Proving Type Safety for Java Simple Closures

Marco Bellia and M. Eugenia Occhiuto

Dipartimento di Informatica, Università di Pisa, Italy {*bellia,occhiuto*}*@di.unipi.it*

Abstract. In the last years, the Java community has been arguing about adding closures to Java in order to improve expressivity. The debate has not yet terminated but all proposals seem to converge towards a notion of Simple Closures which contain only essential features. This paper addresses the problem of defining a rigorous semantics for Simple Closures. The technique adopted is well known and has already been used to prove interesting properties of the language. A minimal calculus is defined: Featherweight Java extended with Simple Closures. Syntax and semantics of such calculus are defined and type safety and backward compatibility are proved.

1 Introduction

In the last few years extensions to Java focus on higher order mechanisms to enhance expressivity, conciseness, good structuring, reusability, and factoring of code [OW97,Mei01,McC01,Set03,Cor04,Mic04,BO05,Bri05,GHJV05,SS07]. Proposals to add closures in Java have been discussed since 2006 [LLB06,CS06,BGGA06]. Recent revisions [GA09,Col09,Rei09] of the proposals agree on Simple Closures. A Simple Closure is a value that abstracts an arbitrary Java code and makes such a value available for assignment, parameter transmission, value returning and invocation. However, all the proposals fail to provide a formal semantics for closures. Such problem is addressed in this paper, resorting to a minimal calculus: Featherweight Java (FJ for short).

FJ was presented at the 1999 ACM Symposium [IPW99] as a minimal core calculus for a formal model to study design properties of Java. The idea is to omit most of the concrete features of the full language to concentrate on a small core language, fully significant, for all the relevant aspects of the properties under investigation. Initially, it was introduced to design and formalize the generic type system of Java [BOSW98] and prove its soundness [IPW01]. More recently [IP02], it has been used *i)* to define a reduction semantics of inner classes in Java, *ii)* in the investigation of properties of inner classes, including abstraction features and interaction with inheritance, and *iii)* to provide for compilation issues of inner classes.

In this paper, we start using the variant of FJ, called FGJ, which deals with generic types and extends it to model all essential features of Java involved in the properties of closures we want to investigate. The new calculus Featherweight Generic Java with Closures is here called FGCJ. The structure of closures, we consider, is the one of Simple Closures, hereafter called closures, [Col09,Rei09]. In particular, closures *i)* require to extend the type system with closure types and *ii)* are first class values which can be bound to parameters, hence passed to methods or other closures. The reduction semantics is then used to prove two fundamental properties which are type soundness, which asserts consistency between the type system and computation, and backward compatibility which asserts the consistency between old rules and extended rules.

The paper is organized as follows: Section 2 resumes FGJ features, Section 3 defines syntax and semantics of FGCJ, Section 4 states and prove the properties, eventually Section 5 concludes the paper and Appendix A contains proof details.

2 Featherweight Java

A program in FJ [IPW99] consists of a declaration of generic class definitions and of an expression to be evaluated. The expression corresponds to the body of the 0-arguments main method of Java. Here is a declaration for some typical class definitions in FGJ .

```
class Pair⟨X ◁ Object, Y ◁ Object⟩ ◁ Object{
  X fst;Y snd;
  Pair(X fst, Y snd){super(); this.fst=fst; this.snd=snd; };
  Pair(X fst, Y snd){ super(); this.fst=fst; this.snd=snd; };
  ⟨Z ◁ Object⟩ Pair⟨Z,Y⟩ setfst(Z newfst){
    return new Pair⟨Z,Y⟩(newfst,this.snd); }}
class A ◁Object{A(){super();};}
class B ◁Object{B(){super();};}
```

A complete definition of the syntax of FGJ consists of the grammar rules in **Table 1** that are labelled by the defined grammatical category indexed by FGJ. Symbol ◁ is a notational shorthand for Java keyword `extends`. For syntactic regularity, (a) classes always specify the super class, possibly `Object`, and have exactly one constructor definition; (b) class constructors have one parameter for each class field with the same name as the field, invoke the super constructor on the fields of the super class and initialize the remaining fields to the corresponding parameters; (c) field access always specifies the receiver (object), possibly `this`. This results in the stylized form of the constructors in the example above. Both classes and methods may have generic type parameters. In the example, `X` and `Y` are type parameters of the class `Pair`, while `Z` is type parameter of method `setfst`. Each type parameter has a *bound*, in the example `Object` for all of them.

FGJ has not side effects. Hence, *sequencing* and *assignment* are strictly confined to constructor bodies. In particular, method bodies have always the form `return`, followed by an expression, as in the body of `setfst` in the example. The lack of Java constructs for sequencing control and for store updating (along with that of concurrency, and reflection) is the main advantage of the calculus in studying language properties that are non affected by side effects. In this way the calculus is, as much as possible, compact and takes advantage of the referential transparency. The latter one provides a simple reduction semantics which is crucial for rigorous, ease to derive, proofs of the language properties [Fri87]. About compactness, FGJ has only five forms of expressions: One for *Object Creation*, as `new Pair(...)` in the body of `setfst` in the example, another for variable (namely, *parameter naming*), as `newfst` and `this`, one for *field access*, as `this.snd` always in the body of `setfst`. From a syntactic point of view, `this` is a keyword in Java, and is a variable in FGJ, however, in both languages, it has the semantics of *object self-reference*, see rule `GR-Invk`. The remaining two forms are *method invocation* and *cast* as in the expression below.

```
(e)   ((Pair) new Pair<A,B>(new A(),new B())).setfst<B>(new B())
```

The presence of *cast* in FGJ is justified from its fundamental role in compiling generic classes and could be ruled out of FGCJ. We conclude this presentation, by considering the twofold role of referential transparency: first evaluation is entirely formalized within the syntax of FGJ (hence, the evaluation process results in a sequence of FGJ expressions reducing the first one to the last one, if any, which represents an error or its value), second the order in which expressions are reduced, if more than one can be selected, does not affect the final result. The reduction semantics of FGJ consists of the first three rules that appear in **Table 2: Computation** and deal with term evaluation, and of the first five rules in **Table 2: Congruence** that deal with redex selection. The remaining 18 rules of the semantics of FGJ deal with the type system and with term well-formedness. The rules of FGJ have labels that are indexed by FGJ in **Table 2, 4, 5**. As an example of computation, expression `e`, evaluated in the context of the declaration of the classes `Pair`, `A` and `B` (namely, the evaluation of the FGJ program constituted from the declaration plus expression `e`), results in the sequence:

```
((Pair) new Pair<A,B>(new A(),new B())).setfst<B>(new B())
(new Pair<A,B>(new A(),new B())).setfst<B>(new B())          by GR-CAST_FGJ
new Pair<B,B>(new B(),new B())                               by GR-INVK_FGJ
```

3 Featherweight GCJ

3.1 Syntax

The syntax of closures is very close to [GA09] and allows to define closures (*lambda expressions*) and closure types (*function types*). *Lambda expressions* consist of closures whose body is an expression and closures whose body is a statement. In FGCJ, since sequencing and assignment are omitted, the body of a closure can only be an expression. The syntax for closures is $\#(\overline{\mathtt{T}}\ \overline{\mathtt{x}})\mathtt{e}$, where $\overline{\mathtt{x}}$ are the formal names, $\overline{\mathtt{T}}$ are the formal types and `e` is the body. [1]

