The Non-deterministic Catch and Throw Mechanism and Its Subject Reduction Property

Hiroshi Nakano

Department of Applied Mathematics and Informatics,
Ryukoku University, Seta, Otsu, 520-21, Japan
nakano@rins.ryukoku.ac.jp

Abstract. A simple programming language and its typing system is introduced to capture the catch and throw mechanism with its non-deterministic feature. The subject reduction property of the system, which compensates for the unpleasant feature of the non-determinism, is shown.

1 Introduction

The catch and throw mechanism is a programming facility for non-local exit which plays an important role when programmers handle exceptional situations. In a previous paper [4], the author showed that the catch/throw mechanism corresponds to a variant formulation of Genzen's LJ following the Curry-Howard isomorphism in the opposite direction, and gave a realizability interpretation to the formal system by an abstract stack machine, in which the computational behavior of the mechanism was treated by a fixed evaluation strategy, and therefore the result of evaluation was unique. However, generally, the catch/throw mechanism introduces a non-determinism to evaluation processes, that is, the result of evaluation depends on the evaluation strategy. For example, let M be a term defined by

$$M = \mathbf{catch}\ u\ ((\lambda\, x.\, \lambda\, y.\, 1)\, (\mathbf{throw}\ u\ 2)\, (\mathbf{throw}\ u\ 3)).$$

There are three possible results for the evaluation of M depending on the evaluation strategy.

In this paper, we first extend the language to capture the non-deterministic feature of the catch/throw mechanism, and introduce its typing system. We next show the subject reduction property of the system.

2 A programming language with catch/throw

We first introduce a programming language based on λ-calculus. The language has the catch and throw mechanism.

2.1 Syntax

Constants and variables. We first assume the following disjoint sets of individual constants, individual variables and tag variables are given.

C_i A set of individual constants $c, d, \ldots$.
V_i A countably infinite set of individual variables $x, y, z, \ldots$.
V_t A countably infinite set of tag variables $u, v, w, \ldots$.

Tag variables are called tags.

Terms. The set of *terms* E are defined as follows:

$$\begin{array}{rl} E ::= & C_i \mid V_i \mid \mathbf{catch}\, V_t\, E \mid \mathbf{throw}\, V_t\, E \\ & \mid \lambda V_i.\, E \mid E\, E \mid \kappa V_t.\, E \mid E\, V_t \\ & \mid {<}E,\, E{>} \mid \mathbf{proj_1}\, E \mid \mathbf{proj_2}\, E \\ & \mid \mathbf{inj_1}\, E \mid \mathbf{inj_2}\, E \mid \mathbf{case}\, E\ V_i.\, E\ V_i.\, E\ . \end{array}$$

Example 1.

$$\lambda x.\, \mathbf{case}\ x\ y.(\mathbf{inj_2}\, y)\ z.(\mathbf{inj_1}\, z)$$
$$\mathbf{catch}\ u\ ((\kappa v.\, \mathbf{proj_1} {<}x,\ \mathbf{throw}\ v\ y{>})\, u)$$

We use $M, N, \ldots$ to denote terms. The terms $\kappa V_t.\, E$ and $E\, V_t$ are used to denote a tag-abstraction and a tag-instantiation, respectively, c.f. [4]. Free and bound occurrences of variables are defined in the standard manner. We regard a tag variable u as bound in **catch** $u\ M$ and $\kappa u.\, M$. We also define the alpha-convertibility in the standard manner where we admit renaming of bound tag variable as well as bound individual variables. Hereafter, we treat terms modulo this alpha-convertibility. A term M so represents an equivalence class of terms which are alpha-convertible to M. We denote the set of individual and tag variables occurring freely in M by $FIV(M)$ and $FTV(M)$, respectively.

Definition 1 (Substitution). Let M, $N_1, \ldots, N_n$ be terms, and let $x_1, \ldots, x_n$ be individual variables. We use $M[N_1/x_1, \ldots, N_n/x_n]$ to denote the term obtained from M by replacing all free occurrences of $x_1, \ldots, x_n$ by $N_1, \ldots, N_n$, respectively. $M[v_1/u_1, \ldots, v_n/u_n]$ is defined similarly, where $u_1, \ldots, u_n$ and $v_1, \ldots,$ v_n are tag variables.

2.2 Operational semantics

Now we define an operational semantics of the language by a set of reduction rules on terms. The non-deterministic feature of the catch/throw mechanism is introduced by the following rule.

Definition 2 ($\underset{t}{\mapsto}$). A relation $\underset{t}{\mapsto}$ on terms is defined as follows:

$$M[\mathbf{throw}\ u\ N/x] \underset{t}{\mapsto} \mathbf{throw}\ u\ N \qquad (x \in FIV(M),\, x \not\equiv M)\,.$$

Example 2.

$$
\begin{aligned}
\texttt{<}\mathbf{inj_1}\,(\mathbf{throw}\;u\;M),\;\mathbf{throw}\;v\;N\texttt{>} &\underset{\mathrm{t}}{\mapsto} \mathbf{throw}\;u\;M\\
\texttt{<}\mathbf{inj_1}\,(\mathbf{throw}\;u\;M),\;\mathbf{throw}\;v\;N\texttt{>} &\underset{\mathrm{t}}{\mapsto} \mathbf{throw}\;v\;N\\
\mathbf{throw}\;u\;M &\underset{\mathrm{t}}{\not\mapsto} \mathbf{throw}\;u\;M\\
\mathbf{case}\;z\;x.(\mathbf{throw}\;u\;x)\;y.y &\underset{\mathrm{t}}{\not\mapsto} \mathbf{throw}\;u\;x\\
\mathbf{catch}\;u\;(\mathbf{throw}\;u\;M) &\underset{\mathrm{t}}{\not\mapsto} \mathbf{throw}\;u\;M\\
\mathbf{catch}\;v\;(\mathbf{throw}\;u\;(M\,v)) &\underset{\mathrm{t}}{\not\mapsto} \mathbf{throw}\;u\;(M\,v)
\end{aligned}
$$

The rest is defined by the following rules.

