Jonathan Lam

Circuit Analysis Exam 1

Professor: Brian Frost

Fall 2019

1 Theoretical Questions (3 points each)

1. What can be said about two voltage sources in parallel with one another?
2. What can be said of two current sources in series with one another?
3. Draw a circuit which mesh analysis would fail to solve.
4. The passive sign convention, which we use in class, determines the sign of power across acircuit element. The active sign convention makes the exact opposite choice (all power values under the active sign convention are negatives of those under the passive sign convention). Describe, in words, the active sign convention without mentioning the passive sign convention.
5. With respect to the active sign convention, write an inequality which defines a passive element(including an integral). Is a resistor still passive?
6. Consider a resistor with resistance $R < 0$ in parallel with a voltage source of voltage $V > 0$. Does the resistor generate or dissipate power?
7. Consider a circuit with a complex resistive network, and a second circuit where this networkis replaced by one equivalent resistor. Which of the following are necessarily the same across the network and the single resistor in their respective circuits: Voltage, current, power?
8. Figure 1 shows a branch in a circuit. Write its branch constituent equation such that thebranch current is in terms of the branch voltage.
9. Write a brief argument using only Kirchoff's laws for the equivalent conductance formula for N conductors in parallel.
10. Write an argument for the equivalent resistance formula for N resistors in parallel, either by arguing from your above answer or by deriving it from Kirchoff's laws.
11. Spike failed to pay attention in circuits class, and is now at a loss when building his newestinvention. He has access to only a 5V battery, but wishes to apply 2.5V to a load resistor with resistance 250Ω. Spike can use two resistors to provide this voltage to the load, and he wants to make it as precise as possible. Draw the circuit to "divide" Spike's voltage, and write a small note about the size of resistors Spike should choose.

2 Computational Questions (33 points total)

12. (11 points) Consider the circuit in Figure 2. Using mesh analysis, write a matrix equation which will solve the circuit upon solution.

13. (6 points) Consider the circuit in Figure 2. Count the number of nodes, meshes and branches.

14. (11 points) Consider the circuit in Figure 3. Using modified nodal analysis, write a matrix equation which will solve the circuit upon solution.

15. (5 points) Consider the resistive network in Figure 4. Find the equivalent resistance of the network assuming all resistors have the same value R.

3 Expository Example (34 points total)

16. (2 points) This question will introduce you to the concept of voltage gain. In a circuit which is interpreted as having a voltage input and a voltage output, the voltage gain is defined as V_{OUT}/V_{IN}. For Spike's circuit in question 11, what is the desired voltage gain?

17. (4 points) With only resistors, what is the possible range of voltage gains one can achieve? Briefly explain.

18. (2 points) Theoretically, can one achieve voltage gains outside of this range using only passive elements? Hint: Does the integral definition of passive preclude this?.

19. (4 points) For the circuit in Figure 5, what is the voltage gain?

20. (2 point) There are elements other than resistors in this device – is it still limited to the range determined in question 17?

21. (1 point) The effective voltage gain of Spike's circuit is changed by the addition of a load. If you add a resistive load at the output of the circuit in Figure 5, does the gain change?

22. (3 points) This is related to a concept called output impedance. Considering the input to a circuit as an open circuit, one can compute the equivalent resistance of the network across the output terminals. In Spike's circuit, the output impedance is simply the resistance of the resistor across the output terminals. When a load is added, it is effectively placed in parallel with the output impedance of the circuit. For a load to have minimal effect on the gain, should the output impedance be small or large relative to the load?

23. (4 points) Inspired by your above answers, what do you think the output impedance of a dependent voltage source is? More specifically, what is the output impedance of the circuit in Figure 5.

24. (5 points) What is the voltage gain of the circuit in Figure 6?

25. (2 points) Is this circuit impacted by a load?

26. (2 points) What is this circuit's output impedance?

27. (1 points) What does the sign of gain represent?

28. (2 points) In reality, do you think we can build circuits with gains that are not impacted by loads? Explain.