Table 1

Syntax

$\mathtt{T} ::= \mathtt{X} \mid \mathtt{N} \mid$	$(\mathtt{T}_{\mathrm{FGJ}})$
$\quad \mid \#\mathtt{T}(\overline{\mathtt{T}})$	$(\mathtt{T}_{\mathrm{FGCJ}})$
$\mathtt{N} ::= \mathtt{C}\langle\overline{\mathtt{T}}\rangle$	$(\mathtt{N}_{\mathrm{FGJ}})$
$\mathtt{L} ::= \mathtt{class}\ \mathtt{C}\langle\overline{\mathtt{X}} \lhd \overline{\mathtt{N}}\rangle \lhd \mathtt{N}\{\overline{\mathtt{T}}\ \overline{\mathtt{f}}; \mathtt{K}\ \overline{\mathtt{M}}\}$	$(\mathtt{L}_{\mathrm{FGJ}})$
$\mathtt{K} ::= \mathtt{C}(\overline{\mathtt{T}}\ \overline{\mathtt{f}})\{\mathtt{super}(\overline{\mathtt{f}}); \mathtt{this}.\overline{\mathtt{f}} = \overline{\mathtt{f}};\}$	$(\mathtt{K}_{\mathrm{FGJ}})$
$\mathtt{M} ::= \langle\overline{\mathtt{X}} \lhd \overline{\mathtt{N}}\rangle \mathtt{T}\ \mathtt{m}(\overline{\mathtt{T}}\ \overline{\mathtt{x}})\{\uparrow \mathtt{e};\}$	$(\mathtt{M}_{\mathrm{FGJ}})$
$\mathtt{e} ::= \mathtt{x} \mid \mathtt{e}.\mathtt{f} \mid \mathtt{e}.\mathtt{m}\langle\overline{\mathtt{T}}\rangle(\overline{\mathtt{e}}) \mid \mathtt{new}\ \mathtt{N}(\overline{\mathtt{e}}) \mid (\mathtt{N})\mathtt{e} \mid$	$(\mathtt{e}_{\mathrm{FGJ}})$
$\quad \mid \mathtt{F} \mid \mathtt{e}.\mathtt{invoke}(\overline{\mathtt{e}})$	$(\mathtt{e}_{\mathrm{FGCJ}})$
$\mathtt{F} ::= \#(\overline{\mathtt{T}}\ \overline{\mathtt{x}})\mathtt{e}$	$(\mathtt{F}_{\mathrm{FGCJ}})$

[1] According to [IPW01] $\overline{\mathtt{f}}$ is a shorthand for a possibly empty sequence $\mathtt{f}_1, \ldots, \mathtt{f}_n$ (and similarly for $\overline{\mathtt{T}}, \overline{\mathtt{x}}$, etc.) and $\overline{\mathtt{M}}$ is a shorthand for $\mathtt{M}_1 \ldots \mathtt{M}_n$ with no commas). The empty sequence is $\bullet$ and denotes concatention of sequences using commas. Operations on pairs of sequences are abbreviated in the obvious way $\overline{\mathtt{C}}\,\overline{\mathtt{f}}$ is $\mathtt{C}_1\ \mathtt{f}_1, \ldots, \mathtt{C}_n\ \mathtt{f}_n$ and similarly $\overline{\mathtt{C}}\,\overline{\mathtt{f}}$; is $\mathtt{C}_1\ \mathtt{f}_1; \ldots \mathtt{C}_n\ \mathtt{f}_n$; and $\mathtt{this}.\overline{\mathtt{f}} = \overline{\mathtt{f}}$; is a shorthand for $\mathtt{this}.\mathtt{f}_1 = \mathtt{f}_1; \ldots \mathtt{this}.\mathtt{f}_n = \mathtt{f}_n$; Sequences of field declarations, parameters and method declaration cannot contain duplications. Casts have lower precedence than other operators.

A closure type specifies the formal types and the result type. Hence syntax for closure types is $\#\texttt{T}(\overline{\texttt{T}})$. An example is $\#(\texttt{Integer x}, \texttt{Integer y})\ \texttt{x} + \texttt{y}$ which is a closure with two arguments, whose type is $\#\texttt{Integer}(\texttt{Integer}, \texttt{Integer})$. No generic variables can be introduced when defining a closures but of course generic variables introduced in class or method declarations can be used inside closures. The complete syntax of the extended language is reported in `Table 1`. For the reader convenience, in all tables, but **Table 3**, the lines relative to FGJ have a label which is indexed by FGJ, while the lines relative to FGCJ have a label which is indexed by FGCJ .

3.2 Semantics: Reduction

The reduction semantics is given through the inference rules in **Table 2**, which define the reduction relation $\texttt{e} \rightarrow \texttt{e}'$ that says that "expression `e` reduces to expression $\texttt{e}'$ in one step". The values in FGCJ are not only objects but also closures, hence we have two cases:

1. $\texttt{new N}(\overline{\texttt{w}})$ Object as in FJ
2. $\#(\overline{\texttt{T}}\ \overline{\texttt{x}})\texttt{e}$ A non-recursive closure.

Since a closure is a value it cannot be further reduced, hence the only expressions which need to be considered, for reduction, are those containing a closure invocation. The rules contained in **Table 2: Congruence** are those which reduce a sub-expression contained in the expression being evaluated. We have added two rules GRC-Invk-Clos and GRC-Clos-Arg which consider the cases in which, the closure expression which is invoked, can be reduced and the case, instead, in which an actual parameter in the invoking expression can be reduced. The rules contained in **Table 2: Computation** are those which really show how the computation is carried on. In these rules, we only add rule GR-Invk-Clos which shows how a closure invocation is reduced. That is the closure invocation is replaced by its body expression in which the formal parameters are replaced by the corresponding actual ones. [2]The auxiliary functions for FGCJ are reported in **Table 3**. Actually they are the same as in [IPW01]. We only add a case (Over-Object) for predicate `override` when the class to which it is applied is `Object`.

3.3 Semantics: Typing

The typing rules are given through inference rules that use five different typing judgements: one for each different term structure of the language. The judgement for a (generic) type T see **Table 5** has the form $\Delta \vdash \texttt{T ok}$ meaning that "`T` is a well-formed type in the (well formed) type environment Δ". A (well formed) type environement Δ is a mapping from type variables to (well formed, in Δ) types written as a list of `X<:T`, meaning that type variable `X` must be bound to a subtype of type `T`. The typing judgements for subtyping (see **Table 5**) has the form $\Delta \vdash \texttt{S<:T}$ meaning that "`S` is a subtype of `T` in Δ". The judgement for classes (see rule GT-Class$_{\text{FGJ}}$in **Table 4**) has the form `C OK` meaning that "`C` is well typed". The typing judgements for methods (see GT-Method$_{\text{FGJ}}$in **Table 4**) has the form `M OK in C` meaning that "`M` is well typed when its declaration occurs in class `C`". The judgement for expressions (see the first nine

[2] What does E[#()`this`] compute? where E is an arbitrary correct expression containing [#()`this`]. The semantics guarantees `this` transparency.

rules of **Table 4**) has the form $\Delta; \Gamma \vdash \mathtt{e} : \mathtt{T}$ meaning that expression $\mathtt{e}$ has type $\mathtt{T}$ in the typing environment Δ and in the (variable) environment Γ. An environment Γ is a mapping from variables to types written as a list of $\mathtt{x} : \mathtt{T}$ meaning that "$\mathtt{x}$ has type $\mathtt{T}$". The typing rules are contained in **Table 4** and extends those of FGJ . Two rules have been added for closure construction and closure invocation. Such rules simply assert the correctness of the involved types.

