset \vdash (\bullet, l) : [i \to \bullet](T\ ref,\ i')\&\emptyset$ である。
最後の規則が T-Dead のとき、自明。
最後の規則が T-Ref1 のとき、$\Gamma; \Delta, \rho; \Sigma_1; \emptyset \vdash ref\ t\ at\ \rho : (T\ ref,\ \rho)\&\phi \cup \{\rho\}$ である。定義より、$\Gamma; \Delta, \rho; \Sigma_1; \emptyset \vdash t : T\&\phi$ である。帰納法の仮定を適用して $[i \to \bullet]\Gamma; [i \to \bullet]\Delta, \rho; [i \to \bullet]\Sigma_1; \emptyset \vdash [i \to \bullet]t : [i \to \bullet]T\&[i \to \bullet]\phi$ である。T-Ref1 を適用して、$[i \to \bullet]\Gamma; [i \to \bullet]\Delta, \rho; [i \to \bullet]\Sigma_1; \emptyset \vdash [i \to \bullet](ref\ t\ at\ \rho) : [i \to \bullet](T\ ref,\ \rho)\&[i \to \bullet](\phi \cup \{\rho\})$ である。
最後の規則が T-Ref2 のとき、$\Gamma; \Delta, i'; \Sigma_1; \emptyset \vdash ref\ t\ at\ i' : (T\ ref,\ i')\&\phi \cup \{i'\}$ である。定義よ

り、$\Gamma;\Delta,i';\Sigma_1;\emptyset \vdash t : T\&\phi$、$i \notin PFPV(t)$ である。よって、$i' \neq i$ である。帰納法の仮定を適用して $[i \to \bullet]\Gamma;[i \to \bullet]\Delta,i';[i \to \bullet]\Sigma_1;\emptyset \vdash [i \to \bullet]t : [i \to \bullet]T\&[i \to \bullet]\phi$ である。T-Ref2 を適用して、$[i \to \bullet]\Gamma;[i \to \bullet]\Delta,i';[i \to \bullet]\Sigma_1;\emptyset \vdash [i \to \bullet](ref\ t\ at\ i') : [i \to \bullet](T\ ref,\ i')\&[i \to \bullet](\phi \cup \{i'\})$ である。
□

Lemma 2.11.
$\Gamma;\Delta,i,\Delta';\Sigma_1,i:\sigma;\emptyset \vdash t : T\&\phi$ かつ $i \notin FPV(t)$ ならば $\Gamma;\Delta,\Delta';\Sigma_1;\emptyset \vdash t : T\&\phi$

証明.
これもたぶんあってると思う。not yet
□

Lemma 2.12 (stack pop 補題)**.**
$\Gamma;\Delta,i;\Sigma_1,i : \sigma;\emptyset \vdash t : T\&\phi$ かつ $i \notin FPV(\Gamma,T)$ かつ $i \notin \phi$ かつ $FPV(\Sigma_1) \subseteq dom(\Sigma_1)$ ならば $\Gamma;\Delta;\Sigma_1;\emptyset \vdash [i \to \bullet]t : T\&\phi$

証明.
Lemma 2.9 より $i \notin PFPV(t)$。よって、Lemma 2.10 より $[i \to \bullet]\Gamma;[i \to \bullet]\Delta,i;[i \to \bullet]\Sigma_1,i : \sigma;\emptyset \vdash [i \to \bullet]t : [i \to \bullet]T\&[i \to \bullet]\phi$ を得る。Δ の要素は重複しない上、$i \notin FPV(\Gamma,T)$ かつ $i \notin \phi$ かつ $i \notin FPV(\Sigma_1)$ より、 $\Gamma;\Delta,i;\Sigma_1,i : [i \to \bullet]\sigma;\emptyset \vdash [i \to \bullet]t : T\&\phi$ を得る。$i \notin FPV([i \to \bullet]t)$ であるから、Lemma 2.11 より、 $\Gamma;\Delta;\Sigma_1;\emptyset \vdash [i \to \bullet]t : T\&\phi$ を得る。
ここの議論はちょっと怪しい Lemma 2.11 の代わりに
$\Gamma;\Delta;\Sigma_1,i : \sigma;\emptyset \vdash t : T\&\phi$ かつ $i \notin FPV(t)$ ならば $\Gamma;\Delta;\Sigma_1;\emptyset \vdash t : T\&\phi$ の補題がいえるべきか？わからん。Δ に何も仮定しなくてもいえるのか？
□

Lemma 2.13.
$\Gamma;\Delta,i;\Sigma_1;\Sigma_2 \vdash v : T\&\phi$ ならば $\Sigma_2 = \emptyset$ である。

