Page No.
Date

Sahil Jindal 18CS10048

Compilers Assignment 7

// Peep Optimization

1. // int n = 10, i;

100 : t1 = 10 XXX
100: 101 : n = ~~t1~~ 10 // For (i = 0; i<n; i++) {
102 : t2 = 0 X XX
101: 103 : i = ~~t2~~ 0 <== def use
102: 104 : if i<n go to 109 // True
103: 105 : go to 125 // False
106 : t3 = i <= = unused XXX
104: 107 : i = ~~t3+1~~ i+1
105: 108 : go to 104 // if (a[i]%2 == 0) {
106: 109 : t4 = 4*i // strength reduction
107: 110 : t5 = a[t4]
108: 111 : t6 = t5%2
109: 112 : t7 = 0 <= ~~[illegible]~~ jump over jump
110: 113 : if t6 != t7 go to ~~115~~ 120 // T
114 : go to 120 XXX // F
111: 115 : t8 = 4*i // c[i] = 0 // strength reduction
112: 116 : t9 = c + t8
117 : t10 = 0 XXX
113: 118 : t9 = ~~t10~~ 0 <== def use
114: 119 : go to 106 // next // c[i] = 1
115: 120 : t11 = 4*i // strength reduction
116: 121 : t12 = c + t11
122 : t13 = 1 XXX
117: 123 : t12 = ~~t13~~ 1 // } <= def use
118: 124 : go to 106 // for
119: 125 : return // return

Optimised Code after Peephole Opt.

New Quad No.		
	100: t1 = 10	xxx
100	101: n = 10	<== def use
	102: t2 = 0	6: xxx
101	103: i = 0	<== def use
102	104: if i < n go to ~~104~~ 106	
103	105: go to ~~125~~ 119	
	106: t3 = i	<== Unused xxx
104	107: i = i+1	
105	108: go to ~~104~~ 102	
106	109: t4 = i << 2	// Strength reduction
107	110: t5 = a[t4]	
108	111: t6 = t5 % 2	
109	112: t7 = 0	
110	113: if t6 != t7 go to ~~120~~ 115	<== jump over jump
	114: go to 125	xxx
111	115: t8 = i << 2	// strength reduction
112	116: t9 = c + t8	
	117: t10 = 0	xxx
113	118: t9 = 0	<== def use
114	119: go to ~~106~~ 104	
115	120: t11 = i << 2	// strength reduction
116	121: t12 = c + t11	
	122: t13 = 1	xxx
117	123: t12 = 1	<== def use
118	124: go to ~~106~~ 104	
119	125: return	

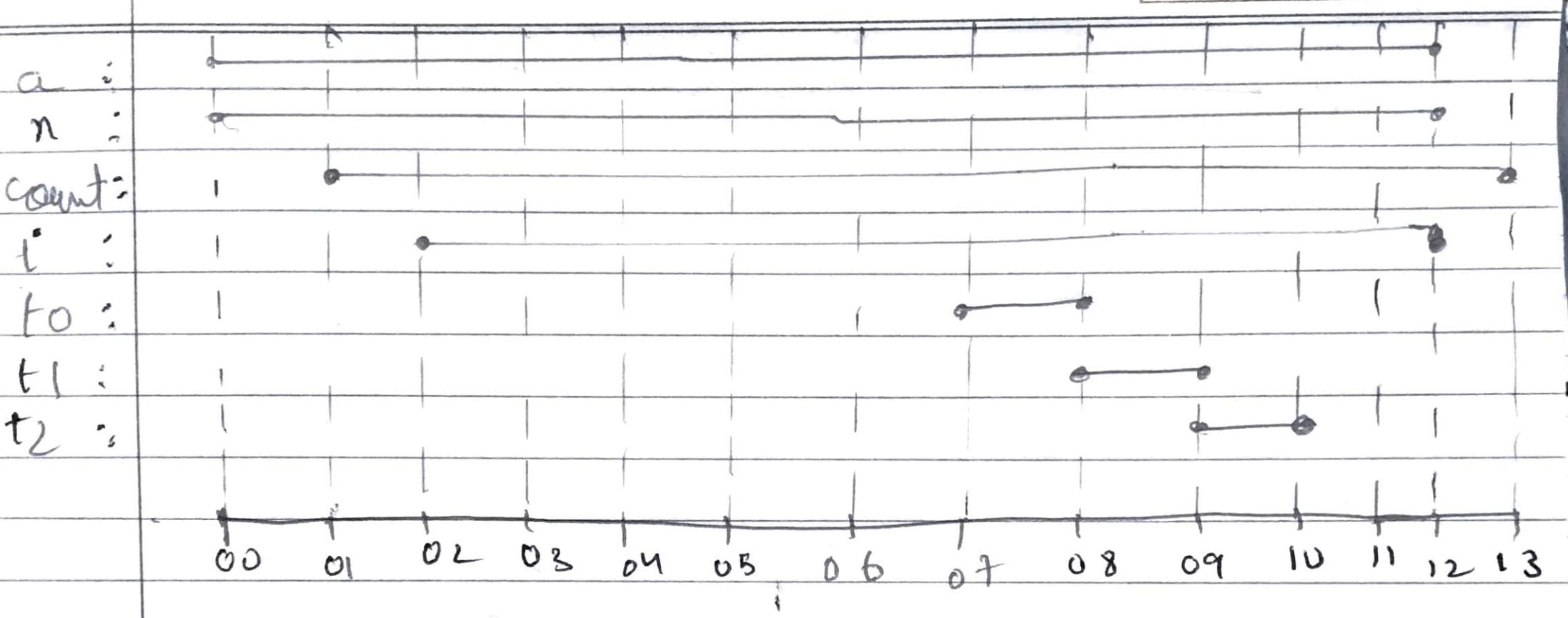
Date

Q2 (a): • push ebp → push old base pointer onto stack.
• mov ebp, esp → assign ebp = esp to create a new stack frame on top of old frame.
• sub esp, 12 → decrease stackpointer to store local variables of function (if any).
• push esi → save registers into stack.

(b) • pop esi → restore register from stack
• mov esp, ebp → restore stack pointer
• pop ebp → pop base pointer off the stack
• ret 0 → return to calling function.

Q3

```
000 :                          //a, n
001 :        count = 0         //a, n, count
002 :        i = 0             //a, n, count, i
003 : L0:    if i<n goto L2    //a, n, count, i
004 :        goto L3           //a, n, count, i
005 : L1:    i = i+1           //a, n, count, i
006 :        goto L0           //a, n, count, i
007 : L2:    t0 = 4*i          //a, n, count, i, t0
008 :        t1 = a[t0]        //a, n, count, i, t0, t1
009 :        t2 = t1 % 2       //a, n, count, i, t1, t2
010 :        if t2 != 0 goto L1 //a, n, count, i, t2
011 :        count = count + 1 //a, n, count, i
012 :        go to L1          //a, n, count, i
013 : L3:    return count      // count.
```



Interval Graph