Table 2

Computation

$$\frac{\mathit{fields}(\mathtt{N}) = \overline{\mathtt{T}}\ \overline{\mathtt{f}}}{(\mathtt{new}\ \mathtt{N}(\overline{\mathtt{e}})).\mathtt{f}_i \longrightarrow \mathtt{e}_i} \qquad (\text{GR-FIELD}_{\text{FGJ}})$$

$$\frac{\mathit{mbody}(\mathtt{m}\langle\overline{\mathtt{V}}\rangle, \mathtt{N}) = (\overline{\mathtt{x}}, \mathtt{e})}{(\mathtt{new}\ \mathtt{N}(\overline{\mathtt{e}})).\mathtt{m}\langle\overline{\mathtt{V}}\rangle(\overline{\mathtt{d}}) \longrightarrow [\overline{\mathtt{d}}/\overline{\mathtt{x}}, \mathtt{new}\ \mathtt{N}(\overline{\mathtt{e}})/\mathtt{this}]\mathtt{e}} \qquad (\text{GR-INVK}_{\text{FGJ}})$$

$$\frac{\emptyset \vdash \mathtt{N} <: \mathtt{P}}{(\mathtt{P})(\mathtt{new}\ \mathtt{N}(\overline{\mathtt{e}})) \rightarrow \mathtt{new}\ \mathtt{N}(\overline{\mathtt{e}})} \qquad (\text{GR-CAST}_{\text{FGJ}})$$

$$\#(\overline{\mathtt{T}}\,\overline{\mathtt{x}})\mathtt{e}.\mathtt{invoke}(\overline{\mathtt{d}}) \longrightarrow [\overline{\mathtt{d}}/\overline{\mathtt{x}}]\mathtt{e} \qquad (\text{GR-INVK-CLOS}_{\text{FGCJ}})$$

Congruence

$$\frac{\mathtt{e}_0 \longrightarrow \mathtt{e}_0'}{\mathtt{e}_0.\mathtt{f} \longrightarrow \mathtt{e}_0'.\mathtt{f}} \qquad (\text{GRC-FIELD}_{\text{FGJ}})$$

$$\frac{\mathtt{e}_0 \longrightarrow \mathtt{e}_0'}{\mathtt{e}_0.\mathtt{m}\langle\overline{\mathtt{T}}\rangle(\overline{\mathtt{e}}) \longrightarrow \mathtt{e}_0'.\mathtt{m}\langle\overline{\mathtt{T}}\rangle(\overline{\mathtt{e}})} \qquad (\text{GRC-T-INV}_{\text{FGJ}})$$

$$\frac{\mathtt{e}_i \longrightarrow \mathtt{e}_i'}{\mathtt{e}_0.\mathtt{m}\langle\overline{\mathtt{T}}\rangle(\ldots, \mathtt{e}_i, \ldots) \longrightarrow \mathtt{e}_0.\mathtt{m}\langle\overline{\mathtt{T}}\rangle(\ldots, \mathtt{e}_i' \ldots)} \qquad (\text{GRC-INV-ARG}_{\text{FGJ}})$$

$$\frac{\mathtt{e}_i \longrightarrow \mathtt{e}_i'}{\mathtt{new}\ \mathtt{N}(\ldots, \mathtt{e}_i, \ldots) \longrightarrow \mathtt{new}\ \mathtt{N}(\ldots, \mathtt{e}_i' \ldots)} \qquad (\text{GRC-NEW}_{\text{FGJ}})$$

$$\frac{\mathtt{e} \longrightarrow \mathtt{e}'}{(\mathtt{N})\mathtt{e} \longrightarrow (\mathtt{N})\mathtt{e}'} \qquad (\text{GRC-CAST}_{\text{FGJ}})$$

$$\frac{\mathtt{e} \longrightarrow \mathtt{e}'}{\mathtt{e}.\mathtt{invoke}(\overline{\mathtt{e}}) \longrightarrow \mathtt{e}'.\mathtt{invoke}(\overline{\mathtt{e}})} \qquad (\text{GRC-INV-CLOS}_{\text{FGCJ}})$$

$$\frac{\mathtt{e}_i \longrightarrow \mathtt{e}_i'}{\mathtt{e}.\mathtt{invoke}(\ldots, \mathtt{e}_i, \ldots) \longrightarrow \mathtt{e}.\mathtt{invoke}(\ldots, \mathtt{e}_i' \ldots)} \qquad (\text{GRC-CLOS-ARG}_{\text{FGCJ}})$$

Table 3

Subclassing

$$\mathtt{C} \trianglelefteq \mathtt{C} \qquad \frac{\mathtt{C} \trianglelefteq \mathtt{D} \qquad \mathtt{D} \trianglelefteq \mathtt{E}}{\mathtt{C} \trianglelefteq \mathtt{E}} \qquad \frac{\mathtt{class\ C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle \lhd \mathtt{D}\ \{\overline{\mathtt{S}}\ \overline{\mathtt{f}}; \mathtt{K}\ \overline{\mathtt{M}}\}}{\mathtt{C} \trianglelefteq \mathtt{D}}$$

Auxiliary functions

$$\mathit{fields}(\mathtt{Object}) = \bullet \qquad \text{(F-Object)}$$

$$\frac{\mathtt{class\ C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle \lhd \mathtt{N}\ \{\overline{\mathtt{S}}\ \overline{\mathtt{f}}; \mathtt{K}\ \overline{\mathtt{M}}\} \qquad \mathit{fields}([\overline{\mathtt{T}}/\overline{\mathtt{X}}]\mathtt{N}) = \overline{\mathtt{U}}\ \overline{\mathtt{g}}}{\mathit{fields}(\mathtt{C}\langle \overline{\mathtt{T}} \rangle) = \overline{\mathtt{U}}\ \overline{\mathtt{g}}, [\overline{\mathtt{T}}/\overline{\mathtt{X}}]\overline{\mathtt{S}}\ \overline{\mathtt{f}}} \qquad \text{(F-Class)}$$

$$\frac{\mathtt{class\ C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle \lhd \mathtt{N}\ \{\overline{\mathtt{S}}\ \overline{\mathtt{f}}; \mathtt{K}\ \overline{\mathtt{M}}\} \qquad \langle \overline{\mathtt{Y}} \lhd \overline{\mathtt{P}} \rangle \mathtt{U}\ \mathtt{m}\ (\overline{\mathtt{U}}\ \overline{\mathtt{x}})\{\uparrow \mathtt{e};\} \in \overline{\mathtt{M}}}{\mathit{mtype}(\mathtt{m}, \mathtt{C}\langle \overline{\mathtt{T}} \rangle) = [\overline{\mathtt{T}}/\overline{\mathtt{X}}](\langle \overline{\mathtt{Y}} \lhd \overline{\mathtt{P}} \rangle \overline{\mathtt{U}} \to \mathtt{U})} \qquad \text{(MT-Class)}$$

$$\frac{\mathtt{class\ C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle \lhd \mathtt{N}\ \{\overline{\mathtt{S}}\ \overline{\mathtt{f}}; \mathtt{K}\ \overline{\mathtt{M}}\} \qquad \mathtt{m} \notin \overline{\mathtt{M}}}{\mathit{mtype}(\mathtt{m}, \mathtt{C}\langle \overline{\mathtt{T}} \rangle) = \mathit{mtype}([\overline{\mathtt{T}}/\overline{\mathtt{X}}]\mathtt{N})} \qquad \text{(MT-Super)}$$

$$\frac{\mathtt{class\ C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle \lhd \mathtt{N}\ \{\overline{\mathtt{S}}\ \overline{\mathtt{f}}; \mathtt{K}\ \overline{\mathtt{M}}\} \qquad \langle \overline{\mathtt{Y}} \lhd \overline{\mathtt{P}} \rangle \mathtt{U}\ \mathtt{m}\ (\overline{\mathtt{U}}\ \overline{\mathtt{x}})\{\uparrow \mathtt{e};\} \in \overline{\mathtt{M}}}{\mathit{mbody}(\mathtt{m}\langle \overline{\mathtt{V}} \rangle, \mathtt{C}\langle \overline{\mathtt{T}} \rangle) = \overline{\mathtt{x}}.[\overline{\mathtt{T}}/\overline{\mathtt{X}}, \overline{\mathtt{V}}/\overline{\mathtt{Y}}]\mathtt{e}} \qquad \text{(MB-Class)}$$

$$\frac{\mathtt{class\ C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle \lhd \mathtt{N}\ \{\overline{\mathtt{S}}\ \overline{\mathtt{f}}; \mathtt{K}\ \overline{\mathtt{M}}\} \qquad \mathtt{m} \notin \overline{\mathtt{M}}}{\mathit{mbody}(\mathtt{m}\langle \overline{\mathtt{V}} \rangle, \mathtt{C}\langle \overline{\mathtt{T}} \rangle) = \mathit{mbody}([\overline{\mathtt{T}}/\overline{\mathtt{X}}]\mathtt{N})} \qquad \text{(MB-Super)}$$