Definition 3 ($\underset{\mathrm{n}}{\mapsto}$). A relation $\underset{\mathrm{n}}{\mapsto}$ on terms is defined as follows:

$$
\begin{aligned}
\mathbf{catch}\;u\;M &\underset{\mathrm{n}}{\mapsto} M && (u \notin FTV(M))\\
\mathbf{catch}\;u\;(\mathbf{throw}\;u\;M) &\underset{\mathrm{n}}{\mapsto} M && (u \notin FTV(M))\\
(\lambda\,x.\,M)\,N &\underset{\mathrm{n}}{\mapsto} M[N/x]\\
(\kappa\,u.\,M)\,v &\underset{\mathrm{n}}{\mapsto} M[v/u]\\
\mathbf{proj_1}\,\texttt{<}M,\;N\texttt{>} &\underset{\mathrm{n}}{\mapsto} M\\
\mathbf{proj_2}\,\texttt{<}M,\;N\texttt{>} &\underset{\mathrm{n}}{\mapsto} N\\
\mathbf{case}\;(\mathbf{inj_1}\,L)\;x.M\;y.N &\underset{\mathrm{n}}{\mapsto} M[L/x]\\
\mathbf{case}\;(\mathbf{inj_2}\,L)\;x.M\;y.N &\underset{\mathrm{n}}{\mapsto} N[L/y]
\end{aligned}
$$

Definition 4 (Reduction rules). We define a relation, denoted by $\mapsto$, by the union of $\underset{\mathrm{t}}{\mapsto}$ and $\underset{\mathrm{n}}{\mapsto}$, that is,

$$M \mapsto N \quad \text{iff} \quad M \underset{\mathrm{t}}{\mapsto} N \text{ or } M \underset{\mathrm{n}}{\mapsto} N.$$

Definition 5 ($\to$). We define a relation, denoted by $\to$, as follows: $M \to N$ if and only if N is obtained from M by replacing an occurrence of M' in M by N' such that $M' \mapsto N'$. Let $\overset{*}{\to}$ be the transitive and reflexive closure of the relation $\to$.

Example 3. Let 1, 2 and 3 be distinct individual constants, and let M be as $M = \mathbf{catch}\;u\;((\lambda\,x.\,\lambda\,y.\,1)\,(\mathbf{throw}\;u\;2)\,(\mathbf{throw}\;u\;3))$.

$$
\begin{aligned}
&M \to \mathbf{catch}\;u\;((\lambda\,y.\,1)\,(\mathbf{throw}\;u\;3)) \to \mathbf{catch}\;u\;1 \to 1\\
&M \to \mathbf{catch}\;u\;(\mathbf{throw}\;u\;2) \to 2\\
&M \to \mathbf{catch}\;u\;(\mathbf{throw}\;u\;3) \to 3
\end{aligned}
$$

3 A typing system

We now introduce a typing system for the programming language.

3.1 Syntax of typing judgements

Type expressions. Type expressions of our typing system consist of atomic types, conjunctions $(A \wedge B)$, disjunctions $(A \vee B)$, implications $(A \supset B)$ and exceptions $(A \triangleleft B)$. The last one is introduced to handle the catch/throw mechanism and represents another kind of disjunction (c.f. [4]).

Individual contexts. An *individual context* is a finite mapping which assigns a type expression to each individual variable in its domain. We use $\Gamma, \Gamma', \ldots$ to denote individual contexts, and denote the domain of an individual context Γ by $Dom(\Gamma)$. Let $A_1, \ldots, A_n$ be type expressions, and $x_1, \ldots, x_n$ individual variables such that if $x_i = x_j$ then $A_i = A_j$ for any i and j. We use $\{x_1 : A_1, \ldots, x_n : A_n\}$ to denote an individual context whose domain is $\{x_1, \ldots, x_n\}$ and which assigns A_i to x_i for each i.

Tag contexts. An *tag context* is a finite mapping which assigns a pair of a type expression and a set of individual variables to each tag variable in its domain. We use $\Delta, \Delta', \ldots$ to denote tag contexts. Let $u_1, \ldots, u_n$ be tag variables. Let $B_1, \ldots, B_n$ be type expressions, and let $V_1, \ldots, V_n$ be sets of individual variables such that if $u_i = u_j$ then $B_i = B_j$ and $V_i = V_j$ for any i and j. We use $\{u_1 : B_1^{V_1}, \ldots, u_n : B_n^{V_n}\}$ to denote a tag context whose domain is $\{u_1, \ldots, u_n\}$ and which assigns the pair (B_i, V_i) to u_i for each i. We denote the first and the second components of $\Delta(u)$ by $\Delta^t(u)$ and $\Delta^v(u)$, respectively. For example, $\Delta^t(u_i) = B_i$ and $\Delta^v(u_i) = V_i$ if $\Delta = \{u_1 : B_1^{V_1}, \ldots, u_n : B_n^{V_n}\}$.

