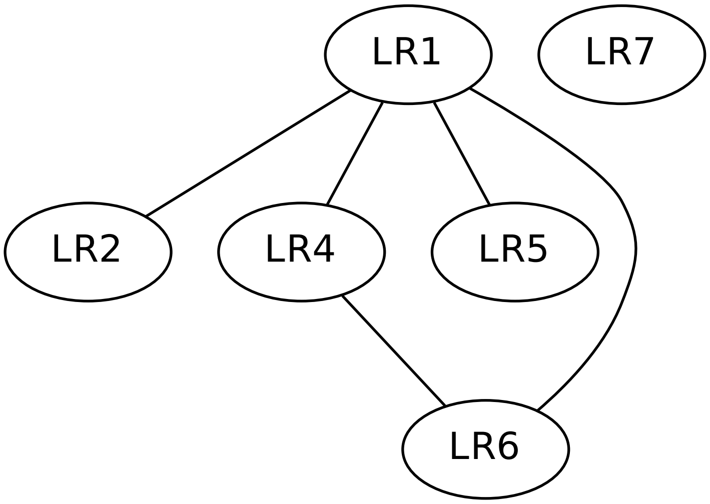
Homework 8 Solutions

1 Register Allocation

1.1 Interference Graph



1.2 Spill Costs

LR1	LR2	LR4	LR5	LR6	LR7
3	∞	2	∞	∞	∞

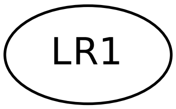
1.3 Chaitin-Briggs

LR7	unconstrained	any order
LR2	unconstrained	
LR5	unconstrained	
LR4	1	(LR1 = 3/2, LR6 = ∞/2)
LR6	unconstrained	any order
LR1	unconstrained	

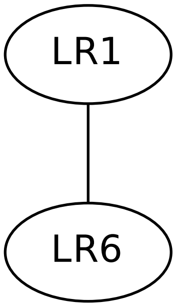
1.4 Ordered Live Ranges

LR1, LR6, LR4, LR5, LR2, LR7

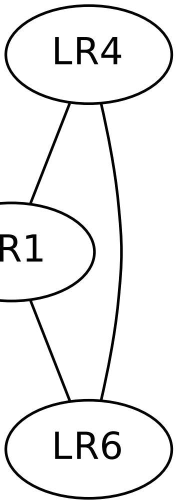
1.5 Chaitin-Briggs



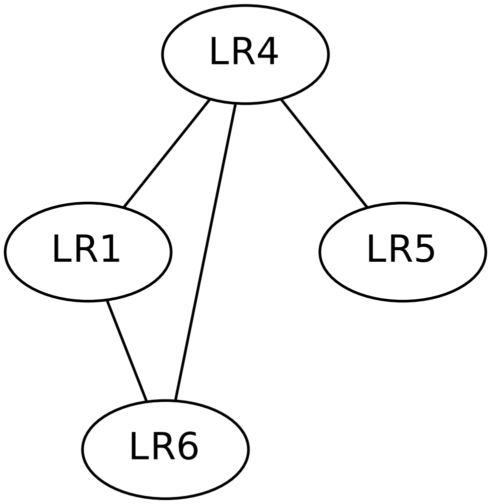
1. LR1 ($t0)



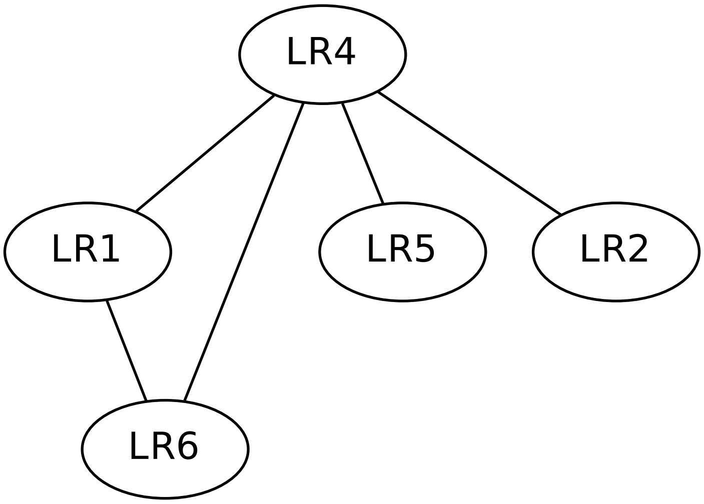
2. LR6 ($t1)



