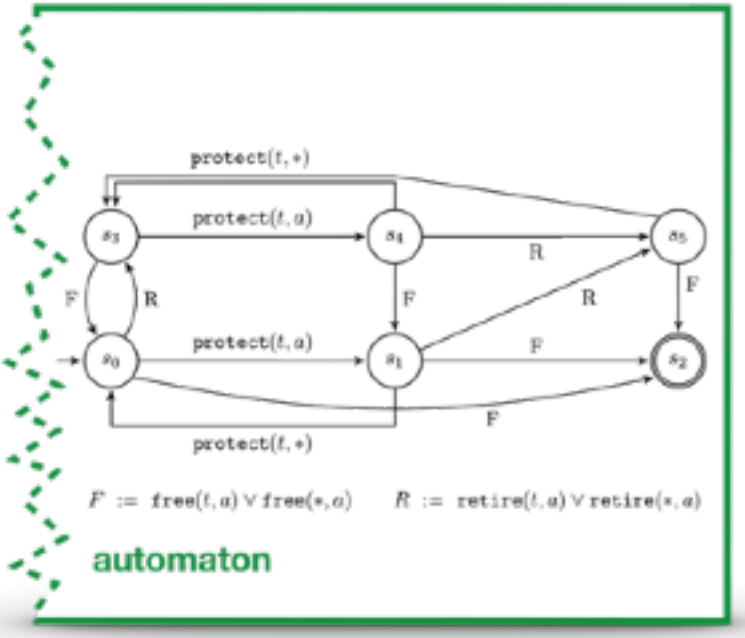
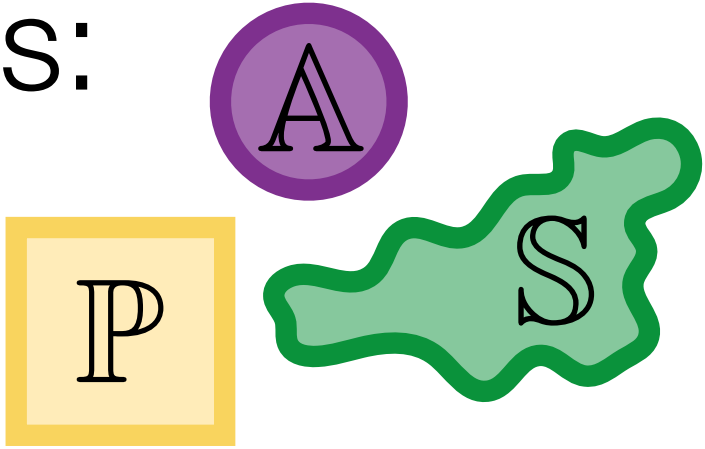
off-the-shelf verifier

Final Approach


protect(t, ∗)
protect(t, a)
protect(t, a)
protect(t, ∗)
R
F
R
F
R
F
F
F := free(t, a) ∨ free(∗, a)
R := retire(t, a) ∨ retire(∗, a)
automaton