Definition 6 (Compatible contexts). Let Γ and Γ' be individual contexts. Γ is *compatible* with Γ' if and only if $\Gamma(x) = \Gamma'(x)$ for any individual variable $x \in Dom(\Gamma) \cap Dom(\Gamma')$. We denote it by $\Gamma \parallel \Gamma'$. Note that $\Gamma \cup \Gamma'$ is also an individual context if $\Gamma \parallel \Gamma'$. The compatibility of tag contexts is also defined as follows: Δ is *compatible* with Δ' if and only if $\Delta^t(u) = \Delta'^t(u)$ for any individual variable $u \in Dom(\Delta) \cap Dom(\Delta')$. We denote it by $\Delta \parallel \Delta'$. When Δ and Δ' are compatible, we define a new tag context $\Delta \sqcup \Delta'$ as follows.

$$(\Delta \sqcup \Delta')(u) = \begin{cases} (\Delta^t(u),\ \Delta^v(u) \cup \Delta'^v(u)) & \text{if } u \in Dom(\Delta) \cap Dom(\Delta') \\ \Delta(u) & \text{if } u \in Dom(\Delta) \text{ and } u \notin Dom(\Delta') \\ \Delta'(u) & \text{if } u \notin Dom(\Delta) \text{ and } u \in Dom(\Delta') \end{cases}$$

Note that $Dom(\Delta \sqcup \Delta') = Dom(\Delta) \cup Dom(\Delta')$.

Definition 7. Let Δ be as $\Delta = \{u_1 : B_1^{V_1}, \ldots, u_n : B_n^{V_n}\}$, and let u and v be tag variables. If $\{u, v\} \subset Dom(\Delta)$ implies $\Delta^t(u) = \Delta^t(v)$, then we define a tag context $\Delta[v/u]$ as follows.

$$\Delta[v/u] = \{u_1[v/u] : B_1^{V_1}, \ldots, u_n[v/u] : B_n^{V_n}\}.$$

We define $\Gamma[y/x]$ similarly for an individual context Γ and individual variables x and y.

Definition 8. Let V be a set of individual variables. We define a tag context $\Delta[V/\{x\}]$ as follows.

$$\begin{aligned} &Dom(\Delta[V/\{x\}]) = Dom(\Delta) \\ &\Delta[V/\{x\}]^t(u) = \Delta^t(u) \\ &\Delta[V/\{x\}]^v(u) = \begin{cases} (\Delta^v(u) - \{x\}) \cup V & \text{if } x \in \Delta^v(u) \\ \Delta^v(u) & \text{otherwise} \end{cases} \end{aligned}$$

Typing judgement. Let Γ and Δ be an individual context and a tag context, respectively, such that $\Delta^v(u) \subset Dom(\Gamma)$ for any $u \in Dom(\Delta)$. Let M be a term, and C a type expression. *Typing judgements* have the following form.

$$\Gamma \vdash M : C\, ;\ \Delta$$

The intended meaning of a typing judgement $\{x_1 : A_1, \ldots, x_m : A_m\} \vdash M : C\, ;$ $\{u_1 : B_1^{V_1}, \ldots, u_n : B_n^{V_n}\}$ is roughly that when we execute the program M supplying values of the types $A_1 \ldots A_m$ for the corresponding free variables $x_1, \ldots, x_m$ of M, it normally reduces to a value of the type C, otherwise the program throws a value of B_j with a tag u_j for some j $(1 \le j \le n)$, and the thrown value depends on only the individual variables which belong to V_j.

3.2 $L_{c/t}$

We denote the typing system by $L_{c/t}$, which can be considered as a natural-deduction-style reformulation of the logical system presented in [4]. We can see a more direct correspondence between proofs and programs in $L_{c/t}$.

Definition 9 (Typing rules). $L_{c/t}$ is defined by the following set of typing rules.

$$\frac{}{\Gamma \cup \{x : A\} \vdash x : A\, ;\ \Delta}\ (var) \qquad \frac{\Gamma \vdash M : A\, ;\ \Delta \sqcup \{u : A^V\}}{\Gamma \vdash \mathbf{catch}\ u\ M : A\, ;\ \Delta}\ (catch)$$

$$\frac{\Gamma_1 \vdash M : E\, ;\ \Delta}{\Gamma_1 \cup \Gamma_2 \vdash \mathbf{throw}\ u\ M : A\, ;\ \Delta \sqcup \{u : E^{Dom(\Gamma_1)}\}}\ (throw)$$

$$\frac{\Gamma \cup \{x : A\} \vdash M : B\, ;\ \Delta}{\Gamma \vdash \lambda x.\, M : A \supset B\, ;\ \Delta}\ (\supset\text{-I}) \quad (x \notin \Delta^v(u) \text{ for any } u \in Dom(\Delta))$$

$$\frac{\Gamma_1 \vdash M : A \supset B\, ;\ \Delta_1 \quad \Gamma_2 \vdash N : A\, ;\ \Delta_2}{\Gamma_1 \cup \Gamma_2 \vdash M\, N : B\, ;\ \Delta_1 \sqcup \Delta_2}\ (\supset\text{-E})$$

$$\frac{\Gamma \vdash M : A\, ;\ \Delta \sqcup \{u : E^V\}}{\Gamma \vdash \kappa u.\, M : A \triangleleft E\, ;\ \Delta}\ (\triangleleft\text{-I}) \qquad \frac{\Gamma_1 \vdash M : A \triangleleft E\, ;\ \Delta}{\Gamma_1 \cup \Gamma_2 \vdash M\, u : A\, ;\ \Delta \sqcup \{u : E^{Dom(\Gamma_1)}\}}\ (\triangleleft\text{-E})$$

