Errata to "Atomic Shared Register Access by asynchronous Hardware," FOCS '86.

Paul M.B. Vitányi
CWI, Amsterdam

Baruch Awerbuch
M.I.T., Cambridge, Mass.

The above paper [VA] has the following error in the 'Struldbrugg' algorithm (the main bounded tag algorithm). In the following, we assume familiarity with the paper. The "selection rule" of the reader is not sufficient to guarantee atomicity, but only the weaker regularity condition. Informally, regularity for multiwriter registers means that a reader does return a value written by a write, such that there is no complete other write strictly in between the durations of the read and the write it returns. See [L] or [AK] for the definitions.

Here is the counterexample. Let the selection rule for readers be to select the least indexed writer which is not seen by other writers. (In the paper we used the greatest index.)

1. Writers 1,2,3 do initial writes (doesn't matter in which order) and write values a,b,c, respectively.
2. Writer 1 starts write and reads values b and c, and gets stuck.
3. Writer 2 starts write and reads values a and c, and gets stuck.
4. Writer 3 starts write and reads value b, and gets stuck.
5. Writers 1 and 2 terminate and write values x and y, respectively.
6. The reader executes the first read and reads values x,y,c. It now chooses x, because both x and y have seen c, have not seen each other, and x is by writer 1.
7. Writer 3 reads value x and terminates, and writes value z.
8. The reader executes a second read, and now reads values x,y,z. Now, x,y have not seen z, and z has seen x but not y. So the reader selects between y and z and chooses the value (y) by the least indexed writer, which is 2. (Oops!)

This is not atomic run, since there exists no shrinking mapping consistent with the reading mapping. Informally, after both writer 1 and 2 terminated, two subsequent nonoverlapping reads returned first the value written by 1 and then the value written by 2, while after both writers were terminated the compound register should contain either the value of 1 or the value of 2. The proof of atomicity in the paper breaks down in Lemma 7, in the proof that atomicity condition (P1) b) holds for the Struldbrugg algorithm. There, the conclusion that the constructed action digraph has no directed cycles is false.

A similar example was pointed out to us by G. Peterson. We have ideas how to fix the bug (staying close to the original version), but have not worked out the details yet. G. Peterson and J. Burns have found an alternative way of getting rid of this problem. They replace the 'send-receive mechanism' of Struldbrugg by an 'atomic scan' and refine the selection rule in our paper, see this Proceedings.

References

[VA] P.M.B. Vitanyi and B. Awerbuch, Atomic shared register access by asynchronous hardware, Proc. 27th IEEE Symp. on Foundations of Computer Science (1986), 233-243.

[L] L. Lamport, On interprocess communication, *Distributed Computing* 1(1986), 77-101.

[AK] B. Awerbuch, L.M. Kirousis, E. Kranakis, P.M.B. Vitányi, On proving register atomicity, Tech. Rept. CS-R8707, CWI, May 1987 (Submitted).