Lemma 2.14 (標準形補題)**.**
v が $T_1 \to^{\phi} T_2$ 型をとる値ならば、$v = \lambda x : T_1.t$ である。v が $(T\ ref,\ i)$ 型をとる値ならば、$v = (i,l)$ である。v が $Unit$ 型をとる値ならば、$v = unit$ である。

Lemma 2.15 (代入補題)**.**
$\Gamma,x : T_1;\Delta;\Sigma_1;\emptyset \vdash t_1 : T_2\&\phi$ かつ $\Gamma;\Delta;\Sigma_1;\emptyset \vdash t_2 : T_1\&\emptyset$ ならば $\Gamma;\Delta;\Sigma_1;\emptyset \vdash [x \to t_2]t_1 : T_2\&\phi$

Theorem 2.16 (型保存定理)**.**
$\Sigma_1;\Sigma_2 \vdash t,M : T\&\phi$ かつ $t,M \to t',M'$ ならば
$\Sigma_1';\Sigma_2' \vdash t',M' : T\&\phi'$ かつ $\forall i.\Sigma_1(i) \subseteq \Sigma_1'(i)$ かつ $\phi' \subseteq \phi$ なる
$\Sigma_1',\Sigma_2',\phi'$ が存在する。

証明.
$t,M \to t',M'$ の導出に関する帰納法によって証明する。T-Conf の定義より、$\emptyset;\emptyset;\Sigma_1;\Sigma_2 \vdash t : T\&\phi$ かつ $\Sigma_1,\Sigma_2 \vdash M$ である。
最後に使われた規則によって場合分けする。
最後の規則が E-RefV のとき、$ref\ v\ at\ i,M \to (i,l),M'$ である。T-Ref2 の定義より、$\Gamma;\Delta,i;\Sigma_1,i : \sigma;\Sigma_2 \vdash v : T\&\phi$ かつ $T = (T'\ ref,i)$ かつ $\phi = \phi' \cup \{i\}$ をみたすような T',ϕ' が存在する。Lemma 2.13 より $\Sigma_2 = \emptyset$ である。よって、$\Sigma_1,i : \sigma \vdash M$ である。T-Stack2 の定義より、$M = (M'',i = \mu)$ となる M'',μ が存在する。$\Sigma' = \Sigma_1,i : (\sigma,l : T)$ とおく。T-Loc より、$\Gamma;\Delta,i;\Sigma';\emptyset \vdash (i,l) : (T'\ ref,i)\&\emptyset$ である。また、$\Sigma' \vdash M'',i = (\mu,l = v)$ であるので、T-Conf を適用して、$\Sigma;\vdash t',M'',i = (\mu,l = v) : T\&\emptyset$ である。
最後の規則が E-DerefLoc のとき、$!(i,l),M \to v,M$ である。E-DerefLoc の定義より、$M(i)(l) = v$ である。T-Deref の定義より、$\Gamma;\Delta;\Sigma_1;\Sigma_2 \vdash v : (T'\ ref,i)\&\phi'$ かつ $\phi = \phi' \cup \{i\}$ をみたすような T',ϕ' が存在する。 Lemma 2.13 より $\Sigma_2 = \emptyset$ である。T-Loc の定義より、 $\Sigma(i)(l) = T$ である。T-Stack2 の定義より、$\Sigma \vdash \mu : \sigma$ かつ $\Sigma(i) = \sigma, M(i) = \mu$ をみたすような σ,μ が存在する。T-Store の定義より、$\emptyset;\emptyset;\Sigma;\emptyset \vdash \mu : \sigma\&\emptyset$。よって、 T-Conf を適用して、$\Sigma_1;\emptyset \vdash v,M' : T\&\emptyset$ である。
最後の規則が E-Assign のとき、$(i,l) := v,M \to unit,[i \to [l \to v]M(i)]M$ である。T-Assign の定義