$$\frac{\Gamma_1 \vdash M : A\,;\ \Delta_1 \quad \Gamma_2 \vdash N : B\,;\ \Delta_2}{\Gamma_1 \cup \Gamma_2 \vdash \mathord{<}M,\ N\mathord{>} : A \wedge B\,;\ \Delta_1 \sqcup \Delta_2}\ (\wedge\text{-I})$$

$$\frac{\Gamma \vdash M : A \wedge B\,;\ \Delta}{\Gamma \vdash \mathbf{proj_1}\ M : A\,;\ \Delta}\ (\wedge_1\text{-E}) \qquad \frac{\Gamma \vdash M : A \wedge B\,;\ \Delta}{\Gamma \vdash \mathbf{proj_2}\ M : B\,;\ \Delta}\ (\wedge_2\text{-E})$$

$$\frac{\Gamma \vdash M : A\,;\ \Delta}{\Gamma \vdash \mathbf{inj_1}\ M : A \vee B\,;\ \Delta}\ (\vee_1\text{-I}) \qquad \frac{\Gamma \vdash M : B\,;\ \Delta}{\Gamma \vdash \mathbf{inj_2}\ M : A \vee B\,;\ \Delta}\ (\vee_2\text{-I})$$

$$\frac{\Gamma_1 \vdash L : A \vee B\,;\ \Delta_1 \quad \Gamma_2 \cup \{x : A\} \vdash M : C\,;\ \Delta_2 \quad \Gamma_3 \cup \{y : B\} \vdash N : C\,;\ \Delta_3}{\begin{array}{l}\Gamma_1 \cup \Gamma_2 \cup \Gamma_3 \vdash \mathbf{case}\ L\ x.M\ y.N : C\,;\\ \qquad \Delta_1 \sqcup \Delta_2[Dom(\Gamma_1)/\{x\}] \sqcup \Delta_3[Dom(\Gamma_1)/\{y\}]\end{array}}\ (\vee\text{-E})$$

The side condition for ($\supset$-I) is necessary to keep the system constructive. Note that the following inference rule of [4] corresponds to ($\supset$-I) of $L_{c/t}$.

$$\frac{\Gamma\ A \to B\,;}{\Gamma \to A \supset B\,;}\ (\to\supset)$$

A natural translation of this rule into $L_{c/t}$ would be as follows.

$$\frac{\Gamma \cup \{x : A\} \vdash M : B\,;\ \{\}}{\Gamma \vdash \lambda x.\, M : A \supset B\,;\ \{\}}\ (\supset\text{-I})'$$

As a logic, $(\supset\text{-I})'$ is equivalent to ($\supset$-I) of Definition 9, but is too restrictive with respect to the variation of proofs, i.e., typed programs. For example, the following typing judgement, which is derivable in $L_{c/t}$, would not be derivable if we replaced ($\supset$-I) by $(\supset\text{-I})'$.

$$\{\} \vdash \mathbf{catch}\ u\ (\lambda x.\, \mathbf{throw}\ u\ (\lambda y.\, y)) : A \supset A\,;\ \{\}$$

Moreover, the language would not have a subject reduction property, because

$$\{\} \vdash \mathbf{catch}\ u\ ((\lambda z.\, \lambda x.\, z)\,(\mathbf{throw}\ u\ (\lambda y.\, y))) : A \supset A\,;\ \{\}$$

would be still derivable, but

$$\mathbf{catch}\ u\ ((\lambda z.\, \lambda x.\, z)\,(\mathbf{throw}\ u\ (\lambda y.\, y))) \to \mathbf{catch}\ u\ (\lambda x.\, \mathbf{throw}\ u\ (\lambda y.\, y)).$$

This is the reason why we maintain the set of the relevant individual variables to each tag in tag contexts of typing judgements.

The following example of a derivation shows that the programming language does not have Church-Rosser property even if we consider only the well-typed terms. Let M be the term $\lambda x.\, \lambda f.\, \mathbf{catch}\ u\ ((\lambda y.\, x)\,(\mathbf{throw}\ u\ (f\, x)))$. The well-typed term M has two normal forms as follows.

$$M \to \lambda x.\, \lambda f.\, \mathbf{catch}\ u\ (\mathbf{throw}\ u\ (f\, x)) \to \lambda x.\, \lambda f.\, f\, x$$
$$M \to \lambda x.\, \lambda f.\, \mathbf{catch}\ u\ x \to \lambda x.\, \lambda f.\, x$$

Example 4. Let Γ be as $\Gamma = \{x : A, f : A \supset A\}$.

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\{y : B\} \vdash x : A\,;\ \{\}}\,(var)}{\{\} \vdash \lambda y.\, x : B \supset A\,;\ \{\}}\,(\supset\text{-I}) \qquad \dfrac{\dfrac{\dfrac{}{\Gamma \vdash f : A \supset A\,;\ \{\}}\,(var) \quad \dfrac{}{\Gamma \vdash x : A\,;\ \{\}}\,(var)}{\Gamma \vdash f\,x : A\,;\ \{\}}\,(\supset\text{-E})}{\Gamma \vdash \mathbf{throw}\ u\ (f\,x) : B\,;\ \{u : A^{\{x,f\}}\}}\,(throw)}{\Gamma \vdash (\lambda y.\, x)\,(\mathbf{throw}\ u\ (f\,x)) : A\,;\ \{u : A^{\{x,f\}}\}}\,(\supset\text{-E})}{\Gamma \vdash \mathbf{catch}\ u\ ((\lambda y.\, x)\,(\mathbf{throw}\ u\ (f\,x))) : A\,;\ \{\}}\,(catch)}{\{x : A\} \vdash \lambda f.\, \mathbf{catch}\ u\ ((\lambda y.\, x)\,(\mathbf{throw}\ u\ (f\,x))) : (A \supset A) \supset A\,;\ \{\}}\,(\supset\text{-I})}{\{\} \vdash \lambda x.\, \lambda f.\, \mathbf{catch}\ u\ ((\lambda y.\, x)\,(\mathbf{throw}\ u\ (f\,x))) : A \supset (A \supset A) \supset A\,;\ \{\}}\,(\supset\text{-I})
$$

