Gradual typing is morally incorrect; we're all monsters now

Timothy Jones
Victoria University of Wellington
tim@ecs.vuw.ac.nz

Michael Homer
Victoria University of Wellington
mwh@ecs.vuw.ac.nz

The aspiration of gradual typing [4] — that typed and untyped code can coexist happily, with errors reported at runtime when the types are found to be wrong — has led many languages, including our own, to adopt the paradigm. The practice is different; even the term itself has been refined short of this aspiration [5], and languages abdicate their responsibilities as soon as they become too onerous: when an object reënters dynamically-typed code, in the presence of aliasing, or with regard to secure object identity.

Gradual typing that uses proxies either does not retain object identity — breaking aliasing — or introduces a mechanism to avoid it [6], breaking security guarantees. When the object passes back to the dynamic world, the proxy is often removed, rendering the type assertion meaningless.

If instead object identities are mapped to sets of contracts, which the monotonic semantics [7] approximates, we must update these contracts as new information refines them. These updates are visible and have a semantic effect on type tests or even the ability to call a typed function. If we do not update in every instance where new information is gained then we instead commit to actions we know to be wrong.

Extra information about the type of an object is gained in many more places than just when the object passes through a type assertion. When an object's 'foo' method returns a string, then it should fail any future assertion that foo returns a number, even if such a property holds true from that point on. Unions of types must also collapse to to a smaller type when one of the members of the union is proved false, and as a result any update of type information can potentially have a cascading effect through the current state of the program.

If runtime type errors are catchable, the program behaviour can even change based on the inferred type information. Gradual type errors must be uncatchably fatal in order to conform to the refined criteria for gradual typing [5], as a complete (though not necessarily sound) type test can be recovered from whether such an error appears as the result of an assertion (or a probe of higher-order contract). Fatal errors are provided for by the theory [8], but in practice much effort has instead been made to ensure that the *right kind* of error is raised precisely for the purpose of being caught by the appropriate guard [7].

Each different interpretation of gradual typing ascribes different spatial and temporal meaning to their types, but the literature has tended to only focus on soundness within that particular ascription and the performance differences of the resulting implementations. The clearly-correct thing to do is to ensure that incorrect code does not execute to begin with: when a type assumption is shown unfounded, unexecute the code and reverse the assumption [1]. Unfortunately, in the general case this option only works at 88mph [3].

A type assertion should be meaningful. If the assertion is known to be false, an error should be reported. Aliases of the same object ascribed different types, the collapse of variant or generic types, objects passed to dynamically-typed code with the understanding that they had a particular type, and even past behaviour, are all routes through which an assertion can be falsified, but all practical gradual systems ignore some or all of these cases. The level of knowledge the system has can change behaviour, and the only morally correct [2] behaviour is to retain an impossibly complete level of information on the types of objects.

References

[1] BENTON, N. Undoing dynamic typing. In *FLOPS* (2008), Springer-Verlag, pp. 224–238.

[2] DANIELSSON, N. A., HUGHES, J., JANSSON, P., AND GIBBONS, J. Fast and loose reasoning is morally correct. In *POPL* (January 2006), pp. 206–217.

[3] NORTH, R., AND GIPE, G. *BˆF: Steven Spielberg Presents: Back To The Future: A Robert Zemeckis Film: The Novel by George Gipe based on a screenplay by Robert Zemeckis and Bob Gale: Reviewed by Ryan North*. 2012.

[4] SIEK, J., AND TAHA, W. Gradual typing for objects. In *ECOOP* (2007), Springer-Verlag, pp. 2–27.

[5] SIEK, J. G., VITOUSEK, M. M., CIMINI, M., AND BOYLAND, J. T. Refined criteria for gradual typing. In *SNAPL* (May 2015), pp. 274–293.

[6] STRICKLAND, T. S., TOBIN-HOCHSTADT, S., FINDLER, R. B., AND FLATT, M. Chaperones and impersonators: runtime support for reasonable interposition. In *OOPSLA* (October 2012), pp. 943–962.

[7] VITOUSEK, M. M., KENT, A. M., SIEK, J. G., AND BAKER, J. Design and evaluation of gradual typing for Python. In *DLS* (October 2014), pp. 45–56.

[8] WADLER, P., AND FINDLER, R. B. Well-typed programs can't be blamed. In *ESOP* (March 2009), pp. 1–16.