There is an error in

The definition of simple closures on page 5 is too strict; the concurrent set example of Figure 12 does not comply with it.

Therefore, on p. 5, instead of

We say a program has simple closures if there exists a partitioning of the procedures into ghost procedures and non-ghost procedures such that command expressions and ghost procedures contain no exec commands and no calls of non-ghost procedures. It follows that closures are never recursive. Applying the specification approach of this paper requires only simple closures. As we will see, simple closures admit a very simple proof system.

and, further down the column, instead of

This simple approach is sufficient for programs where the graph of procedure calls and closure executions is acyclic. This is the case if the program has simple closures and the procedure call graph is acyclic. The examples of this paper satisfy these constraints.

Second, to prove well-definedness of function valid, in the Coq proof we pass an explicit closure level parameter, which is a natural number that decreases at each exec command. Correspondingly, the correctness judgment mentions a closure level; $L, \Gamma \vdash \{P\} c \{Q\}$ means that command $c$ is correct with respect to precondition $P$, postcondition $Q$, and function environment $\Gamma$, and closure executions in $c$ are nested no more than $L$ levels deep. The correspondence with the approach of the paper is as follows: the closure level of a main command is the maximum static nesting depth of exec commands after inlining all procedure calls.

The corrected version has been published as