より、$\Gamma;\Delta;\Sigma_1;\Sigma_2 \vdash (i,l) : (T'\ ref,\ i)\&\phi_1$ $\Gamma;\Delta;\Sigma_1;\Sigma_3 \vdash v : T\&\phi_2$ かつ $\phi = \phi_1 \cup \phi_2 \cup \{i\}$ をみたすような T', ϕ_1, ϕ_2 が存在する。 Lemma 2.13 より $\Sigma_2 = \Sigma_3 = \emptyset$ である。T-Loc の定義より、 $\Sigma(i)(l) = T$ である。T-Stack2 の定義より、$\Sigma_1 \vdash \mu : \sigma$ かつ $\Sigma_1(i) = \sigma, M(i) = \mu$ をみたすような σ, μ が存在する。$\sigma' = [l \to T]\sigma$ と置く。$\Sigma_1' = [i \to \sigma']\Sigma_1$ と置くと、$\Sigma_1' = \Sigma_1$ である。store 型付けに対する代入補題より、$\Sigma_1 \vdash [l \to v]\mu : \sigma'$ である。T-Stack2 をどんどん適用して、 $\Sigma_1 \vdash [i \to [l \to v]\mu]M$ を得る。T-Unit を適用して、$\Gamma;\Delta;\Sigma_1;\emptyset \vdash unit : Unit\&\emptyset$ である。よって、 T-Conf を適用して、$\Sigma_1;\emptyset \vdash unit, M' : T\&\emptyset$ である。

最後の規則が E-Block のとき、$begin\ i\ t\ end, M \to begin\ i\ t'\ end, M'$ である。E-Block の定義より、$t, M \to t', M'$ である。T-Block2 の定義より、$\Gamma;\Delta;\Sigma_1, i : \sigma;\Sigma_2' \vdash t : T\&\phi'$ かつ $i \notin FPV(\Gamma, T)$ かつ $i \notin \Delta$ かつ $\phi = \phi' \setminus \{i\}$ をみたすような ϕ' が存在する。T-Conf を適用して、$\Sigma_1, i : \sigma, \Sigma_2; \vdash t, M : T\&\phi$ である。帰納法の仮定を適用して、 $\Sigma_1', i : \sigma', \Sigma_2'; \vdash t', M' : T\&\phi'$ かつ $\forall j.(\Sigma_1, i : \sigma)(j) \subseteq (\Sigma_1', i : \sigma')(j)$ かつ $\phi' \subseteq \phi$ なる $\Sigma_1', \sigma', \Sigma_2', \phi'$ が存在する。T-Conf の定義より、$\emptyset;\emptyset;\Sigma_1;\Sigma_2 \vdash t' : T\&\phi'$ かつ $\Sigma_1', i : \sigma', \Sigma_2'; \vdash M'$ である。T-Block2 を適用して、$\Gamma;\Delta;\Sigma_1';i : \sigma', \Sigma_2' \vdash begin\ i\ t'\ end : T\&\phi \setminus \{i\}$ T-Conf を適用して、$\Sigma_1', i : \sigma', \Sigma_2'; \vdash begin\ i\ t'\ end, M' : T\&\phi' \setminus \{i\}$ かつ $\forall j.(\Sigma_1, i : \sigma)(j) \subseteq (\Sigma_1', i : \sigma')(j)$ かつ $\phi' \setminus \{i\} \subseteq \phi \setminus \{i\}$ なる $\Sigma_1', \sigma', \Sigma_2', \phi'$ が存在する。

最後の規則が E-BlockBegin のとき、$begin\ \rho\ t\ end, M \to [\rho \to i]begin\ \rho\ t\ end, M$ である。E-BlockBegin の定義より、$i \notin dom(M)$ である。T-Block1 の定義より、$\Gamma;\Delta,\rho;\Sigma_1;\emptyset \vdash t : T\&\phi'$ かつ $\rho \notin FPV(\Gamma, T)$ かつ $\rho \notin \Delta$ かつ $\phi = \phi' \setminus \{\rho\}$ をみたすような ϕ' が存在する。stack push 補題より、$[\rho \to i]\Gamma; [\rho \to i]\Delta, \rho; [\rho \to i]\Sigma_1, i : \emptyset;\emptyset \vdash [\rho \to i]t : [\rho \to i]T\&[\rho \to i]\phi'$。T-Stack1, T-Stack2 の定義より、$FPV(\Sigma_1) \subseteq dom(\Sigma_1)$ であるから、$\rho \notin FPV(\Sigma_1)$ である。よって、Δ の要素は重複せず、$\rho \notin FPV(\Gamma, T)$ かつ $\rho \notin FPV(\Sigma_1)$ であるので、$\Gamma;\Delta, i;\Sigma_1, i : \emptyset;\emptyset \vdash [\rho \to i]t : T\&[\rho \to i]\phi'$ である。T-Block2 を適用して、$\Gamma;\Delta;\Sigma_1;i : \emptyset \vdash begin\ i\ [\rho \to i]t\ end : T\&([\rho \to i]\phi') \setminus \{i\}$ 、すなわち $\Gamma;\Delta;\Sigma_1;i : \emptyset \vdash [\rho \to i]begin\ \rho\ t\ end : T\&\phi$ を得る。T-Conf を適用して、$\Sigma_1, i : \sigma, \emptyset; \vdash [\rho \to i]begin\ \rho\ t\ end, M : T\&\phi$ である。T-Block2 を適用して、$\Gamma;\Delta;\Sigma_1;i : \emptyset \vdash begin\ i\ [\rho \to i]t\ end : T\&([\rho \to i]\phi') \setminus \{i\}$ 、すなわち $\Gamma;\Delta;\Sigma_1;i : \emptyset \vdash [\rho \to i]begin\ \rho\ t\ end : T\&\phi$ を得る。