Auxiliary predicates

$$\mathtt{override}(\mathtt{m}, \mathtt{Object}, \langle \overline{\mathtt{Y}} \lhd \overline{\mathtt{P}} \rangle \overline{\mathtt{T}} \to \mathtt{T}_0) \qquad \text{(Over-Object)}$$

$$\frac{\mathit{mtype}(\mathtt{m}, \mathtt{N}) = \langle \overline{\mathtt{Z}} \lhd \overline{\mathtt{Q}} \rangle \overline{\mathtt{U}} \to\ \mathtt{U}_0\ \mathtt{implies}\ (\overline{\mathtt{P}}, \overline{\mathtt{T}}) = [\overline{\mathtt{Y}}/\overline{\mathtt{Z}}](\overline{\mathtt{Q}}, \overline{\mathtt{U}})\ \mathtt{and}\ \overline{\mathtt{Y}} <: \overline{\mathtt{P}} \vdash \mathtt{T}_0 <: [\overline{\mathtt{Y}}/\overline{\mathtt{Z}}]\mathtt{U}_0}{\mathtt{override}(\mathtt{m}, \mathtt{N}, \langle \overline{\mathtt{Y}} \lhd \overline{\mathtt{P}} \rangle \overline{\mathtt{T}} \to \mathtt{T}_0)} \qquad \text{(Over)}$$

DCast

$$\frac{\mathit{dcast}(\mathtt{C}, \mathtt{D}) \qquad \mathit{dcast}(\mathtt{D}, \mathtt{E})}{\mathit{dcast}(\mathtt{C}, \mathtt{E})} \qquad \frac{\mathtt{class\ C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle \lhd \mathtt{D}\langle \overline{\mathtt{T}} \rangle\ \{\ldots\} \qquad \overline{\mathtt{X}} = \mathit{FV}(\overline{\mathtt{T}})}{\mathit{dcast}(\mathtt{C}, \mathtt{D})} \qquad \text{(DCast)}$$

$\mathit{FV}(\overline{\mathtt{T}})$ denotes the set of type variables in $\overline{\mathtt{T}}$

The rules for subtypes and wellformed types are reported in **Table 5**. Two rules are added (B-Closure) which states that the *bound* for a closure type is the closure type itself, and (WF-Closure) which states that a closure type is well-formed if the types involved are well-formed. Note that there are no subtypying rules for closures, this means that if $\mathtt{T}_1$ and $\mathtt{T}_2$ are two closure types $\mathtt{T}_1 <: \mathtt{T}_2 \iff \mathtt{T}_1 = \mathtt{T}_2$.

Table 4

Typing rules

$$\Delta; \Gamma \vdash \mathtt{x} : \Gamma(\mathtt{x}) \qquad \text{(GT-VAR}_{\text{FGJ}}\text{)}$$

$$\frac{\Delta; \Gamma \vdash \mathtt{e}_0 : \mathtt{T}_0 \qquad \mathit{fields}(\mathit{bound}_\Delta(\mathtt{T}_0)) = \overline{\mathtt{T}}\ \overline{\mathtt{f}}}{\Delta; \Gamma \vdash \mathtt{e}_0.\mathtt{f}_i : \mathtt{T}_i} \qquad \text{(GT-FIELD}_{\text{FGJ}}\text{)}$$

$$\frac{\begin{array}{c}\mathit{mtype}(\mathtt{m}, \mathit{bound}_\Delta(\mathtt{T}_0)) = \langle \overline{\mathtt{Y}} \lhd \overline{\mathtt{P}} \rangle \overline{\mathtt{U}} \to \mathtt{U} \\ \Delta; \Gamma \vdash \mathtt{e}_0 : \mathtt{T}_0 \qquad \Delta \vdash \overline{\mathtt{V}}\ \mathtt{ok} \qquad \Delta \vdash \overline{\mathtt{V}} <: [\overline{\mathtt{V}}/\overline{\mathtt{Y}}]\overline{\mathtt{P}} \\ \Delta; \Gamma \vdash \overline{\mathtt{e}} : \overline{\mathtt{S}} \qquad \Delta \vdash \overline{\mathtt{S}} <: [\overline{\mathtt{V}}/\overline{\mathtt{Y}}]\overline{\mathtt{U}}\end{array}}{\Delta; \Gamma \vdash \mathtt{e}_0.\mathtt{m}\langle \overline{\mathtt{V}} \rangle (\overline{\mathtt{e}}) : [\overline{\mathtt{V}}/\overline{\mathtt{Y}}]\mathtt{U}} \qquad \text{(GT-INV}_{\text{FGJ}}\text{)}$$

$$\frac{\begin{array}{c}\Delta \vdash \mathtt{N}\ \mathtt{ok} \qquad \mathit{fields}(\mathtt{N}) = \overline{\mathtt{T}}\ \overline{\mathtt{f}} \\ \Delta; \Gamma \vdash \overline{\mathtt{e}} : \overline{\mathtt{S}} \qquad \Delta \vdash \overline{\mathtt{S}} <: \overline{\mathtt{T}}\end{array}}{\Delta; \Gamma \vdash \mathtt{new}\ \mathtt{N}(\overline{\mathtt{e}}) : \mathtt{N}} \qquad \text{(GT-NEW}_{\text{FGJ}}\text{)}$$

$$\frac{\Delta; \Gamma \vdash \mathtt{e}_0 : \mathtt{T}_0 \qquad \Delta \vdash \mathit{bound}_\Delta(\mathtt{T}_0) <: \mathtt{N}}{\Delta; \Gamma \vdash (\mathtt{N})\mathtt{e}_0 : \mathtt{N}} \qquad \text{(GT-UCAST}_{\text{FGJ}}\text{)}$$

$$\frac{\begin{array}{c}\Delta; \Gamma \vdash \mathtt{e}_0 : \mathtt{T}_0 \qquad \Delta \vdash \mathtt{N}\ \mathtt{ok} \qquad \Delta \vdash \mathtt{N} <: \mathit{bound}_\Delta(\mathtt{T}_0) \\ \mathtt{N} = \mathtt{C}\langle \overline{\mathtt{T}} \rangle \qquad \mathit{bound}_\Delta(\mathtt{T}_0) = \mathtt{D}\langle \overline{\mathtt{T}} \rangle \qquad \mathit{dcast}(\mathtt{C}, \mathtt{D})\end{array}}{\Delta; \Gamma \vdash (\mathtt{N})\mathtt{e}_0 : \mathtt{N}} \qquad \text{(GT-DCAST}_{\text{FGJ}}\text{)}$$

$$\frac{\begin{array}{c}\Delta; \Gamma \vdash \mathtt{e}_0 : \mathtt{T}_0 \qquad \Delta \vdash \mathtt{N}\ \mathtt{ok} \\ \mathtt{N} = \mathtt{C}\langle \overline{\mathtt{T}} \rangle \qquad \mathit{bound}_\Delta(\mathtt{T}_0) = \mathtt{D}\langle \overline{\mathtt{U}} \rangle \qquad \mathtt{C} \not<: \mathtt{D} \qquad \mathtt{D} \not<: \mathtt{C}\end{array}}{\Delta; \Gamma \vdash (\mathtt{N})\mathtt{e}_0 : \mathtt{N}} \qquad \text{(GT-SCAST}_{\text{FGJ}}\text{)}$$

$$\frac{\Delta \vdash \overline{\mathtt{T}}\ \mathtt{ok} \qquad \Delta; \Gamma, \overline{\mathtt{x}} : \overline{\mathtt{T}} \vdash \mathtt{e}\ \ : \mathtt{T}}{\Delta; \Gamma \vdash \#(\overline{\mathtt{T}}\ \overline{\mathtt{x}})\ \mathtt{e} : \#\mathtt{T}(\overline{\mathtt{T}})} \qquad \text{(GT-CLOSURE}_{\text{FGCJ}}\text{)}$$

