ABA Prevention Using Single-Word Instructions

Maged M. Michael

July 1, 2024

1 Introduction

The ABA problem is a fundamental problem that affects almost all lock-free algorithms. The atomic primitives LL/SC/VL (Load-Linked, Store-Conditional, Validate) offer a convenient way for algorithm designers to reason about lock-free algorithms, without concern for the ABA problem. However, for practical architectural reasons, no processor architecture supports the ideal semantics of LL/VL/SC.

This report presents simple lock-free constructions using only practical single-word instructions for implementing ideal LL/SC/VL, and hence preventing the ABA problem, with reasonable space overhead.

```
1  // Shared variables
2  NodeType *Top; // Initially null
3
4  void Push(NodeType *node) {
5      NodeType *t;
6      do {
7          t = Top;
8          node->Next = t;
9      } while (!CAS(&Top, t, node));
10 }
11
12 NodeType *Pop() {
13     NodeType *t, *next;
14     do {
15         t = Top;
16         if (!t) return null;
17         next = t->Next;
18     } while (!CAS(&Top, t, next));
```

```
19      return t;
20  }
```

Consider a list that contains three nodes A, B and C. Thread X reads

2 Problems

3 References

References