3.3 Basic properties of $L_{c/t}$

In this subsection, we presents a some basic properties of the system as a preparation for proving the subject reduction property of $L_{c/t}$.

Proposition 10. *If $\Gamma \vdash M : C\,;\ \Delta$ is derivable, then $FIV(M) \subset Dom(\Gamma)$ and $FTV(M) \subset Dom(\Delta)$.*

Proof. By induction on the derivation of $\Gamma \vdash M : C\,;\ \Delta$. □

Definition 11. Let Δ and Δ' be tag contexts. We define a relation $\Delta \sqsubset \Delta'$ as follows. The relation $\Delta \sqsubset \Delta'$ holds if and only if

- $\Delta \parallel \Delta'$,
- $Dom(\Delta) \subset Dom(\Delta')$, and
- $\Delta^v(u) \subset \Delta'^v(u)$ for any $u \in Dom(\Delta)$.

Note that $\Delta \sqsubset (\Delta \sqcup \Delta')$ if $\Delta \parallel \Delta'$.

Definition 12. Let d be a natural number. We say a typing judgement is *d-derivable* if there exists a derivation of the judgement whose depth is less than or equal to d.

Proposition 13. *Let d be a natural number, and let $\Gamma \vdash M : C\,;\ \Delta$ be a d-derivable typing judgement.*

1. *If $\Gamma \subset \Gamma'$ and $\Delta \sqsubset \Delta'$, then $\Gamma' \vdash M : C\,;\ \Delta'$ is also d-derivable.*
2. *If $\Gamma[y/x]$ is well defined, then $\Gamma[y/x] \vdash M[y/x] : C\,;\ \Delta[\{y\}/\{x\}]$ is also d-derivable.*
3. *If $\Delta[v/u]$ is well defined, then $\Gamma \vdash M[v/u] : C\,;\ \Delta[v/u]$ is also d-derivable.*

Proof. By simultaneous inductions on d. □

Proposition 14. *Let x and u be as $x \notin FIV(M)$ and $u \notin FTV(M)$.*

1. *If $\Gamma \cup \{x : A\} \vdash M : C\,;\ \Delta$ is derivable, then $\Gamma \vdash M : C\,;\ \Delta$ is also derivable.*
2. *If $\Gamma \vdash M : C\,;\ \Delta \sqcup \{u : E^V\}$ is derivable, then $\Gamma \vdash M : C\,;\ \Delta$ is also derivable.*

Proof. Straightforward induction on the derivations. □

Proposition 15. *Let M be term, and let u be a tag variable. If $\Gamma \vdash$ **throw** u $M : C\,;\, \Delta$ is derivable, then $\Gamma \vdash$ **throw** u $M : A\,;\, \Delta$ is also derivable for any type A.*

Proof. Since $\Gamma \vdash$ **throw** u $M : C\,;\, \Delta$ is derivable, so is $\Gamma \vdash M : E\,;\, \Delta'$ for some E and Δ' such that $\Delta = \Delta' \sqcup \{u : E^{Dom(\Gamma)}\}$. Therefore, we can derive $\Gamma \vdash$ **throw** u $M : A;\ \Delta$ for any A by $(throw)$. □

Proposition 16 (Substitution). *Let Γ_1, Γ_2, Δ_1 and Δ_2 be as $\Gamma_1 \parallel \Gamma_2$ and $\Delta_1 \parallel \Delta_2$. If $\Gamma_1 \vdash N : A;\ \Delta_1$ and $\Gamma_2 \cup \{x : A\} \vdash M : C\,;\ \Delta_2$ are derivable, then $\Gamma_1 \cup \Gamma_2 \vdash M[N/x] : C\,;\ \Delta_1 \sqcup \Delta_2[Dom(\Gamma_1)/\{x\}]$ is also derivable.*

Proof. By induction on the depth of the derivation of $\Gamma_2 \cup \{x : A\} \vdash M : C\,;\ \Delta_2$. Suppose that $\Gamma_1 \vdash N : A;\ \Delta_1$ and $\Gamma_2 \cup \{x : A\} \vdash M : C\,;\ \Delta_2$ are derivable. By cases on the last rule used in the derivation of $\Gamma_2 \cup \{x : A\} \vdash M : C\,;\ \Delta_2$.

Case 1: The last rule is (var). That is, $M = y$ for some individual variable y such that $\{y : C\} \subset \Gamma_2 \cup \{x : A\}$. If $M = x$, then we can derive $\Gamma_1 \cup \Gamma_2 \vdash M[N/x] : C\,; \Delta_1 \sqcup \Delta_2[Dom(\Gamma_1)/\{x\}]$ by applying Proposition 13 to the derivation of $\Gamma_1 \vdash N : A;\ \Delta_1$ since $M[N/x] = N$ and $C = A$ in this case. If $M \neq x$, then we can derive it by (var) since $M[N/x] = y$ and $\{y : C\} \subset \Gamma_2$ in this case.

