Survey : Ambient Calculus

星名大輔

2000 年 2 月 17 日

要旨

今までの Ambient Calculus の発展の歴史と最近の状況を報告する。 Ambietn Calculus の大き流れとしては、 Cardelli 達によるものと Nielson 達によるものがあるが、今回は前者の論文を主に紹介する。

1 はじめに

1.1 歴史

1998	Mobile Ambients [3]
1999	Types for Mobile Ambients [4]
	Mobility Types for Mobile Ambients [2]
	Ambient Groups and Mobility Types [1]
	Equational Properties of mobile Ambients [8]
2000	Anytime,Anywhere. Modal Logic for Mobile Ambients [5]

まず、 98 年に [3] で、 Ambient Calculus が提案され、 99 年になって型システムが提案された（[4][2][1]）。 [1] での型システムがそれ以外（[4][2]）含んでいるため、 [1] を読めば十分であるが、 [1] は型の表記がごちゃごちゃしているため、個人的には [2] から読むのがいいと思う（[4] はつまらない）。これらの型システムと [8] については、 2 章で紹介する。また、 [5] は Sangiorgi による論文 [6] とともに 3 章で紹介する。

また、参考までに Nielson 達の論文を挙げると以下のようなものがある[1]。

1999	Validating Firewalls in Mobile Ambients [7]
	Flow Logics for Mobile Ambients [11]
	Abstract Interpretation of Mobile Ambients [10]
2000	Shape Analysis for Mobile Ambients [9]

1.2 Ambient とは？

- 計算の起こる named（名前の付いた）かつ bounded（境界を持っている）な space である。例えば、オブジェクト、ノートパソコン、 web ページなど。

[1] [7] を読んだのですが全然理解できなかったので、今回はこれらの論文については触れません。

- ambient が mobility の単位。
- ambient の中には subambient か process が存在する（階層構造を持っている）。

1.3 ambient calculus における空間の表現方法

下は ambient `Earth` を図で表したもの。

```
+-Earth-------------------------------------+
| +US-----------+ +EU----------------+      |
| |             | |+UK------+        |      |
| |    .....    | ||   ...  |   ...  | .....|
| |             | |+--------+        |      |
| +-------------+ +------------------+      |
+-------------------------------------------+
```

要するに、空間というのは階層構造になっていて、 ambient calculus はこの階層構造を表すことができる:

```
Earth[ US[..] | EU[ UK[..] | .. ] .. ]
```

1.4 ambient calculus における mobility とは？

それは、時間の進行に伴う空間的構造の変化である。例えば、

```
+a--------+ +b--+     +a--+          +b--+    +a--+ +b--------+
| +agent+ | |   |  -> |   | +agent+ |   | -> |   | | +agent+ |
| |     | | |   |     |   | |     | |   |    |   | | |     | |
| +-----+ | |   |     |   | +-----+ |   |    |   | | +-----+ |
+---------+ +---+     +---+          +---+    +---+ +---------+
```

これを ambient calculus で表すと:

```
a[agent[]]|b[]    ->    a[]|agent[]|b[]    ->    a[]|b[agent[]]
```

（本当は、上のような構造的変化は capability によって起る。）
つまり、 ambient calculus は空間的構造を表せるだけでなく、空間的構造の動的変化も表すことができる。

1.5 Process と Reduction の形式的な定義 [3]

1.5.1 Process

Restriction $(\nu n)P$: n は P の中でローカルな name であることを表す。これは、 P の外部に存在するどんな name とも n が異なることを意味する。

Inaction $\mathbf{0}$: 何もしないプロセス。

Parallel $P \mid Q$: プロセス P と Q が並列に存在している状態を表す。

Replication $!P$: iteration や recursion を表すために導入された term で、これは $P \mid !P$ というプロセスと同じと考えてよい。

