1996 Paper 8 Question 7

Optimising Compilers

Explain what is meant by *register allocation by colouring* from the point of view of a compiler-writer (do not give theory, but explain principal concepts and sketch central algorithms). You may assume that the input is in three-address instruction form, but state any presumed properties on temporaries; moreover indicate carefully what form the output from the register allocation phase takes, particularly occurrences of user-variables. [8 marks]

For a machine with n registers give a program which results in not all the (non-address taken) user-variables being allocatable to registers. [2 marks]

Consider compiling the following source file of a program

```
extern void p(void), q(int,int);
extern int f(int);
void proc()
{   int a,b,c;
    a = f(1), b = f(2); p(); q(a,b);
    b = f(3), c = f(4); p(); q(b,c);
    c = f(5), a = f(6); p(); q(c,a);
}
```

on a machine which has eight registers preserved over function call and sufficient temporaries and argument registers corrupted over function call available for allocation. Clearly the code can be compiled using three registers, one for `a`, `b` and `c`. However, it is obvious to any assembly language programmer how two registers suffice. Suggest how your answer to the first part of this question could be adapted to compile the above code better.

[Hint: the usual phrase for this concept is *variable splitting according to live ranges.* You may wish to consider general flowgraphs after considering basic blocks.] [6 marks]

Why might the above idea be helpful in general? Would it affect the resulting code speed, resulting code size or the compilation time? Would it improve the final machine code for `proc` on a register-based architecture of your choice? [4 marks]