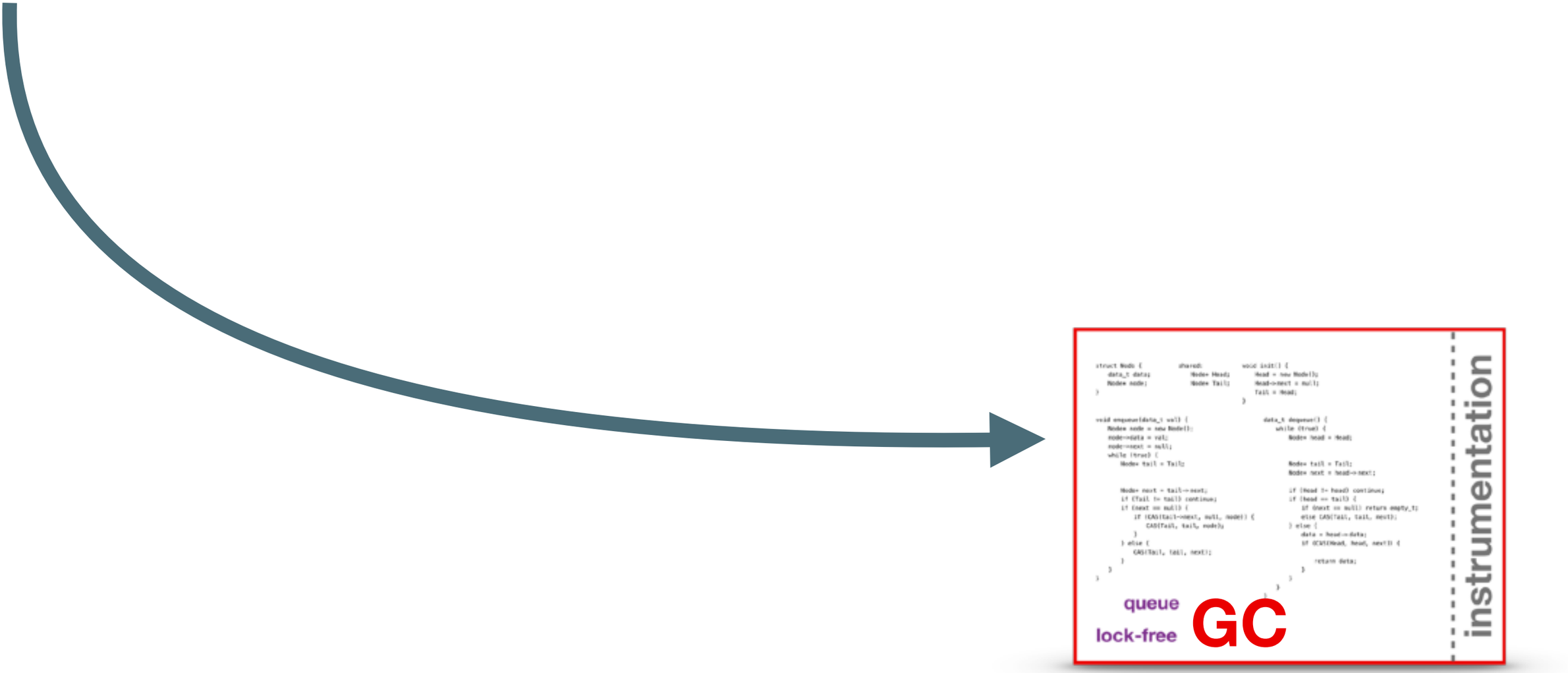
Types:



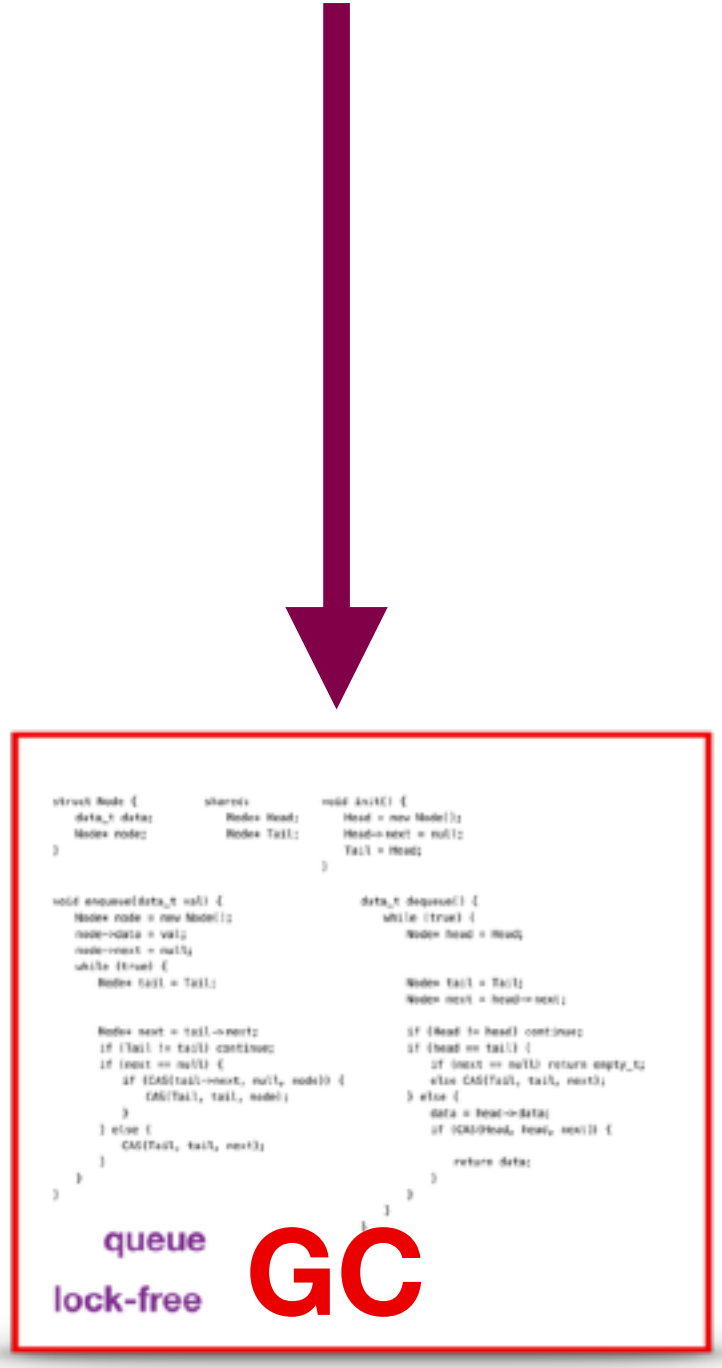

Type Check


Verification


Annotation
Check


struct Node {
data_t data;
Node* node;
}
shared:
Node* Head;
Node* Tail;
void init() {
Head = new Node();
Head->next = null;
Tail = Head;
}
void enqueue(data_t val) {
Node* node = new Node();
node->data = val;
node->next = null;
while (true) {
Node* tail = Tail;
Node* next = tail->next;
if (tail != tail) continue;
if (next == null) {
if (CAS(tail->next, null, node)) {
CAS(Tail, tail, node);
}
} else {
CAS(Tail, tail, next);
}
}
}
data_t dequeue() {
while (true) {
Node* head = Head;
Node* tail = Tail;
Node* next = head->next;
if (head != head) continue;
if (head == tail) {
if (next == null) return empty_t;
else CAS(Tail, tail, next);
} else {
data = head->data;
if (CAS(Head, head, next)) {
return data;
}
}
}
queue
lock-free
GC
instrumentation

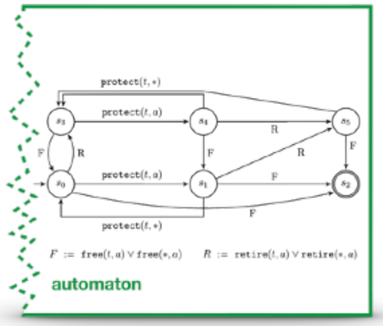

queue
lock-free
SMR
lock-free
HP
no reclamation


queue
lock-free
GC

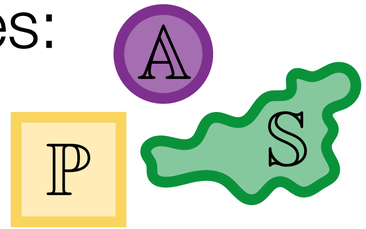

Thanks

Final Approach

Thanks



Types:



Type Check

Verification

Annotation Check

off-the-shelf verifier

Fin.