Erratum for: "Help!", PODC 2015

May 25, 2021

The proceedings version of the paper [2] contains an error in the definition of Exact Order Types. The original definition is such that it excludes the stack type. This was pointed out in [1]. This erratum contains the correct definition that should be used. The proof that a wait-free linearizable implementation of an Exact Order Type using READ, WRITE and CAS cannot be help-free should be fixed accordingly. In this erratum, we illustrate how to adjust the proof as well.

Definition 1 (Exact Order Types – original definition.) *An exact order type t is a type for which there exists an operation op, an infinite sequence of operations W, and a (finite or an infinite) sequence of operations R, such that for every integer $n \geq 0$ there exists an integer $m \geq 1$, such that for at least one operation in $R(m)$, any response it returns in a sequential execution of any sequence in $W(n+1) \circ (R(m) + op?)$ differs from any response it returns in a sequential execution of any sequence in $W(n) \circ op \circ (R(m) + W_{n+1}?)$.*

Definition 2 (Exact Order Types – correct definition.) *An exact order type t is a type for which there exists an operation op, an infinite sequence of operations W, and a (finite or an infinite) sequence of operations R, such that for every integer $n \geq 0$ there exists an integer $m \geq 1$, such that the following holds. Consider the set of two sequences $W(n+1) \circ op? \circ R(m)$ and the set of two sequences $W(n) \circ op \circ W_{n+1}? \circ R(m)$, and for each possible sequence s denote by $resR(s)$ the sequence of responses to the operations in R(m), when s is executed sequentially. Then we require that the set $\{resR(s) \mid s \in W(n+1) \circ op? \circ R(m)\}$ is disjoint from the set $\{resR(s) \mid s \in W(n) \circ op \circ W_{n+1}? \circ R(m)\}$.*

All the claims and theorems in paper [2] hold with regard to the new and correct definition as well. However, the proofs of Claim 4.2 and Claim 4.3 are more complex and require additional logic. We now illustrate how to adjust the proof of Claim 4.2 to the new definition. Recall that Q is a linearizable, help-free implementation of an Exact Order Type.

Claim 4.2 (Unchanged): Let h be a history in H on Q such that the first n operations are decided to be the first n operations of p_2 (which are $W(n)$), and p_3 has not yet taken any step. Formally, in h, for every $1 \leq i < j \leq n$, the ith operation of p_2 is decided before the jth operation of p_2, and before any operation of the other processes (p_1 and p_3). Denote the $(n+1)$-st operation of p_2 by op_2.

1. If in h, op_1 is decided before op_3, then the order between op_1 and op_2 is decided.
2. Similarly, if in h, op_2 is decided before op_3, then the order between op_1 and op_2 is decided.

The proof starts the same as before: For convenience, we prove (1). The proof of (2) is symmetrical. Assume that in h, op_1 is decided before op_3, and let m be the integer corresponding to n by the definition of exact order types. Immediately after h, let p_3 run in a solo execution until it completes exactly m operations. Denote the history after this solo execution of p_3 by h', and consider the linearization of h'. Notice that op_2, which is W_{n+1}, may or may not be a part of h'. The first n operations in the linearization must be $W(n)$. The linearization must also include exactly m operations of p_3 (which are $R(m)$), and somewhere before them, it must also include op_1. The linearization may or may not include op_2. There are two cases. If the $(n+1)$-st operation in the linearization is op_1, then the linearization is in $W(n) \circ op_1 \circ (R(m) + W_{n+1}?)$, while if the $(n+1)$-st operation in the linearization is op_2, then the linearization must be exactly $W(n+1) \circ op_1 \circ R(m)$.

At this point, an adjustment to the proof is required. The crux is to argue that if the $(n+1)$-st operation in the linearization is op_1, and thus the linearization is in $W(n) \circ op_1 \circ (R(m) + W_{n+1}?)$, then for the results of the operations $R(m)$, denoted $resR$, it holds that $resR \in \{resR(s) | s \in W(n) \circ op_1 \circ W_{n+1}? \circ R(m)\}$. The argument is as follows: If the linearization itself is in $W(n) \circ op_1 \circ W_{n+1}? \circ R(m)$, then it is immediate that $resR \in \{resR(s) | s \in W(n) \circ op_1 \circ W_{n+1}? \circ R(m)\}$. Otherwise, there must be some $1 \leq q \leq m$ such that the linearization is: $W(n) \circ op_1 \circ R(q) \circ W(n+1) \circ R(q+1...m)$. It follows by Definition 3.2 that in history h, W_{n+1} is not decided before R_1. Thus, by Claim 3.6, in h, W_{n+1} is not decided before any future operation, and in particular, W_{n+1} is not decided before R_m. Next, since Q is help free and since p_2 did not take any step between h and h', it follows that W_{n+1} is not decided before R_m in h' as well. It follows by Definition 3.2 that there exists a continuation of h', denoted h'', in which R_m is before W_{n+1}. Thus, h'' starts with: $W(n) \circ op_1 \circ R(m)$, and therefore the results of operations $R(m)$ in h'' belongs to $\{resR(s) | s \in W(n) \circ op_1 \circ W_{n+1}? \circ R(m)\}$. Now, in h' all operations $R(m)$ are already completed, so their results must be the same in h' and in h'', and thus $resR \in \{resR(s) | s \in W(n) \circ op_1 \circ W_{n+1}? \circ R(m)\}$. We have obtained that if the $(n+1)$-st operation in the linearization of h' is op_1 then the results $resR \in \{resR(s) | s \in W(n) \circ op_1 \circ W_{n+1}? \circ R(m)\}$, and if the $(n+1)$-st operation in the linearization is W_{n+1} then the results $resR \in \{resR(s) | s \in W(n+1) \circ op_1? \circ R(m)\}$. By the (corrected) definition of Exact Order Types, these two sets are disjoint.

The proof now continues similarly as in the original paper to show that the order between op_1 and in op_2 is decided.

The adjustments to the proof of Claim 4.3 are similar in nature: when we show a history has a linearization which is in $W(n+1) \circ (R(m) + op_1?)$(alternatively, $W(n) \circ op_1 \circ (R(m) + W_{n+1}?)$), we use the same logic as above to show that the results of operations $R(m)$ are in fact in $\{resR(s) | s \in W(n+1) \circ op_1? \circ R(m)\}$ (alternatively, $\{resR(s) | s \in W(n) \circ op_1 \circ W_{n+1}? \circ R(m)\}$).

References

[1] Vitaly Aksenov, Petr Kuznetsov, and Anatoly Shalyto. On helping and stacks. In *Networked Systems - 6th International Conference, NETYS 2018, Essaouira, Morocco, May 9-11, 2018, Revised Selected Papers*, pages 107–121, 2018.

[2] Keren Censor-Hillel, Erez Petrank, and Shahar Timnat. Help! In Chryssis Georgiou and Paul G. Spirakis, editors, *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing, PODC 2015, Donostia-San Sebastián, Spain, July 21 - 23, 2015*, pages 241–250. ACM, 2015.