$$\frac{\Delta; \Gamma \vdash \mathtt{e} : \#\mathtt{T}(\overline{\mathtt{T}}) \qquad \Delta; \Gamma \vdash \overline{\mathtt{e}} : \overline{\mathtt{S}} \qquad \Delta \vdash \overline{\mathtt{S}} <: \overline{\mathtt{T}}}{\Delta; \Gamma \vdash \mathtt{e}.\mathtt{invoke}(\overline{\mathtt{e}}) : \mathtt{T}} \qquad \text{(GT-CLOSURE-INV}_{\text{FGCJ}}\text{)}$$

$$\frac{\begin{array}{c}\Delta = \overline{\mathtt{X}} <: \overline{\mathtt{N}},\ \overline{\mathtt{Y}} <: \overline{\mathtt{P}} \qquad \Delta \vdash \overline{\mathtt{T}}, \mathtt{T}, \overline{\mathtt{P}}\ \mathtt{ok} \\ \Delta; \overline{\mathtt{x}} : \overline{\mathtt{T}}, \mathtt{this} : \mathtt{C}\langle \overline{\mathtt{X}} \rangle \vdash \mathtt{e}_0 : \mathtt{S} \qquad \Delta \vdash \mathtt{S} <: \mathtt{T} \\ \mathtt{class}\ \mathtt{C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle \lhd \mathtt{N}\{...\} \qquad \mathtt{override}(\mathtt{m}, \mathtt{N}, \langle \overline{\mathtt{Y}} \lhd \overline{\mathtt{P}} \rangle \overline{\mathtt{T}} \to \mathtt{T})\end{array}}{\langle \overline{\mathtt{Y}} \lhd \overline{\mathtt{P}} \rangle \mathtt{T}\ \ \mathtt{m}(\overline{\mathtt{T}}\ \overline{\mathtt{x}})\{\uparrow \mathtt{e}_0;\}\mathtt{OK\ IN}\ \ \mathtt{C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle} \qquad \text{(GT-METHOD}_{\text{FGJ}}\text{)}$$

$$\frac{\begin{array}{c}\overline{\mathtt{X}} <: \overline{\mathtt{N}} \vdash \overline{\mathtt{N}}, \mathtt{N}, \overline{\mathtt{T}}\ \mathtt{ok} \qquad \overline{\mathtt{M}}\ \mathtt{ok\ in}\ \mathtt{C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle \\ \mathit{fields}(\mathtt{N}) = \overline{\mathtt{U}}\ \overline{\mathtt{g}} \qquad \mathtt{K} = \mathtt{C}(\overline{\mathtt{U}}\ \overline{\mathtt{g}}, \overline{\mathtt{T}}\ \overline{\mathtt{f}})\{\mathrm{super}(\overline{\mathtt{g}});\ \mathrm{this}.\overline{\mathtt{f}} = \overline{\mathtt{f}};\}\end{array}}{\mathtt{class}\ \mathtt{C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle \lhd \mathtt{N}\{\overline{\mathtt{T}}\ \overline{\mathtt{f}}; \mathtt{K}\ \overline{\mathtt{M}}\}\ \mathtt{OK}} \qquad \text{(GT-CLASS}_{\text{FGJ}}\text{)}$$

Table 5

Subtypes

$$bound_{\Delta}(\mathtt{X}) = \Delta(\mathtt{X}) \qquad (\text{B-VAR}_{\text{FGJ}})$$

$$bound_{\Delta}(\mathtt{N}) = \mathtt{N} \qquad (\text{B-CLASS}_{\text{FGJ}})$$

$$bound_{\Delta}(\#\mathtt{T}(\overline{\mathtt{T}})) = \#\mathtt{T}(\overline{\mathtt{T}}) \qquad (\text{B-CLOSURE}_{\text{FGCJ}})$$

$$\Delta \vdash \mathtt{T} <: \mathtt{T} \qquad (\text{S-REFL}_{\text{FGJ}})$$

$$\frac{\Delta \vdash \mathtt{S} <: \mathtt{T} \qquad \Delta \vdash \mathtt{T} <: \mathtt{U}}{\Delta \vdash \mathtt{S} <: \mathtt{U}} \qquad (\text{S-TRANS}_{\text{FGJ}})$$

$$\Delta \vdash \mathtt{X} <: \Delta(\mathtt{X}) \qquad (\text{S-VAR}_{\text{FGJ}})$$

$$\frac{\mathtt{class}\ \mathtt{C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle \lhd \mathtt{N}\{\ldots\}}{\Delta \vdash \mathtt{C}\langle \overline{\mathtt{T}} \rangle <: [\overline{\mathtt{T}}/\overline{\mathtt{X}}]\mathtt{N}} \qquad (\text{S-CLASS}_{\text{FGJ}})$$

Well-formed types

$$\Delta \vdash \text{Object}\ \mathtt{ok} \qquad (\text{WF-OBJECT}_{\text{FGJ}})$$

$$\frac{\mathtt{X} \in dom(\Delta)}{\Delta \vdash \mathtt{X}\ \mathtt{ok}} \qquad (\text{WF-VAR}_{\text{FGJ}})$$

$$\frac{\mathtt{class}\ \mathtt{C}\langle \overline{\mathtt{X}} \lhd \overline{\mathtt{N}} \rangle \lhd \mathtt{N}\{\ldots\} \qquad \Delta \vdash \overline{\mathtt{T}}\ \mathtt{ok} \qquad \Delta \vdash \overline{\mathtt{T}} <: [\overline{\mathtt{T}}/\overline{\mathtt{X}}]\overline{\mathtt{N}}}{\Delta \vdash \mathtt{C}\langle \overline{\mathtt{T}} \rangle\ \mathtt{ok}} \qquad (\text{WF-CLASS}_{\text{FGJ}})$$

$$\frac{\Delta \vdash \overline{\mathtt{T}}\ \mathtt{ok} \qquad \Delta \vdash \mathtt{T}\ \mathtt{ok}}{\Delta \vdash \#\mathtt{T}(\overline{\mathtt{T}})\ \mathtt{ok}} \qquad (\text{WF-CLOSURE}_{\text{FGCJ}})$$

4 Properties

Analogously to [IPW01] we prove subject reduction theorem and progress theorem. Type soundness immediately follows. Several interesting lemmas are used in the proofs. They are stated and proved in the appendix.

Theorem 1 (Subject reduction). *If* $\Delta; \Gamma \vdash \mathtt{e} : \mathtt{T}$ *and* $\mathtt{e} \to \mathtt{e}'$ *then* $\Delta; \Gamma \vdash \mathtt{e}' : \mathtt{T}'$*, for some* $\mathtt{T}'$ *such that* $\Delta \vdash \mathtt{T}' <: \mathtt{T}$

Proof. See Appendix A

Theorem 2 (Progress). *Suppose* e *is a well-typed expression. If* e *includes as a subexpression:*

1. $\mathtt{new}\ \mathtt{N}(\overline{\mathtt{e}}).\mathtt{f}$ *then* $fields(\mathtt{N}) = \overline{\mathtt{T}}\ \overline{\mathtt{f}}$ *and* $\mathtt{f} \in \overline{\mathtt{f}}$ *for some* $\overline{\mathtt{T}}$ *and* $\overline{\mathtt{f}}$*.*

2. $\texttt{new N}(\overline{\texttt{e}}).\texttt{m}\langle\overline{\texttt{V}}\rangle(\overline{\texttt{d}})$ *then* $mbody(\texttt{m}\langle\overline{\texttt{V}}\rangle, \texttt{N}) = \overline{\texttt{x}}.\texttt{e}_0$ *and* $\#\overline{\texttt{x}} = \#\overline{\texttt{d}}$ *for some* $\overline{\texttt{x}}$ *and* $\texttt{e}_0$.
3. $\texttt{F.invoke}(\overline{\texttt{d}})$ *then* $\texttt{F} = \#(\overline{\texttt{T}}\ \overline{\texttt{x}})\ \texttt{e}_0$ *and* $\#\overline{\texttt{x}} = \#\overline{\texttt{d}}$.