最後の規則が E-BlockEnd のとき、$begin\ i\ v\ end, M \to [i \to \bullet]v, M'$ である。T-Block2 の定義より、$\Gamma;\Delta, i;\Sigma_1, i : \sigma;\Sigma_2 \vdash v : T\&\phi'$ かつ $i \notin FPV(\Gamma, T)$ かつ $i \notin \Delta$ かつ $\phi = \phi' \setminus \{i\}$ をみたすような ϕ' が存在する。T-Stack2 の定義より、$M = (M', i = \mu)$ となる M', μ が存在して、$FPV(\Sigma_1) \subseteq dom(\Sigma_1)$ かつ $\Sigma_1 \vdash M'$ である。stack pop 補題より、$\Gamma;\Delta;\Sigma_1;\emptyset \vdash [i \to \bullet]t : T\&\phi'$ である。よって、 T-Conf を適用して、$\Sigma_1;\emptyset \vdash [i \to \bullet]v, M' : T\&\phi'$ である。

最後の規則が E-App1 のとき、$t_1\ t_2, M \to t_1'\ t_2, M'$ である。E-App1 の定義より、$t_1, M \to t_1', M'$ である。T-App の定義より、$\Gamma;\Delta;\Sigma_1, \Sigma_3;\Sigma_2 \vdash t_1 : T_1 \to^{\phi'} T\&\phi_1$ かつ $\Gamma;\Delta;\Sigma_1;\Sigma_3 \vdash t_2 : T_1\&\phi_2$ かつ $\phi = \phi' \cup \phi_1 \cup \phi_2$ をみたすような ϕ', ϕ_1, ϕ_2 が存在する。T-Conf を適用して、$\Sigma_1, \Sigma_3;\Sigma_2 \vdash t_1, M : T_1 \to^{\phi'} T\&\phi_1$ である。帰納法の仮定を適用して、$\Sigma_1', \Sigma_3';\Sigma_2' \vdash t_1', M' : T_1 \to^{\phi'} T\&\phi_1'$ である。かつ $\forall j.(\Sigma_1, \Sigma_3)(j) \subseteq (\Sigma_1', \Sigma_3')(j)$ かつ $\phi_1' \subseteq \phi_1$ なる $\Sigma_1', \Sigma_2', \Sigma_3', \phi'$ が存在する。T-Conf の定義より、$\emptyset;\emptyset;\Sigma_1';\Sigma_3' \vdash t' : T_1 \to^{\phi'} T\&\phi_1'$ かつ $\Sigma_1', \Sigma_3';\Sigma_2' \vdash M'$ である。T-App を適用して、$\Gamma;\Delta;\Sigma_1';\Sigma_3', \Sigma_2' \vdash t_1'\ t_2 : T\&\phi' \cup \phi_1 \cup \phi_2$ T-Conf を適用して、$\Sigma_1', \Sigma_3';\Sigma_2' \vdash t_1'\ t_2, M' : T\&\phi' \cup \phi_1 \cup \phi_2$ である。

最後の規則が E-AppAbs のとき、$(\lambda x : T.\ t)\ v, M \to [x \to v]t, M$ である。T-App の定義より、$\Gamma;\Delta;\Sigma_1, \Sigma_3;\Sigma_2 \vdash (\lambda x : T_1.\ t) : T_1 \to^{\phi} T\&\emptyset$ かつ $\Gamma;\Delta;\Sigma_1;\Sigma_3 \vdash v : T_1\&\emptyset$ Lemma 2.13 より、$\Sigma_2 = \Sigma_3 = \emptyset$ である。T-Abs の定義より、$\Gamma, x : T_1;\Delta;\Sigma_1;\emptyset \vdash t : T\&\phi$ である。$\Gamma;\Delta;\Sigma_1;\emptyset \vdash [x \to v]t : T\&\phi$ である。よって、 T-Conf を適用して、$\Sigma_1;\emptyset \vdash [x \to v]t, M : T\&\phi$ である。 □