Case 2: The last rule is $(catch)$. In this case, $M =$ **catch** u M' and the following judgement is derivable for some u, V and M'.

$$\Gamma_2 \cup \{x : A\} \vdash M' : C\,;\ \Delta_2 \sqcup \{u : C^V\}$$

We can assume that $u \notin Dom(\Delta_1)$ by Proposition 13. By the induction hypothesis, we have a derivation of

$$\Gamma_1 \cup \Gamma_2 \vdash M'[N/x] : C\,;\ \Delta_1 \sqcup (\Delta_2 \sqcup \{u : C^V\})[Dom(\Gamma_1)/\{x\}]. \qquad (1)$$

Since $u \notin Dom(\Delta_1)$, we get $M[N/x] =$ **catch** u $(M'[N/x])$. By applying $(catch)$ to (1), we get $\Gamma_1 \cup \Gamma_2 \vdash M[N/x] : C\,;\ \Delta_1 \sqcup \Delta_2[Dom(\Gamma_1)/\{x\}]$.

Case 3: The last rule is $(throw)$. In this case, $M =$ **throw** u M' and the following judgement is derivable for some u, M', E, Γ_2' and Δ such that $\Gamma_2' \subset \Gamma_2 \cup \{x : A\}$ and $\Delta_2 = \Delta \sqcup \{u : E^{Dom(\Gamma_2') \cup \{x\}}\}$.

$$\Gamma_2' \vdash M' : E\,;\ \Delta$$

Let Γ be as $\Gamma = \Gamma_2' - \{x : A\}$. Note that $\Gamma \subset \Gamma_2$ and $\Gamma_2' \subset \Gamma \cup \{x : A\}$. Therefore, by Proposition 13,

$$\Gamma \cup \{x : A\} \vdash M' : E\,;\ \Delta.$$

By the induction hypothesis, we have a derivation of

$$\Gamma_1 \cup \Gamma \vdash M'[N/x] : E\,;\ \Delta_1 \sqcup \Delta[Dom(\Gamma_1)/\{x\}].$$

Since $M[N/x] = \textbf{throw}\ u\ (M'[N/x])$, by applying $(throw)$,

$$\Gamma_1 \cup \Gamma \vdash M[N/x] : C\ ;\ \Delta_1 \sqcup \Delta[Dom(\Gamma_1)/\{x\}] \sqcup \{u : E^{Dom(\Gamma_1 \cup \Gamma)}\}.$$

Since $\Gamma \subset \Gamma_2$, by Proposition 13 again,

$$\Gamma_1 \cup \Gamma_2 \vdash M[N/x] : C\ ;\ \Delta_1 \sqcup \Delta[Dom(\Gamma_1)/\{x\}] \sqcup \{u : E^{Dom(\Gamma_1 \cup \Gamma)}\}.$$

Note that $\Delta[Dom(\Gamma_1)/\{x\}] \sqcup \{u : E^{Dom(\Gamma_1 \cup \Gamma)}\} = \Delta_2[Dom(\Gamma_1)/\{x\}]$ because $\Delta_2 = \Delta \sqcup \{u : E^{Dom(\Gamma_2') \cup \{x\}}\}$ and $x \notin Dom(\Gamma)$.

Case 4: The last rule is $(\supset$*-I*$)$. In this case $M = \lambda y. M'$, $C = C_1 \supset C_2$ and the following judgement is derivable for some y, C_1, C_2 and M' such that $y \notin \Delta_2^v(u)$ for any $u \in Dom(\Delta_2)$.

$$\Gamma_2 \cup \{x : A\} \cup \{y : C_1\} \vdash M' : C_2\ ;\ \Delta_2$$

We can assume that $y \notin Dom(\Gamma_1)$ by Proposition 13, and get $M'[N/x] = \lambda y. (M[N/x])$. By the induction hypothesis, we have a derivation of

$$\Gamma_1 \cup \Gamma_2 \cup \{y : C_1\} \vdash M'[N/x] : C_2\ ;\ \Delta_1 \sqcup \Delta_2[Dom(\Gamma_1)/\{x\}]. \qquad (2)$$

Since $y \notin \Delta_2^v(u)$ for any $u \in Dom(\Delta_2)$ and $y \notin Dom(\Gamma_1)$, we get $y \notin (\Delta_1 \sqcup \Delta_2[Dom(\Gamma_1)/\{x\}])^v(u)$ for any $u \in Dom(\Delta_1 \sqcup \Delta_2[Dom(\Gamma_1)/\{x\}])$. Therefore we can derive $\Gamma_1 \cup \Gamma_2 \vdash \lambda y. (M'[N/x]) : C_2\ ;\ \Delta_1 \sqcup \Delta_2[Dom(\Gamma_1)/\{x\}]$ by applying $(\supset$-I$)$ to (2).

Case 5: The last rule is one of others. Similar. □

4 The subject reduction property of $L_{c/t}$

As mentioned in Section 3.2, the language does not have Church-Rosser property even if we consider only the well-typed terms. However, it has the subject reduction property, which compensates for this unpleasant feature. In this section, we show the subject reduction property of $L_{c/t}$.