Proof. **Theorem** 2 The proof is based on the analysis of all well typed expressions, which can be reduced to the above 3 cases to conclude that either it is in normal form or it can be further reduced to obtain a normal form. As already stated in section 3.2, in FGCJ there are 2 possible normal forms i.e. values. They are:

- $\texttt{new N}(\overline{\texttt{w}})$ Object as in FJ
- $\#(\overline{\texttt{T}}\ \overline{\texttt{x}})\texttt{e}$ A non-recursive closure □

Theorem 3 (Type Soundness). *If* $\emptyset;\emptyset \vdash \texttt{e} : \texttt{T}$ *and* $\texttt{e} \longrightarrow^* \texttt{e}'$ *with* $\texttt{e}'$ *a normal form, then* $\texttt{e}'$ *is a value* $\texttt{w}$ *with* $\emptyset;\emptyset \vdash \texttt{w} : \texttt{S}$ *and* $\emptyset \vdash \texttt{S}<:\texttt{T}$

Proof. Immediate from above Theorems □

We prove that the extension preserves the meaning of the programs of the original language FGJ.

Theorem 4 (Backward compatibility). *If an* FGJ *program is well typed under the* FGJ *rules it is also well typed under the* FGCJ *rules. Moreover, for all FJ programs* e *and* e' *(whether well typed or not)* $e \rightarrow_{\text{FGJ}} e' \iff e \rightarrow_{\text{FGCJ}} e'$.

Proof. All FGCJ sets of rules include FGCJ rules □

5 Conclusion

In this paper we addressed the problem of defining a rigorous semantics for closures in Java. A minimal calculus is defined extending Featherweight Java, both for syntax and semantics and two properties: type soundness and backward compatibility are proved. Actually, the official proposal for closures in Java is entrusted to JDK7 project of the OpenJDK committee who has recently communicated a preliminary proposal [BGR10]. In such proposal Simple Closures have been modified for: *i)* the use of `this` in a closure structure which is simplified to the usual *self-reference*, allowing implicit recursive closures; and *ii)* subtyping is allowed for closure types. The FGCJ calculus described in this paper can be easily adapted to new features for closures. In particular *i)* is obtained modifying the rule for closure invocation in a quite obvious way; while *ii)* is obtained adding one rule for subtyping and revisiting asserts and proofs of the various discussed properties. In [BO09] a translation semantics for a somewhat different proposal for closures in Java is defined. Such translational approach defines a rigorous semantics for closures, but is not suited to prove semantic properties. We plan to adapt translational semantics to Simple Closures and analogously to [IPW01] prove that the two semantics commute. Eventually this paper is a partial result of a more complex and ambitious project of extension and prototype implementation of Java with higher order features, besides closures, methods passed as parameters [BO08].

A Lemmas and theorem proofs

Many lemmas are the same as those in [IPW01] in such case the number in [IPW01] of the lemma is reported in the name. Furthermore, proofs are omitted when already contained or easy extensions of those contained in [IPW01].

Lemma 1 (A.2.1 Weakening). *Suppose* $\Delta, \overline{\mathtt{X}}{<:}\overline{\mathtt{N}} \vdash \overline{\mathtt{N}}$ ok *and* $\Delta \vdash \mathtt{U}$ ok

1. *If* $\Delta \vdash \mathtt{S}{<:}\mathtt{T}$*, then* $\Delta, \overline{\mathtt{X}}{<:}\overline{\mathtt{N}} \vdash \mathtt{S}{<:}\mathtt{T}$
2. *If* $\Delta \vdash \mathtt{S}$ ok*, then* $\Delta, \overline{\mathtt{X}}{<:}\overline{\mathtt{N}} \vdash \mathtt{S}$ ok
3. *If* $\Delta; \Gamma \vdash \mathtt{e} : \mathtt{T}$*, then* $\Delta; \Gamma, \mathtt{x} : \mathtt{U} \vdash \mathtt{e} : \mathtt{T}$ *and* $\Delta; \overline{\mathtt{X}}{<:}\overline{\mathtt{N}}; \Gamma \vdash \mathtt{e} : \mathtt{T}$

Lemma 2 (A.2.2.). *If* $\Delta \vdash \mathtt{E}\langle\overline{\mathtt{V}}\rangle{<:}\mathtt{D}\langle\overline{\mathtt{U}}\rangle$ *and* $\mathtt{D} \not<: \mathtt{C}$ *and* $\mathtt{C} \not<: \mathtt{D}$ *then* $\mathtt{E} \not<: \mathtt{C}$ *and* $\mathtt{C} \not<: \mathtt{E}$.

Lemma 3 (A.2.3.). *Suppose* $dcast(\mathtt{C}, \mathtt{D})$ *and* $\Delta \vdash \mathtt{C}\langle\overline{\mathtt{T}}\rangle{<:}\ \mathtt{D}\langle\overline{\mathtt{U}}\rangle$*. If* $\Delta \vdash \mathtt{C}\langle\overline{\mathtt{T}}'\rangle{<:}\ \mathtt{D}\langle\overline{\mathtt{U}}\rangle$ *then* $\overline{\mathtt{T}}' = \overline{\mathtt{T}}$.

Lemma 4 (A.2.4.). *If* $dcast(\mathtt{C}, \mathtt{E})$ *and* $\mathtt{C} \lhd \mathtt{D} \lhd \mathtt{E}$ *with* $\mathtt{C} \neq \mathtt{D} \neq \mathtt{E}$*, then* $dcast(\mathtt{C}, \mathtt{D})$ *and* $dcast(\mathtt{D}, \mathtt{E})$.

Lemma 5 (A.2.5.). *If* $\Delta_1, \overline{\mathtt{X}}{<:}\overline{\mathtt{N}}, \Delta_2 \vdash \mathtt{S}{<:}\mathtt{T}$ *and* $\Delta_1 \vdash \overline{\mathtt{U}}{<:}[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{N}}$ *with* $\Delta_1 \vdash \overline{\mathtt{U}}$ ok *and none* $\overline{\mathtt{X}}$ *appearing in* Δ_1*, then* $\Delta_1, [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\Delta_2 \vdash [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{S}{<:}[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{T}$.

Lemma 6 (A.2.6.). *If* $\Delta_1, \overline{\mathtt{X}}{<:}\overline{\mathtt{N}}, \Delta_2 \vdash \mathtt{T}$ ok *and* $\Delta_1 \vdash \overline{\mathtt{U}}{<:}[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{N}}$ *with* $\Delta_1 \vdash \overline{\mathtt{U}}$ ok *and none of* $\overline{\mathtt{X}}$ *appearing in* Δ_1 *then* $\Delta_1, [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\Delta_2 \vdash [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{T}$ ok

Lemma 7 (A.2.7.). *Suppose* $\Delta_1, \overline{\mathtt{X}}{<:}\overline{\mathtt{N}}, \Delta_2 \vdash \mathtt{T}$ ok *and* $\Delta_1 \vdash \overline{\mathtt{U}}{<:}[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{N}}$ *with* $\Delta_1 \vdash \overline{\mathtt{U}}$ ok *and none of* $\overline{\mathtt{X}}$ *appearing in* Δ_1 *then,*

$$\Delta_1[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\Delta_2 \vdash bound_{\Delta_1, [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\Delta_2}([\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{T}){<:}[\overline{\mathtt{U}}/\overline{\mathtt{X}}](bound_{\Delta_1, \overline{\mathtt{X}}{<:}\overline{\mathtt{N}}, \Delta_2}(\mathtt{T}))$$

Lemma 8 (A.2.8). *If* $\Delta \vdash \mathtt{S}{<:}\mathtt{T}$ *and* $fields(bound_\Delta(\mathtt{T})) = \overline{\mathtt{T}}\ \overline{\mathtt{f}}$*, then* $fields(bound_\Delta(\mathtt{S})) = \overline{\mathtt{S}}\ \overline{\mathtt{g}}$ *and* $\mathtt{S}_i = \mathtt{T}_i$ *and* $\mathtt{g}_i = \mathtt{f}_i$ *for all* $i \leq \#(\overline{\mathtt{f}})$.

