Allocation

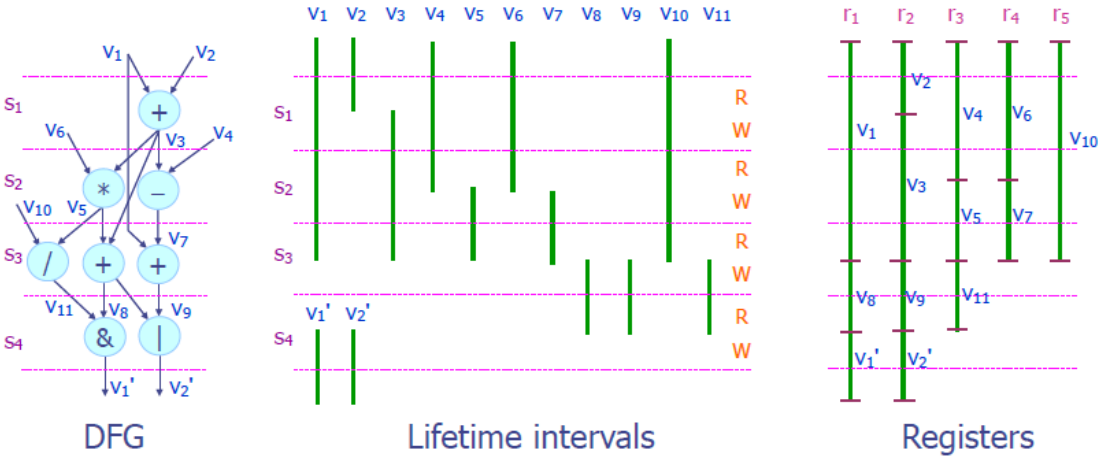
2017/5/16

范真瑋

Left-Edge Algorithm

實驗內容：

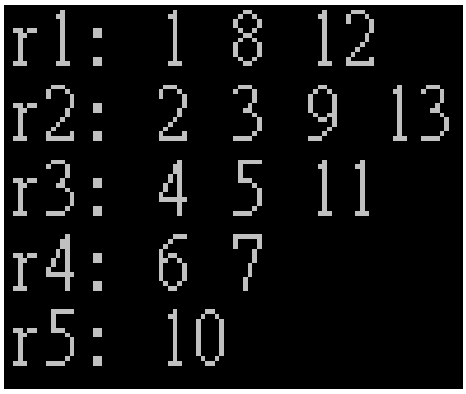
- 撰寫Left-Edge Algorithm的C/C++程式
  - Left-Edge Algorithm for Register Binding



程式碼：

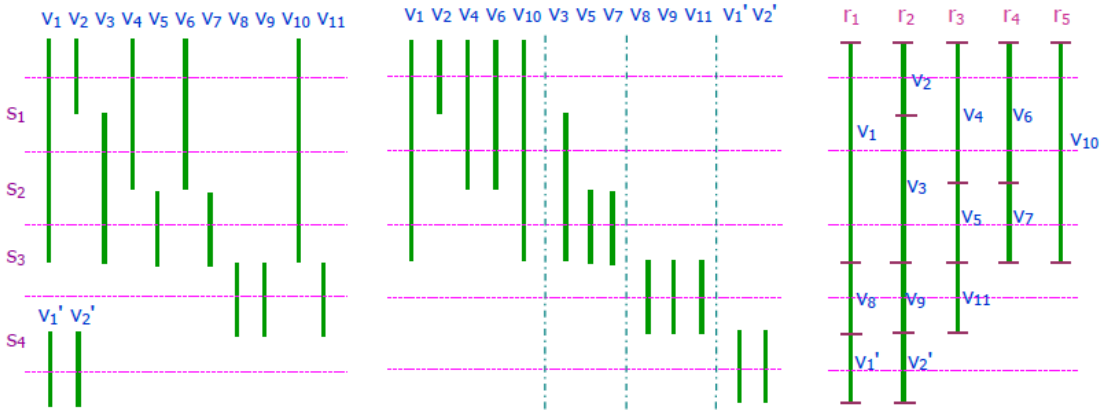
```
43      //Left-Edge Algorithm
44      vector<int> MAP(n);
45      int reg_index = 0;
46      int last;
47      while(L.size()) {
48          ++reg_index;
49          last = 0;
50          for(i = 0; i < L.size(); ++i) {
51              if(L[i].start >= last) {
52                  MAP[L[i].n-1] = reg_index;
53                  last = L[i].end;
54                  L.erase(L.begin()+i);
55              }
56          }
57      }
```

實驗結果及分析：



Left-Edge Algorithm (2/3)

```
for all v ∈ L do MAP[v]=0; endfor
SORT(L);    /* sort the variables in L in acending order with their start times*/
reg_index = 0;
while L ≠ ∅ do
    reg_index = reg_index + 1;
    curr_var = FIRST(L);
    last = 0;
    while curr_var ≠ null do
        if Start(curr_var) ≥ last then
            /* share the register */
            MAP[curr_var] = reg_index;
            last = End(curr_var);
            temp_var = curr_var;
            curr_var = NEXT(L, curr_var);
            DELETE(L, temp_var);
        else
            curr_var = NEXT(L, curr_var);
        endif
    endwhile
endwhile
```



首先依開始時間由小排到大，之後將開始時間小的排在前面(排入 x)，往後尋找開始時間大於或等於 x 的結束時間，後面依此類推，把全部排完。