Lemma 17. *If* $\Gamma \vdash M : C\ ;\ \Delta$ *is derivable and* $M \underset{t}{\mapsto} \textbf{throw}\ v\ N$*, then* $\Gamma \vdash \textbf{throw}\ v\ N : C\ ;\ \Delta$ *is also derivable.*

Proof. By induction on the depth of the derivation of $\Gamma \vdash M : C\ ;\ \Delta$. Suppose that $\Gamma \vdash M : C\ ;\ \Delta$ is derivable and $M \underset{t}{\mapsto} \textbf{throw}\ v\ N$. By Proposition 15, it is enough to show that $\Gamma \vdash \textbf{throw}\ v\ N : C'\ ;\ \Delta$ is derivable for some C'. By cases according to the last rules used in the derivation.

Case 1: The last rule is (var). This is impossible because $M \underset{t}{\mapsto} \textbf{throw}\ v\ N$.

Case 2: The last rule is (*catch*). $M = \textbf{catch}\ u\ M'$ and the following judgement is derivable for some u, V and M'.

$$\Gamma \vdash M' : C\ ;\ \Delta \sqcup \{u : C^V\} \tag{3}$$

We can assume that $u \notin FTV(\textbf{throw}\ v\ N)$ by Proposition 13, and get $M' = \textbf{throw}\ v\ N$ or $M' \underset{t}{\mapsto} \textbf{throw}\ v\ N$ from $M \underset{t}{\mapsto} \textbf{throw}\ v\ N$. Therefore, from (3) or the induction hypothesis on (3),

$$\Gamma \vdash \textbf{throw}\ v\ N : C\ ;\ \Delta \sqcup \{u : C^V\}.$$

We get $\Gamma \vdash \textbf{throw}\ v\ N : C\ ;\ \Delta$ by Proposition 14 since $u \notin FTV(\textbf{throw}\ v\ N)$.

Case 3: The last rule is (*throw*). In this case, $M = \textbf{throw}\ u\ M'$ and the following judgement is derivable for some u, M', E, Γ' and Δ' such that $\Gamma' \subset \Gamma$ and $\Delta = \Delta' \sqcup \{u : E^{Dom(\Gamma')}\}$.

$$\Gamma' \vdash M' : E\ ;\ \Delta' \tag{4}$$

We get $M' = \textbf{throw}\ v\ N$ or $M' \underset{t}{\mapsto} \textbf{throw}\ v\ N$ from $M \underset{t}{\mapsto} \textbf{throw}\ v\ N$. Therefore, from (4) or the induction hypothesis on (4),

$$\Gamma' \vdash \textbf{throw}\ v\ N : E\ ;\ \Delta'.$$

We get $\Gamma \vdash \textbf{throw}\ v\ N : E\ ;\ \Delta$ by Proposition 13 since $\Gamma' \subset \Gamma$ and $\Delta' \sqsubset \Delta$.

Case 4: The last rule is ($\supset$-*I*). $M = \lambda x. M'$, $C = C_1 \supset C_2$ and the following judgement is derivable for some x, C_1, C_2 and M' such that $x \notin \Delta^v(u)$ for any $u \in Dom(\Delta)$.

$$\Gamma \cup \{x : C_1\} \vdash M' : C_2\ ;\ \Delta \tag{5}$$

We can assume that $x \notin FIV(\textbf{throw}\ v\ N)$ by Proposition 13, and get $M' = \textbf{throw}\ v\ N$ or $M' \underset{t}{\mapsto} \textbf{throw}\ v\ N$ from $M \underset{t}{\mapsto} \textbf{throw}\ v\ N$. Therefore, from (5) or the induction hypothesis on (5),

$$\Gamma \cup \{x : C_1\} \vdash \textbf{throw}\ v\ N : C_2\ ;\ \Delta.$$

We get $\Gamma \vdash \textbf{throw}\ v\ N : C_2\ ;\ \Delta$ by Proposition 14 since $x \notin FIV(\textbf{throw}\ v\ N)$.

Case 5: The last rule is one of others. Similar to Case 2 and Case 3. □

Lemma 18. *If* $\Gamma \vdash M : C\ ;\ \Delta$ *is derivable and* $M \underset{n}{\mapsto} N$, *then* $\Gamma \vdash N : C\ ;\ \Delta$ *is also derivable.*

Proof. By induction on the depth of the derivation of $\Gamma \vdash M : C\ ;\ \Delta$. □

Suppose that $\Gamma \vdash M : C\ ;\ \Delta$ is derivable and $M \underset{n}{\mapsto} N$. By cases according to the form of M.

Case 1: $M = \textbf{catch}\ u\ N$ *and* $u \notin FTV(N)$. In this case, $\Gamma \vdash N : C\ ; \Delta \sqcup \{u : C^V\}$ is derivable for some V. We get $\Gamma \vdash N : C\ ;\ \Delta$ by Proposition 14 since $u \notin FTV(N)$.

Case 2: $M = \textbf{catch}\ u\ (\textbf{throw}\ u\ N)$ and $u \notin FTV(N)$. The following judgement is derivable for some V, Γ' and Δ' such that $\Gamma' \subset \Gamma$ and $\Delta \sqcup \{u : C^V\} = \Delta' \sqcup \{u : C^{Dom(\Gamma')}\}$.

$$\Gamma' \vdash N : C\ ;\ \Delta'$$

Since $\Gamma' \subset \Gamma$ and $\Delta' \sqsubset \Delta \sqcup \{u : C^V\}$, $\Gamma \vdash N : C\ ;\ \Delta \sqcup \{u : C^V\}$ is derivable by Proposition 13. Therefore, $\Gamma \vdash N : C\ ;\ \Delta$ is also derivable by Proposition 14 since $u \notin FTV(N)$.