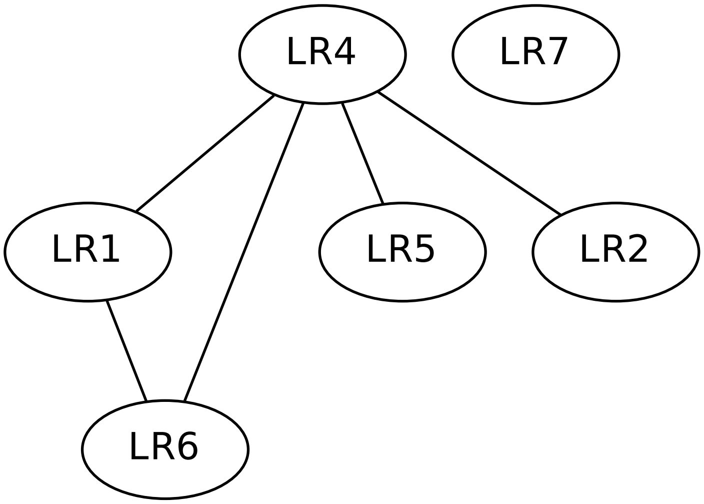
3. LR4 (uncolored)



4. LR5 ($t1)



5. LR2 ($t1)



6. LR7 ($t0/$t1)

1.6 Rewrite

```
...
L2:
loadI 9 => LR8
storeAI LR8 => rarp, -24
addI LR1, LR8 => LR6
storeAI LR6 => rarp, -20
loadAI rarp, -24 => LR9
writeInt LR9
jump L3
...
```

1.7 Register Rewrite

```
L0:
loadI 1 => $t0
cbr $t0 -> L1, L2

L1:
loadI 5 => $t1
addI $t1, 7 => $t1
storeAI $t1 => rarp, -20
jumpI -> L3

L2:
loadI 9 => $t1
storeAI $t1 => rarp, -24
addI $t0, $t1 => $t1
storeAI $t1 => rarp, -20
loadAI rarp, -24 => $t1
writeInt $t1
jump L3

L3:
writeInt $t0
loadAI rarp, -20 => $t0
writeInt $t0
exit
```

1.8 Stopping Condition

Chaitin-Briggs algorithm stops when all live-ranges are colored, or coloring is not possible.

2 Dataflow Analysis

2.1 Partition

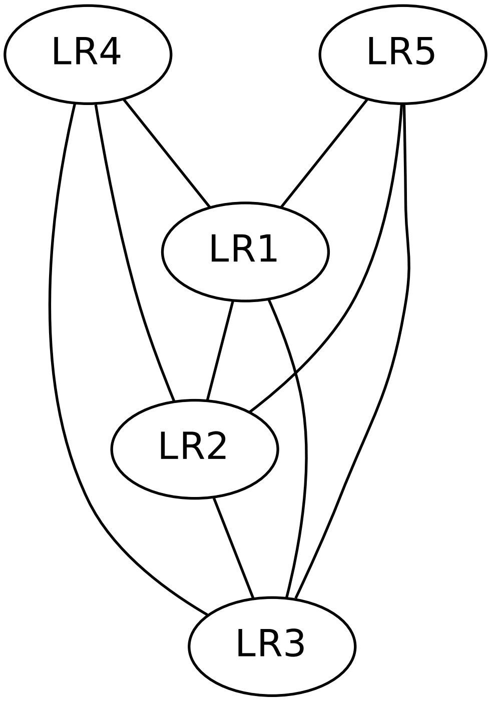
2.2 Rewrite

```
START:
loadI 0 => LR1
loadI 0 -> LR2
loadI 10 => LR3
jumpI -> L1

L1:
cmp_LT LR1, LR3 => LR4
cbr LR4 -> L2, L3

L2:
loadI 1 => LR5
addI LR1, LR5 => LR1
add LR2, LR1 => LR6
jumpI -> L1

L3:
 writeInt LR2
 exit
```



2.3 Find LiveOut

2.3.1 VARKILL and UEVar

	START	L1	L2	L3
VARKILL	LR1, LR2, LR3	LR4	LR1, LR2	∅
UEVar	∅	LR1, LR3	LR1, LR6, LR5	LR2

2.3.2 LiveOut Calculation

	START	L1	L2	L3
init	∅	∅	∅	∅
1	LR1, LR3	LR1, LR2	LR1, LR3, LR2	∅
2	LR1, LR3, LR2	LR1, LR2, LR3	LR1, LR3, LR2	∅
3	LR1, LR3, LR2	LR1, LR2, LR3	LR1, LR3, LR2	∅

2.4 Interference Graph