Lemma 9 (A.2.9.). *If* $\Delta \vdash \mathtt{T}$ ok *and* $mtype(\mathtt{m}, bound_\Delta(\mathtt{T})) = \langle\overline{\mathtt{Y}} \lhd \overline{\mathtt{P}}\rangle\overline{\mathtt{T}} \to \mathtt{U}_0$ *then for any* $\mathtt{S}$ *such that* $\Delta \vdash \mathtt{S}{<:}\mathtt{T}$ *and* $\Delta \vdash \mathtt{S}$ ok*, we have* $mtype(\mathtt{m}, bound_\Delta(\mathtt{S})) = \langle\overline{\mathtt{Y}} \lhd \overline{\mathtt{P}}\rangle\overline{\mathtt{T}} \to \mathtt{U}_0'$ *and* $\Delta, \overline{\mathtt{Y}}{<:}\overline{\mathtt{P}} \vdash \mathtt{U}_0'{<:}\mathtt{U}_0$

Lemma 10 (A.2.10.). *If* $\Delta_1, \overline{\mathtt{X}}{<:}\overline{\mathtt{N}}, \Delta_2; \Gamma \vdash \mathtt{e} : \mathtt{T}$ *and* $\Delta_1 \vdash \overline{\mathtt{U}}{<:}[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{N}}$ *where* $\Delta_1 \vdash \overline{\mathtt{U}}$ ok *and none of* $\overline{\mathtt{X}}$ *appears in* Δ_1*, then* $\Delta_1, [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\Delta_2; [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\Gamma \vdash [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{e} : \mathtt{S}$ *for some* $\mathtt{S}$ *such that* $\Delta_1, [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\Delta_2 \vdash \mathtt{S}{<:}[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{T}$

Proof. As in [IPW01], the proof is given by induction and case analysis. We specify only the new cases.

Case GT-Closure

$$\begin{array}{l}\mathtt{e} = \#(\overline{\mathtt{W}}\ \overline{\mathtt{w}})\ \mathtt{e}_0 \qquad \Delta = \Delta_1,\ \overline{\mathtt{X}}{<:}\overline{\mathtt{N}}, \Delta_2 \\ \Delta; \Gamma \vdash \mathtt{e}_0 : \mathtt{W}_0 \qquad \mathtt{T} = \#\mathtt{W}(\overline{\mathtt{W}}) \qquad \Delta \vdash \mathtt{W}_0{<:}\mathtt{W} \\ [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{e} = \#([\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{W}}\ \overline{\mathtt{w}})[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{e}_0\end{array}$$

Hence $\mathtt{S} = \#[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{W}([\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{W}}) = [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{T}$.

Case GT-Closure-Inv

$$\begin{array}{l}\mathtt{e} = \mathtt{e}_0.\mathtt{invoke}(\overline{\mathtt{e}}) \qquad \mathtt{T} = \mathtt{W} \\ \Delta; \Gamma \vdash \mathtt{e}_0 : \#\mathtt{W}(\overline{\mathtt{W}}) \qquad \Delta; \Gamma \vdash \overline{\mathtt{e}} : \overline{\mathtt{S}}{<:}\overline{\mathtt{W}} \\ [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{e} = [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{e}_0.\mathtt{invoke}([\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{e}})\end{array}$$

By induction hypothesis $\Delta; \Gamma \vdash [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{e}} : \overline{\mathtt{Q}}$ such that $\Delta \vdash \overline{\mathtt{Q}}{<:}[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{S}}$. By Lemma 5 since $\Delta \vdash \overline{\mathtt{S}}{<:}\overline{\mathtt{W}}$ then $\Delta \vdash [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{S}}{<:}[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{W}}$ and $\Delta \vdash \overline{\mathtt{Q}}{<:}[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{W}}$ by rule S-Trans. Eventually, by induction hypothesis $\Delta; \Gamma \vdash [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{e}_0 : \mathtt{Q}$ such that $\Delta \vdash \mathtt{Q}{<:}[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\#(\mathtt{W}(\overline{\mathtt{W}}))$ and by Lemma 13 $\mathtt{Q} \equiv [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\#(\mathtt{W}(\overline{\mathtt{W}})) = \#[\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{W}([\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{W}})$. Hence by rule GT-Closure-Inv $\Delta; \Gamma \vdash [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{e}_0.\mathtt{invoke}([\overline{\mathtt{U}}/\overline{\mathtt{X}}]\overline{\mathtt{e}}) : [\overline{\mathtt{U}}/\overline{\mathtt{X}}]\mathtt{W}$ □

Lemma 11 (A.2.11.). *If* $\Delta; \Gamma, \overline{\mathtt{x}} : \overline{\mathtt{T}} \vdash \mathtt{e} : \mathtt{T}$ *and* $\Delta; \Gamma \vdash \overline{\mathtt{d}} : \overline{\mathtt{S}}$ *where* $\Delta \vdash \overline{\mathtt{S}} <: \overline{\mathtt{T}}$*, then* $\Delta; \Gamma \vdash [\overline{\mathtt{d}}/\overline{\mathtt{x}}]\mathtt{e} : \mathtt{S}$ *for some* $\mathtt{S}$ *such that* $\Delta \vdash \mathtt{S} <: \mathtt{T}$

Proof. s in [IPW01], the proof is given by induction and case analysis. We specify only the new cases.

Case GT-Closure

$$\begin{array}{l} \mathtt{e} = \#(\overline{\mathtt{W}}\ \overline{\mathtt{y}})\mathtt{e}_0 \qquad \Delta \vdash \overline{\mathtt{W}}, \mathtt{W}\ \mathtt{ok} \\ \Delta; \Gamma, \overline{\mathtt{y}} : \overline{\mathtt{W}} \vdash \mathtt{e}_0 : \mathtt{W}_0 \qquad \Delta \vdash \mathtt{W}_0 <: \mathtt{W} \qquad \mathtt{T} = \#\mathtt{W}(\overline{\mathtt{W}}) \end{array}$$

By the induction hypothesis, $\Delta; \Gamma \vdash [\overline{\mathtt{d}}/\overline{\mathtt{x}}]\mathtt{e}_0 : \mathtt{W}_1$ for some $\mathtt{W}_1$ such that $\Delta \vdash \mathtt{W}_1 <: \mathtt{W}_0$ and $[\overline{\mathtt{d}}/\overline{\mathtt{x}}](\#(\overline{\mathtt{W}}\ \overline{\mathtt{y}})\mathtt{e}_0) = \#(\overline{\mathtt{W}}\ \overline{\mathtt{y}})[\overline{\mathtt{d}}/\overline{\mathtt{x}}]\mathtt{e}_0$ since $\overline{\mathtt{x}} \cap \overline{\mathtt{y}} = \{\}$. Hence GT-Closure premises are satisfied and $\mathtt{S} = \#\mathtt{W}(\overline{\mathtt{W}})$.

Case GT-Closure-Inv

$$\begin{array}{l} \mathtt{e} = \mathtt{e}_0.\mathtt{invoke}(\overline{\mathtt{e}}) \qquad \Delta; \Gamma \vdash \mathtt{e}_0 : \#\mathtt{W}(\overline{\mathtt{W}}) \\ \Delta; \Gamma, \overline{\mathtt{e}} : \overline{\mathtt{S}} \qquad \Delta \vdash \overline{\mathtt{S}} <: \overline{\mathtt{W}} \qquad \mathtt{T} = \mathtt{W} \end{array}$$

By the induction hypothesis, $\Delta; \Gamma \vdash [\overline{\mathtt{d}}/\overline{\mathtt{x}}]\mathtt{e}_0 : \mathtt{W}_0$ for some $\mathtt{W}_0$ such that $\Delta \vdash \mathtt{W}_0 <: \#\mathtt{W}(\overline{\mathtt{W}})$ and $\Delta; \Gamma \vdash [\overline{\mathtt{d}}/\overline{\mathtt{x}}]\overline{\mathtt{e}} : \overline{\mathtt{Q}}$ for some $\overline{\mathtt{Q}} <: \overline{\mathtt{W}}$ because of subtyping on closures $\mathtt{W}_0 = \#\mathtt{W}(\overline{\mathtt{W}})$ hence $\mathtt{S} = \mathtt{W}$ □