Case 3: $M = (\lambda\, x.\, M_1)\, M_2$ and $N = M_1[M_2/x]$ for some x, M_1 and M_2. The following two judgements are derivable for some A and $x \notin \Delta^v(u)$ for any $u \in Dom(\Delta)$.

$$\Gamma \cup \{y : A\} \vdash M_1 : C\ ;\ \Delta \qquad (6)$$

$$\Gamma \vdash M_2 : A\ ;\ \Delta \qquad (7)$$

We get $\Gamma \vdash M_1[M_2/x] : C\ ;\ \Delta[Dom(\Gamma)/\{x\}]$ from (6) and (7) by Lemma 16, where $\Delta[Dom(\Gamma)/\{x\}] = \Delta$ since $x \notin \Delta^v(u)$ for any $u \in Dom(\Delta)$.

Case 4: $M = (\kappa\, u.\, M')\, v$ and $N = M'[v/u]$ for some u, v and M'. The following judgement is derivable for some E, Γ', Δ' and V such that $\Gamma' \subset \Gamma$ and $\Delta = \Delta' \sqcup \{v : E^{Dom(\Gamma')}\}$.

$$\Gamma' \vdash M' : C\ ;\ \Delta' \sqcup \{u : E^V\}$$

Since $\Delta' \parallel \{v : E^{Dom(\Gamma')}\}$, $\Gamma' \vdash M'[v/u] : C\ ;\ \Delta'[v/u] \cup \{v : E^V\}$ is derivable by Proposition 13. Since $\Gamma' \subset \Gamma$, by Proposition 13 again,

$$\Gamma \vdash M'[v/u] : C\ ;\ \Delta'[v/u] \cup \{v : E^V\}.$$

Since $V \subset Dom(\Gamma')$,

$$\Delta'[v/u] \sqcup \{v : E^V\} \sqsubset \Delta'[v/u] \sqcup \{v : E^{Dom(\Gamma')}\} \sqsubset \Delta' \sqcup \{v : E^{Dom(\Gamma')}\} = \Delta.$$

Therefore, $\Gamma \vdash M'[v/u] : C\ ;\ \Delta$ is derivable by Proposition 13.

Case 5: $M = \textbf{proj}_i <M_1,\ M_2>$ and $N = M_i$ for some i $(i = 1, 2)$. Similar.

Case 6: $M = \textbf{case}\ (\textbf{inj}_i\, M_0)\ x_1.M_1\ x_2.M_2$ and $N = M_i[M_0/x_i]$ for some i $(i = 1, 2)$. Similar. □

Lemma 19. *If $\Gamma \vdash M : C\ ;\ \Delta$ is derivable and $M \mapsto N$, then $\Gamma \vdash N : C\ ;\ \Delta$ is also derivable.*

Proof. Straightforward from Lemma 17 and Lemma 18. □

Theorem 20 (Subject reduction). *If $\Gamma \vdash M : C\ ;\ \Delta$ is derivable and $M \to N$, then $\Gamma \vdash N : C\ ;\ \Delta$ is also derivable.*

Proof. By induction on the depth of the derivation of $\Gamma \vdash M : C \,;\, \Delta$. Suppose that $\Gamma \vdash M : C \,;\, \Delta$ is derivable and $M \to N$. If $M \mapsto N$, then trivial by Lemma 19. Therefore we can assume that $M \to N$ and $M \not\mapsto N$. By cases according to the last rules used in the derivation. A typical one is the case that the last rule is $(throw)$. In this case, $M = \textbf{throw}\ u\ M'$ and

$$\Gamma' \vdash M' : E \,;\, \Delta'$$

is derivable for some u, M', E, Γ' and Δ' such that $\Gamma' \subset \Gamma$ and $\Delta = \Delta' \sqcup \{u : E^{Dom(\Gamma')}\}$. Since $M \to N$ and $M \not\mapsto N$, $M' \to N'$ and $N = \textbf{throw}\ u\ N'$ for some N'. Therefore, $\Gamma' \vdash N' : E \,;\, \Delta'$ is derivable by the induction hypothesis. We get $\Gamma \vdash \textbf{throw}\ u\ N' : E \,;\, \Delta$ by applying $(throw)$. The proofs for other cases are just similar. □

5 Concluding remarks

We have presented a programming language and its typing system which capture the non-deterministic feature of the catch/throw mechanism. We have shown that the system has subject reduction property, which compensates for the unpleasant feature of the non-determinism.

There remain some problems which should be considered. Two major ones are (1) semantics, especially realizability interpretations, of typing judgements, and (2) normalizability, especially strong normalizability, of well-typed terms. The subject reduction property is a good news to these problems, but both are still open.

References

1. M. Felleisen, D. Friedman, E. Kohlbecker, and B. Duba, A syntactic theory of sequential control, Theoretical Computer Science **52**(1987) 205-237.
2. T. G. Griffin, A formulae-as-types notion of control, Conf. Rec. ACM Symp. on Principles of Programming Languages (1990) 47-58.
3. C. R. Murthy, An evaluation semantics for classical proofs, Proc. the 6th Annual IEEE Symp. on Logic in Computer Science (1991) 96-107.
4. H. Nakano, A Constructive Formalization of the Catch and Throw Mechanism, Proc. the 7th Annual IEEE Symp. on Logic in Computer Science (1992) 82-89.
5. G. D. Plotkin, Call-by-name, call-by-value and the λ-calculus, Theoretical Computer Science **1**(1975) 125-159.