Figures

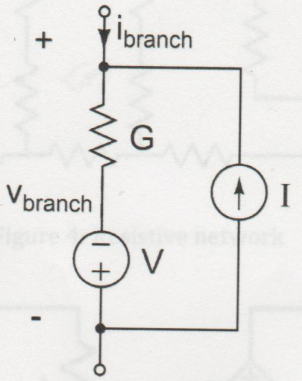


Figure 1: A given branch

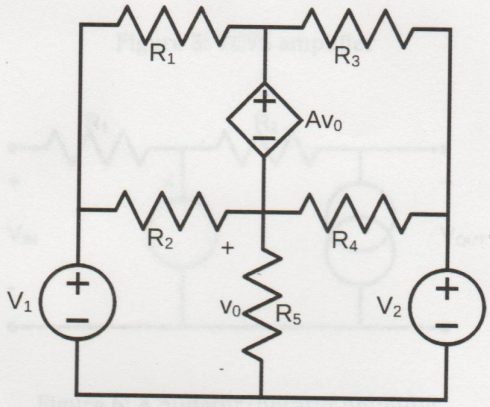


Figure 2: Using Mesh Analyisis

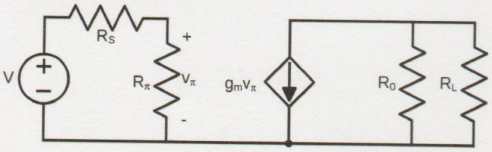


Figure 3: Using MNA

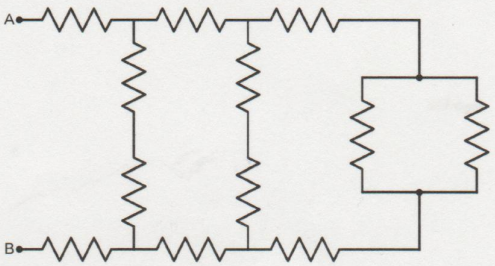


Figure 4: Resistive network

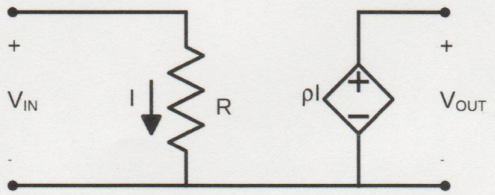


Figure 5: VCVS amplifier

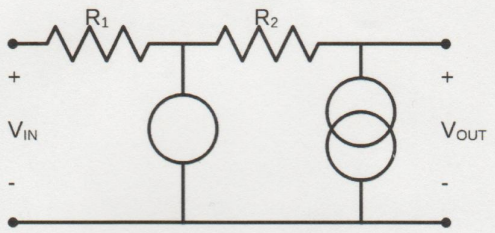
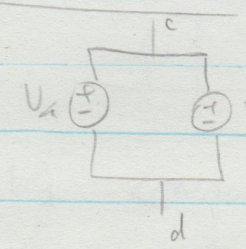


Figure 6: A nullator/norator amplifier

Theoretical Q's -

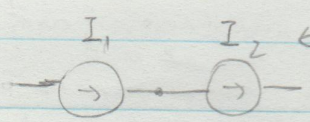
✓ (1)



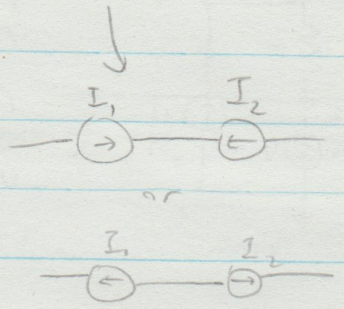
This can only exist if $V_a = V_b$, since the BCEs are:

$V_b = V_{cd} = V_a$.