Theorem 2.17 (進行定理)**.**
$\Sigma_1;\Sigma_2 \vdash t, M : T\&\phi$ かつ $\rho, \bullet \notin \phi$ ならば t は値であるか、あるいは、$t, M \to t', M'$ を満たすような、t', M' が存在する。

証明.
$\emptyset;\emptyset;\Sigma_1;\Sigma_2 \vdash t : T\&\phi$ の導出に関する帰納法によって証明する。T-Conf の定義より、$\emptyset;\emptyset;\Sigma_1;\Sigma_2 \vdash t : T\&\phi$ かつ $\Sigma_1, \Sigma_2 \vdash M$ である。

最後に使われた規則によって場合分けする。

最後の規則が T-Var のとき、vacuously true で真。

最後の規則が T-Abs のとき、値であるから即座に真。

最後の規則が T-App のとき、帰納法の仮定を適用して t_1 は値であるか、あるいは、$t_1, M \to t_1', M'$ を

満たすような、t_1', M' が存在する。t_2 に対しても同様の命題が成り立つ。t_1 が簡約できる場合、E-App1 を適用して、t_1 が値で t_2 が簡約できる場、E-App2 を適用する。t_1 と t_2 がともに値の場合、標準形補題により、t_1 が $\lambda x : T_1.t$ の形をとることが分かり、E-AppAbs を適用できる。
最後の規則が T-Unit, T-Loc, T-Dead のとき、値であるから即座に真。
最後の規則が T-Ref1 のとき、$\rho \notin \phi$ であるから、vacuously true で真。
最後の規則が T-Ref2 のとき、帰納法の仮定を適用して、t_1 は値であるか、あるいは、$t_1, M \to t_1', M'$ を満たすような、t_1', M' が存在する。t_1 が値の場合、 $\Sigma_1, i : \sigma \vdash M$ であるから、$M = M', i = \mu$ という形をしている。よって、E-RefV が適用できる。 t_1 が値でない場合、E-Ref が適用できる。
最後の規則が T-Deref のとき、帰納法の仮定を適用して、t_1 は値であるか、あるいは、$t_1, M \to t_1', M'$ を満たすような、t_1', M' が存在する。t_1 が値のとき、標準形補題により t_1 は (i, l)。よって、T-Loc の定義より、$\Sigma_1(i)(l) = v$。よって、E-DerefLoc を適用する。t_1 が値でない場合、E-Deref を適用する。T-Assign の定義より、$\Gamma; \Delta; \Sigma_1; \Sigma_2 \vdash (i, l) : (T'\ ref,\ i) \& \phi_1\ \Gamma; \Delta; \Sigma_1; \Sigma_3 \vdash v : T \& \phi_2$ かつ $\phi = \phi_1 \cup \phi_2 \cup \{i\}$ をみたすような T', ϕ_1, ϕ_2 が存在する。帰納法の仮定を適用して t_1 は値であるか、あるいは、$t_1, M \to t_1', M'$ を満たすような、t_1', M' が存在する。t_1 が値のとき、標準形補題より (i, l)。帰納法の仮定を適用して t_2 は値であるか、あるいは、$t_2, M \to t_2', M'$ を満たすような、t_2', M' が存在する。t_1, t_2 がともに値の場合、E-Assign より、$(i, l) := v, M \to unit, [i \to [l \to v]M(i)]M$。$t_2$ が値でない場合、E-Assign2 より、$v := t_2, M \to v := t_2', M'$。$t_1$ が値でない場合、E-Assign1 より、$t_1 := t_2, M \to t_1' := t_2, M'$。
最後の規則が T-Block1 のとき、E-BlockBegin を適用する。
最後の規則が T-Block2 のとき、帰納法の仮定を適用して t_1 は値であるか、あるいは、$t_1, M \to t_1', M'$ を満たすような、t_1', M' が存在する。t_1 が値の場合、$\Sigma_1, i : \sigma \vdash M$ であるから、$M = M', i = \mu$ という形をしている。よって、E-BlockEnd が適用できる。t_1 が値でない場合、E-Block を適用する。 □

References

[1] Calcagno, Cristiano. Stratified operational semantics for safety and correctness of re- gion calculus. In ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), London, England, pages 155–165, 2001.

[2] Harper, Robert. A simplified account of polymorphic references, Information Pro- cessing Letters, 51(4):201-206, August 1994. See also (Harper, 1996).

[3] Tofte, Mads and Jean-Pierre Talpin. Region-based memory management. Information and Computation, 132(2):109–176, February 1997.

[4] Wright, Andrew K. and Matthias Feilleisen. A syntactic approach to type soundness. Information and Computation, 113.