### Homework 7 solution

(30pts) Register Allocation

b. (20p) Exercise 11.1 in the textbook

c. (10p) Exercises 11.2 (a) in the textbook

a. 11.1

Note that although r2 is not used in the original program. Interference edges about r2 must be inserted as long as it is a caller save register.

Cannot simplify any non-move relevant nodes.
Try to coalesce. According to Briggs, none of the pairs can be coalesced. Consider George, u and r1 can be merged. **Note that when applying George to pairs involving a pre-colored node, always pick the one that is not pre-colored to test the rule.** In this case, considering u, its neighbor c is also the neighbor of r1. So u and r1 can be merged.

Node “r1 & u” is further merged with t according to George.

c and r3 cannot be merged (when applying George on c).

We cannot proceed with the remaining graph, even after freezing the edge c-r3.

We hence look for a node to spill. According to the equation, node c has the lowest spill cost. After removing c, the remaining graph is still not colorable. We further spill s. Finally, p has r3, and u, t having r1.

The final program is

\[
\begin{align*}
f & : M[\text{address for } c] \leftarrow r3 \\
r3 & \leftarrow r1 \\
\text{if } (r3=0) & \text{ goto L1} \\
r1 & \leftarrow M[r3] \\
call f \\
M[\text{address for } s] & \leftarrow r1
\end{align*}
\]
r1<-M[r3+4]
call f
r1<-M[address for s]+r1
goto L2

L1: r1<-1
r3<-M[addr of c]
return

c.
Exercise 11.2 (a)
Simplify:
Remove A
Remove B
Remove G
All the remaining nodes have degree of at least 8.
Randomly pick up the node for potential spill.
Spill
Remove H
Simplify
Remove D
Remove E
Remove F
Remove C
Select:
Add C with color 1
Add F with color 5
Add E with color 4
Add D with color 2
Add H with color 3
Add G with color 3
Add B with color 2

Add A with color 7

The nodes are 8 colorable. No actual spill is required.

(30p) Garbage collection.

Apply the Mark-Sweep (10p) and Copying (20p) GC algorithms to the following program at the end of the execution
class B{
    int val;
    B next;
    static B gp;
}
...
F () {
    B head= new B(0, nil);
    B tail = head;
    for (int i=1; i< 5; i++) {
        tail.next=new B(i, nil);
        tail=tail.next;
    }
    head=head->next->next;
    gp=head;
    head=head->next;
}

Mark and Sweep: The plain nodes are garbage collected.
Heap 1

After forwarding all roots

Heap 1

After scanning and forwarding all the pointers in Heap 2, the entire Heap 1 will be discarded.

Heap 1