Ambient $n[P]$: n という名の ambient を表す。また、この ambient の内部ではプロセス P が存在する。

capability action $M.P$: capability M を実行して P になるようなプロセス。

input action $(x).P$: 入力を受け付けて、それを自分の中の自由な変数 x に代入するようなプロセス。

output action $\langle M \rangle$: capability M を出力するようなプロセス。

1.5.2 Reduction

Entry Capability : $n[in\ m.P \mid Q] \mid m[R] \rightarrow m[n[P \mid Q] \mid R]$

```
                                        +m-----------------+
+n-------------+    +m--+               | +n--------+      |
|  in m.P | Q   | | | R |    ->         | |  P | Q  | | R  |
+--------------+    +---+               | +---------+      |
                                        +------------------+
```

Exit Capability : $n[m[out\ m.P \mid Q] \mid R] \rightarrow n[P \mid Q] \mid m[R]$

```
+m---------------------+
| +n------------+      |     +n--------+    +m--+
| | out m.P | Q | | R  |  -> |  P | Q  | |  | R |
| +-------------+      |     +---------+    +---+
+----------------------+
```

Open Capability : $open\ n.P \mid n[Q] \rightarrow P \mid Q$

```
             +n--+
open n.P  |  | Q |  ->  P | Q
             +---+
```

1.5.3 簡単な例：Firewrall Access

あかじめパスワードを知っている agent は firewall を越えられるという例。

$$Firewall \triangleq (\nu w)(k[in\ k.in\ w] \mid w[open\ k.P])$$
$$Agent \triangleq k[open\ k.Q]$$

$$
\begin{aligned}
&Agent \mid Firewall \\
&\equiv (\nu w)(k[open\ k.Q] \mid k[in\ k.in\ w] \mid w[open\ k.P]) \\
&\to^* (\nu w)(k[k[in\ w] \mid open\ k.Q] \mid w[open\ k.P]) \\
&\to^* (\nu w)(k[in\ w \mid Q] \mid w[open\ k.P]) \\
&\to^* (\nu w)(w[k[Q] \mid open\ k.P]) \\
&\to^* (\nu w)(w[Q \mid P])
\end{aligned}
$$

ambient w が firewall で、この ambient の name は restrict されていて（νw）、外部のプロセスはこの name を知ることができないため、w 内の入ることができない（つまり firewall を越えられない）。しかし、この例の場合、パスワード k を知っている agent（ambient）に関しては、$Firewall$ 内の $k[in\ k.in\ w]$ が w の中へ導いてくれて、その結果 $Agent$ は w 内に入ることができる（つまり firewall を越えられる）。

2 Cardelli による型システム [4][2][1] と Equational Theory[8]

型システムは [4][2][1] の順に発展した。[4][2] を中心に触れる。[1] については、[2] との相違を述べるだけにする。そして、最後に [8] を紹介する。

2.1 Simply-Typed Ambient Calculus[4]

2.1.1 目的

- $(in\ n)[P]$、$n.P$、$\langle in\ (in\ n)\rangle$ というような意味のないプロセスを排除する。
- 受信する側の型と送信する側の型が一致することを保証する。例えば、$a[(x).P \mid open\ b] \mid b[in\ a.\langle 3\rangle]$ というプロセスは reduction の結果 $a[P\{x \leftarrow 3\}]$ となるが、x は Int 型であることを保証したい。

2.1.2 型についての説明

- $P : T$
  型 T のメッセージを送受信するようなプロセスを表す。
- $n : Amb[T]$
  n は内部で型 T のメッセージが送受信が行われるような ambient。例えば、$n[(x : int).P \mid \langle 3\rangle]$ なら n は型 $Amb[int]$ を持つ。
- $M : Cap[T]$
  この capability M は実行されると、型が T であるようなプロセスになることを意味する。つまり、$M.P$ とあれば、P の型は T。

2.2 Mobility Types[2]

2.2.1 目的

先ほどの型システムを拡張して、型に次の 2 つの情報を加える。

mobile,immobile reduction の間、 capability *in*,*out* によって移動する ambient は *mobile* と呼び、そうでないものを *immobile* と呼ぶ。

lock,unlock reduction の間、 capability *open* によって、 open されるものを *unlock* と呼び、そうでないものを *lock* と呼ぶ。