Lemma 12 (A.2.12.). *If* $mtype(\mathtt{m}, \mathtt{C}\langle\mathtt{T}\rangle) = \langle\overline{\mathtt{Y}} \lhd \overline{\mathtt{P}}\rangle\overline{\mathtt{U}} \to \mathtt{U}$ *and* $mbody(\mathtt{m}\langle\overline{\mathtt{V}}\rangle, \mathtt{C}\langle\overline{\mathtt{T}}\rangle) = \overline{\mathtt{x}}.\mathtt{e}_0$ *where* $\Delta \vdash \mathtt{C}\langle\overline{\mathtt{T}}\rangle\ \mathtt{ok}$*) and* $\Delta \vdash \overline{\mathtt{V}}\ \mathtt{ok}$ *and* $\Delta \vdash \overline{\mathtt{V}} <: [\overline{\mathtt{V}}/\overline{\mathtt{Y}}]\overline{\mathtt{P}}$*, then there exist some* $\mathtt{N}$ *and* $\mathtt{S}$ *such that* $\Delta \vdash \mathtt{C}\langle\overline{\mathtt{T}}\rangle <: \mathtt{N}$ *and* $\Delta \vdash \mathtt{S} <: [\overline{\mathtt{V}}/\overline{\mathtt{Y}}]\mathtt{U}$ *and* $\Delta \vdash \mathtt{S}\ \mathtt{ok}$ *and* $\Delta; \overline{\mathtt{x}} : [\overline{\mathtt{V}}/\overline{\mathtt{Y}}]\overline{\mathtt{U}}, \mathtt{this} : \mathtt{N} \vdash \mathtt{e}_0 : \mathtt{S}$*.*

Lemma 13 (Closure subtyping). *If* $\Delta \vdash \mathtt{W}, \mathtt{T}, \overline{\mathtt{T}}\ \mathtt{ok}$ *then*

1. $\Delta \vdash \mathtt{W} <: \mathtt{T}(\overline{\mathtt{T}}) \iff \mathtt{W} \equiv \mathtt{T}(\overline{\mathtt{T}})$
2. $\Delta \vdash \mathtt{T}(\overline{\mathtt{T}}) <: \mathtt{W} \iff \mathtt{W} \equiv \mathtt{T}(\overline{\mathtt{T}})$

Proof. Immediate after subtyping rules □

Proof. **Theorem** 1 By induction on the reduction $\mathtt{e} \to \mathtt{e}'$, with a case analysis on the reduction rule used. It extends the proof, [IPW01] (pp. 426-428), of the corresponding theorem for FGJ with the following cases

Case GR-Inv-Clos $\mathtt{e} = \{\overline{\mathtt{T}}\ \overline{\mathtt{x}} : \mathtt{T} \Rightarrow\uparrow\ \mathtt{e}_0\}(\overline{\mathtt{d}})$ and by rules GT-Closure and GT-Closure-Inv, we have: $\Delta; \Gamma \vdash \{\overline{\mathtt{T}}\ \overline{\mathtt{x}} : \mathtt{T} \Rightarrow\uparrow\ \mathtt{e}_0\} : \mathtt{T}(\overline{\mathtt{T}})$, $\overline{\mathtt{d}} : \overline{\mathtt{S}}$, $\overline{\mathtt{S}} <: \overline{\mathtt{T}}$ and $\Delta; \Gamma, \overline{\mathtt{x}} : \overline{\mathtt{T}} \vdash \mathtt{e}_0 : \mathtt{S} <: \mathtt{T}$. Furthermore $\mathtt{e}' = [\overline{\mathtt{d}}/\overline{\mathtt{x}}]\mathtt{e}_0$ and by the Lemma 11 $\Delta; \Gamma \vdash \mathtt{e}' = [\overline{\mathtt{d}}/\overline{\mathtt{x}}]\mathtt{e}_0 : \mathtt{R}$ for some $\mathtt{R} <: \mathtt{S}$. □

References

[BGGA06] G. Bracha, N. Gafter, J. Gosling, and P. von der Ahe. Closures for java, 2006. //blogs.sun.com/ahe/resource/closures.pdf.

[BGR10] A. Buckley, J. Gibbons, and M. Reinhold. *Project Lambda: Java Language Specification. Version 0.1.* Sun Microsystem, Inc., 2010. //openJDK.java.net/projects/JDK7/features.

[BO05] M. Bellia and M.E. Occhiuto. Higher order programming in Java: Introspection, Subsumption and Extraction. *Fundamenta Informaticae*, 67(1):29–44, 2005.

[BO08] M. Bellia and M.E. Occhiuto. Methods as parameters: A preprocessing approach to higher order in java. *Fundamenta Informaticae*, 85(1):35–50, 2008.

[BO09] M. Bellia and M.E. Occhiuto. *JavaΩ: with Processing Closures in Java*. University of Pisa, Dipartimento Informatica, 2009. Submitted for publication.

[BOSW98] G. Bracha, M. Odersky, D. Stoutamire, and P. Wadler. Making the future safe for the past: Adding genericity to the Java programming language. In *OOPSLA 1998*, pages 183–200. Sigplan, 1998.

[Bri05] B. Bringert. HOJ - higher-order Java, 2005. cs.chalmers.se/bringert/hoj.

[Col09] S. Colebourne. Closures in JDK7, 2009. //www.jroller.com/_scolebourne/entry /more_detail_on_closure_in.

[Cor04] Microsoft Corporation. Delegates in visual J++, 2004. //msdn.microsoft.com-/vjsh arp/productinfo/visualj/visualj6/technical/articles/general/delegates/default.aspx.

[CS06] S. Colebourne and S. Shulz. First_class methods: Java style closures, 2006. //-docs.google.com /view?docid=ddhp95vd_6hg3qhc.

[Fri87] M. Felleisen D.P. Friedman. A reduction semantics for imperative high-order languages. In *PARLE Parallel Architectures and Languages Europe*, volume 2007 of *LNCS*, pages 206–223. Springer, 1987.

[GHJV05] E. Gamma, R. Helm, R. Johnson, and J.M. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 2005.

[GA09] N.M. Gafter and P. von der Ahe. Closures for java (v0.6a), 2009. //javac.info /closures-v06a.html.

[IP02] A. Igarashi and B. Pierce. On inner classes. *Information and Computation*, 77(1):56–89, 2002.

[IPW99] A. Igarashi, B. Pierce, and P. Wadler. Featherweight Java: A minimal core calculus for Java and GJ. *Sigplan Notices*, 34(10):132–146, 1999.

[IPW01] A. Igarashi, B. Pierce, and P. Wadler. Featherweight java: A minimal core calculus for Java and GJ. *ACM TOPLAS*, 23:396–450, 2001.

[LLB06] B. Lee, D. Lea, and J. Bloch. Concise instance creation expressions: Closure without complexity, 2006. crazybob.org/2006/10/java-closure-spectrum.html.

[McC01] G. McCluskey. Using method pointers and abstract classes vs interfaces. *Electronic Notes TCS*, 2001. //java.sun.com/developer/JDCTechTips-/2001/tt1106.html.

[Mei01] E. Meijer. Lambada, Haskell as a better Java. *Electronic Notes TCS*, 41(1), 2001.

[Mic04] Sun Microsystems. About Microsoft's delegates, 2004. java.sun.com/docs /white/delegates.html.

[OW97] M. Odersky and P. Wadler. Pizza into Java: translating theory into practice. In *Proc. 24th Symposium on Principles of Programming Languages*, pages 146–159, 1997.

[Rei09] M. Reinhold. Project lambda: Straw-man proposal, 2009. //cr.openjdk.java.net/ mr/lambda/straw-man/.

[Set03] A. Setzer. Java as a functional programming language. In *TYPES 2002,LNCS 2646.*, pages 279–298, 2003.

[SS07] N. Sridranop and R. Stansifer. Higher-order functional programming and wildcards in java. In *ACMSE 2007, ACM*, pages 42–46, 2007.