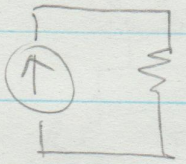
✓ (2)



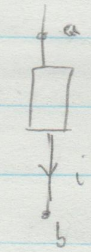
Since current is equal for 2 circuit elements in series, only possible if $I_1 = I_2$ (if facing same direction) or $I_1 = -I_2$ (if facing opposite directions)



✓ (3)



✓ (4)



If, in the diagram on the left, we specify $P = iV_{ab}$ as negative then the active sign convention is in use. i.e., this convention means a drop in voltage (and therefore power dissipated) is negative power.

perfect. first great answer I've seen for this problem

cannot generate power, so $P \neq 0$. Thus

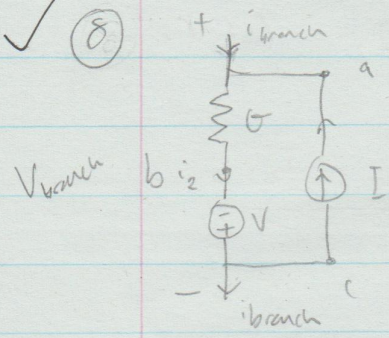
✓ (5) $\int_0^T P\,dt \leq 0$ (i.e., reverse sign from passive sign convention)

Yes, a resistor is still passive

✓ (6) It generates power (and is thus active)

✓ (7) Between the start and end node of the more complex resistive circuit, the power, voltage, and current would all be equal to the single resistor

✓ (8)



node a: $i_{branch} + I = i_2$

for conductor: $i_2 = (V_a - V_b)\,G$

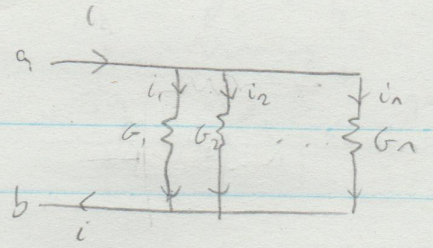
voltage source: $V = V_c - V_b$

branch voltage: $V_{branch} = V_a - V_c$

Solve for i_{branch} in terms of V_{branch}

$$i_{branch} = i_2 - I$$
$$= (V_a - V_b)\,G - I$$
$$= ((V_{branch} + V_c) - V_b)\,G - I$$
$$= (V_{branch} + (V_c - V_b))\,G - I$$
$$i_{branch} = (V_{branch} + V)\,G - I$$

✓ (9)



by KCL at node a, $i = i_1 + i_2 + \ldots + i_n$

(same at node b)

since $i_1 = V_{ab} G_1$ (by BLE of conductor),

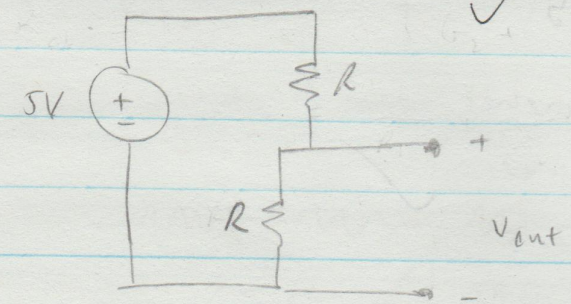
$i = V_{ab} G_1 + V_{ab} G_2 + \ldots + V_{ab} G_n$

divide both sides by V_{ab}

$\frac{i}{V_{ab}} = G_1 + G_2 + \ldots + G_n = G_{eq}$ (since $G = \frac{i}{V}$)

✓ (10) $R_{eq} = \frac{1}{G_{eq}} = \frac{1}{G_1 + G_2 + \ldots + G_n} = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2} + \ldots + \frac{1}{R_n}}$

✓



(resistor values are equal)

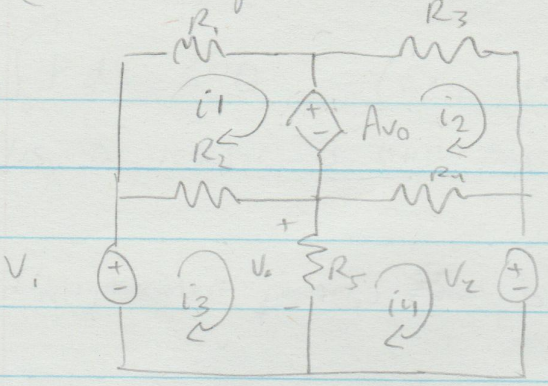
If spike connects his circuit to the +/− pins of V_{out}, then he can power his circuit with roughly 2.5 V.

✓ Note that to be more accurate means choosing smaller Rs (i.e., R should be much smaller than resistive load), but choosing too small an R could mean a very high current.