2.2.2 型についての説明

- $n : Amb^Y[^Z T] : Y$ は mobile($\rightsquigarrow$) または immobile($\vee$)、 Z は lock($\circ$) または unlock($\bullet$) を表す。それ以外は前回と同様。
- $M : Cap[^Z T] : Z$ は lock($\circ$) または unlock($\bullet$) を表す。それ以外は前回と同様。

immobile と宣言した ambient は決して移動することはないし、 *lock* と宣言した ambient は決して open されることはない。例えば、 $n : Amb^Y[^\vee T]$ は capability *in*,*out* によって移動することがない ambient であり、 $m : Amb^\bullet[^Z T]$ は
capability *open* によって、 open されることがない ambient である。

2.2.3 簡単な例

$E \vdash a : Amb^\bullet[^\vee Shh]$、 $E \vdash p : Amb^\circ[^\rightsquigarrow W]$、 $E \vdash b : Amb^\bullet[^\rightsquigarrow W]$、 $E \vdash M : W$、 $E, x : W \vdash P : {}^\rightsquigarrow W$ という条件の下で、

$$E \vdash a[p[out\ a.in\ b.\langle M \rangle]] \mid b[open\ p.(x : W).P] : {}^\vee Shh$$

が導ける。つまり、 a は *immobile* かつ *lock*、 p は *mobile* かつ *unlock*、 b は *mobile* かつ *lock* である時、上のプロセスの型付けが可能である。

ここで、 b の中に capability *in*,*out* がないのに、 *mobile* になっている理由は、 p が b の中へ入ってきた場合、 p が open されるので、 p 内の *mobile* なプロセスが b 内の存在し得るためである（つまり、中で mobile な ambient が open されるような ambient は mobile でなければならない）。

2.3 Group Types[1]

Cardelli による型システムの一連の発展は、この論文で終了する。 group という概念を導入した。各 ambient はいずれかの group に属していて、「ある group に属する ambient に対してしか in/out できない」とか、「ある group に属する ambient に対してしか open できない」ということを型によって保証できる。

2.4 Equational Theory[8]

どのような ambient（プロセス）同士を等しいと見なすか（振る舞いが等しいと見なすか）ということを形式的に定義し、具体的に 2 つの ambient について等しいことを証明する際のテクニックを紹介した論文。この論文は数学的な話ばかりで、ひたすら、定義／定理／証明の繰り返しなので、つまらないと思う人が多いと思う。

2.4.1 2 つのプロセスの振る舞いが等しいとは？

P を何回かリダクションした結果、 P のトップレベルに n という名前の ambient が観測されることを $P \Downarrow n$ と書くとする。フォーマルな定義をすると、

$P \downarrow n \stackrel{\text{def}}{=} P$ が $(\nu \vec{m})(n[P'] \mid P'')$ と表せる（ただし、 $n \notin \{\vec{m}\}$ ）。
$P \Downarrow n \stackrel{\text{def}}{=} P \downarrow n$ または $P \rightarrow Q \wedge Q \Downarrow n$

以上のような準備の下で、「2 つのプロセスの振る舞いが等しさ」を以下のように定義する（Contextual Equivalence と呼ばれている）。

$P \simeq Q \stackrel{\text{def}}{=}$ 任意のコンテキスト $\mathcal{C}()$ について、 $\mathcal{C}(P) \Downarrow n \Leftrightarrow \mathcal{C}(Q) \Downarrow n$

つまり、 2 つのプロセスが、任意のコンテキストに置かれた時、コンテキスと含めた全体について、互いに同じ名前の ambient が観測されたら、その 2 つのプロセスの振る舞いは同じとみなす、ということを意味している。

例えば、直観的に考えて、以下の等式は成り立つだろう。

- $(\nu n)(n[] \mid open\ n.P) \simeq P$ （ただし、 $n \notin fn(P)$ ）
- $(\nu n)n[P] \simeq \mathbf{0}$ （ただし、 $n \notin fn(P)$ ）

これらのような等式でも成立することを実際に証明するのは大変で、この論文にあるテクニックを使えば、少し楽になる。

2.4.2 Contextual Equivalence の問題点とその解決策

問題点は以下の 2 つである。

- 任意のコンテキストを考えなければならないこと。
- $P \Downarrow n$ の定義を見ると、 $P \rightarrow Q$ とあるが、これは任意の structural congruent なプロセスを考えなくてはならない（リダクションの定義を参照）。

これに対して、

- 任意のコンテキストを考えるのではなく、 Harness という制限されたコンテキストを考える。
- $P \Downarrow n$ の代わりに Hardening Relation というものを、 reduction の代わりに labelled transition を考える。

全体の特徴として、 2 つのプロセスが与えられた時、そのプロセスの syntactic な構造から、 contextual に equivalent かどうかを調べることができる。

3 付録： POPL'00[5][6]

今年の POPL には Ambient 関連の論文が 3 本 [5][6][9] 載っていた。そのうち 2 本を紹介する。

3.1 Modal Logic for Mobile Ambients[5]

Ambient Calculus のための modal かつ spatial な logic を作ったという話。このような logic の持つ利点としては、

> Ambient Calculus の configuration の様々な性質を表現できる。例えば、今までの型システムでは表現できなかった
>
> - いつか agent は firewall を越える。
> - どこかに n という名の agent が 1 つある。
>
> というような性質を表現できる。

また、その結果 Ambient Calculus の model checking に利用できる。

3.2 Logical Formulas

この logic 特有の論理式をいくつか紹介する。

- $\mathbf{T}$: true を表す。
- $\mathbf{0}$: 何もないことを表す。
- $\mathcal{A} \mid \mathcal{B}$: $\mathcal{A}$ を満たすプロセスと $\mathcal{B}$ を満たすプロセスが 2 つ並列にあることを表す。
- $\Diamond\mathcal{A}$: いつか $\mathcal{A}$ を満たすことを表す。
- $\oplus\mathcal{A}$: どこかに $\mathcal{A}$ を満たすプロセスがあることを表す。

3.3 簡単な例

- どんなプロセスも $\mathbf{T}$ を満たす。プロセス 0 は $\mathbf{0}$ を満たす。
- P が $\mathcal{A}$ を満たし、 Q が $\mathcal{B}$ を満たすなら、 $P \mid Q$ は $\mathcal{A} \mid \mathcal{B}$ を満たす。
- P が $\mathcal{A}$ を満たすなら、 $\mathit{open}\ n.0 \mid n[P]$ は $\Diamond(\mathbf{0} \mid \mathcal{A})$ を満たす。
（何故なら $\mathit{open}\ n.0 \mid n[P]$ は reduction の結果 $0 \mid P$ となるから）
- P が $\mathcal{A}$ を満たすなら、 $n[P] \mid Q$ は $\oplus\mathcal{A}$ を満たす。
- $m \neq n$ ならば、プロセス $m[0] \mid n[0]$ は $\exists x.x[\mathbf{T}] \mid x[\mathbf{T}]$ を満たさない。

3.4 Sangiorgi の Mobile Safe Ambient[6]

movement primitive（capability）を改良することによって、 interference を取り除き、その結果、 equational theory が構築できたという話。 interference とは、例えば、

(1) $n[\mathit{in}\ m.P] \mid m[Q] \mid m[R]$
(2) $\mathit{open}\ n.P \mid \mathit{open}\ n.Q \mid n[R]$

のように redex がかぶってることである。これらのような非決定性は並列プログラミングにおいては必要であるが、 Ambient Calculus では、もっと致命的な interference が存在する。例えば、

(4) $open\ n \mid n[in\ m.P] \mid m[Q]$
(5) $h[n[in\ m.P \mid out\ h.R] \mid m[Q]]$
(6) $h[P] \mid n[in\ h \mid m[out\ n.Q]]$

の場合、 (4) では $open\ n$ を先に実行することによって、 $in\ n$ が実行できなくなる。逆に $in\ m$ を先に実行すると $open\ n$ が実行できなくなる。 (5)(6) でも同様の問題が発生する。著者らは、このような interference はプログラミングエラーと見なし、型システムによって、これらを取り除いた。

3.4.1 アイデア

- 1 つの ambient の中に capability を実行できるプロセスは高々 1 つ。
  $\Longrightarrow$ これによって、 (5) は排除される。
- coaction （$\overline{in}\ n, \overline{out}\ n, \overline{open}\ n$）の導入。 reduction は

  $$\begin{aligned} &n[in\ m.P_1 \mid P_2] \mid m[\overline{in}\ n.Q_1 \mid Q_2] \rightarrow m[n[P_1 \mid P_2] \mid Q_1 \mid Q_2] \\ &m[out\ m.P_1 \mid P_2] \mid \overline{out}\ m.Q_1 \mid Q_2] \rightarrow n[P_1 \mid P_2] \mid m[Q_1 \mid Q_2] \\ &open\ n.P \mid n[\overline{open}\ n.Q_1 \mid Q_2] \rightarrow P \mid Q_1 \mid Q_2 \end{aligned}$$

  と定義される。これによって、例えば (4) は

  $$\begin{aligned} &open\ n \mid n[\overline{open}\ n.in\ m.P] \mid m[\overline{in}\ m.Q] \\ &open\ n \mid n[in\ m.\overline{open}\ n.P] \mid m[\overline{in}\ m.Q] \end{aligned}$$

  のどちらかになる。両方とも $open\ n$ か $in\ m$ のどちらかは残るが、どちらが残るか分からないという非決定性は消える（interference がなくなる）。

3.4.2 Equational Theory

以上のアイデアより、以下が成立する。

$$\begin{aligned} &n[in\ m.P_1 \mid P_2] \mid m[\overline{in}\ m.Q_1 \mid Q_2] \simeq m[n[P_1 \mid P_2] \mid Q_1 \mid Q_2] \\ &m[n[out\ m.P_1 \mid P_2] \mid \overline{out}\ m.Q_1 \mid Q_2] \simeq n[P_1 \mid P_2] \mid m[Q_1 \mid Q_2] \\ &open\ n.P \mid n[\overline{open}\ n.Q_1 \mid Q_2] \simeq P \mid Q_1 \mid Q_2 \end{aligned}$$

以前の Ambient Calculus では

(1) $n[in\ m.P_1 \mid P_2] \mid m[Q] \simeq m[n[P_1 \mid P_2] \mid Q]$
(2) $m[n[out\ m.P_1 \mid P_2] \mid Q] \simeq n[P_1 \mid P_2] \mid m[Q]$
(3) $open\ n.P \mid n[Q] \simeq P \mid Q$

は成立しない。何故なら、 (1) では n に in するプロセスが外部にある場合（つまり、 $\mathcal{C}() = k[in\ n.0] \mid ()$ というコンテキストを考えた時）、左辺では k は観測されないが、右辺では k は観測される。すなわち、左辺と右辺は contextual に equivalent でない。

参考文献

[1] Ghelli G. Gordon A.D Cardelli, L. Ambient groups and mobility types.

[2] Ghelli G. Gordon A.D Cardelli, L. Mobility types for mobile ambients. In *ICALP'99*, pages 230–239. Springer, 1999. LNCS 1644.

[3] Gordon A.D Cardelli, L. Mobile ambients. In *FoSSaCS'98*, pages 140–155. Springer, 1998. LNCS 1378.

[4] Gordon A.D Cardelli, L. Types for mobile ambients. In *POPL'99*, pages 79–92, 1999.

[5] Gordon A.D Cardelli, L. Anytime,anywhere. modal logic for mobile ambients. In *POPL'00*, 2000.

[6] D. Sangiorgi F. Levi. Controlling interference in ambients. In *POPL'00*, 2000.

[7] R.R.Hansen J.G.Jensen F.Nielson, H.R.Nielson. Validating firewalls in mobile ambients. In *CONCUR'99*, pages 463–477, 1999.

[8] Cardelli L Gordon, A.D. Equational properties of mobile ambients. In *FoSSaCS'99*. Springer, 1999. LNCS 1578.

[9] F.Nielson H.R.Nielson. Shape analysis for mobile ambients. In *POPL'00*, 2000.

[10] F.Nielson H.R.Nielson R.R.Hansen, J.G.Jensen. Abstract interpretation of mobile ambients. In *SAS'99*, pages 134–148, 1999.

[11] J.G.Jensen R.R.Hansen. Flow logics for mobile ambients. Master's thesis, Aarhus University, 1999.