PERFECT
theoretical

(mesh analysis)

(12)



$V_o = (i_3 - i_4) R_5$ ✓

mesh 1: $R_1 i_1 + A(i_3 - i_4)R_5 + (i_1 - i_3)R_2 = 0$ ✓

" 2: $R_3 i_2 + (i_2 - i_4)R_4 - A(i_3 - i_4)R_5$ ✓

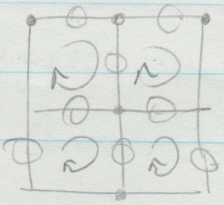
" 3: $(i_3 - i_1)R_2 + (i_3 - i_4)R_5 - V_1 = 0$ ✓

" 4: $(i_4 - i_2)R_4 + V_2 + (i_4 - i_3)R_5 = 0$. ✓

$$\begin{pmatrix} R_1+R_2 & 0 & AR_5 - R_2 & -AR_5 \\ 0 & R_3+R_4 & -AR_5 & -R_4 + AR_5 \\ -R_2 & 0 & R_2+R_5 & -R_5 \\ 0 & -R_4 & -R_5 & R_4+R_5 \end{pmatrix} \begin{pmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ V_1 \\ -V_2 \end{pmatrix}$$

-0.5

(13)



8 branches

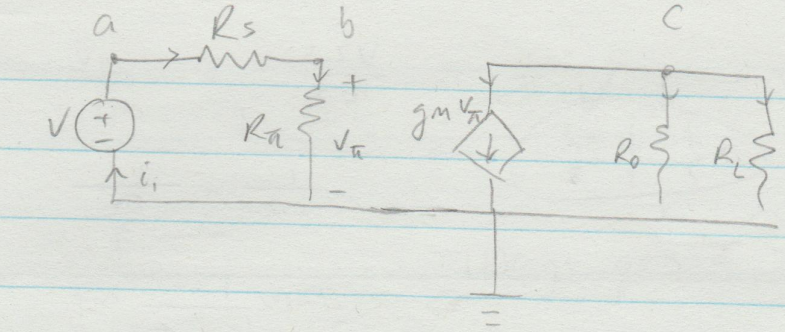
4 meshes ✓

5 nodes

-0.5

[-.5]

(14)



$V_\pi = V_b - V_{gnd} = V_b$ ✓

a: $i_1 = (V_a - V_b)\, G_s$ ✓

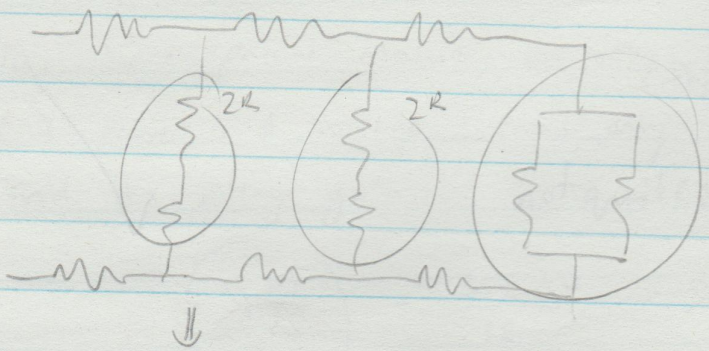
b: $(V_a - V_b)\, G_s = V_b\, G_\pi$ ✓

c: $g_m (V_b) + V_c G_o + V_c G_L = 0$ ✓

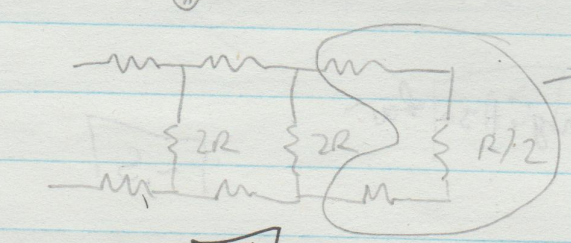
$V = V_a$ ✓

$$\left(\begin{array}{c|c|c|c} -G_s & G_s & 0 & 1 \\ G_s & -G_s - G_\pi & 0 & 0 \\ 0 & g_m & G_o + G_L & 0 \\ 1 & 0 & 0 & 0 \end{array}\right) \left(\begin{array}{c} V_a \\ V_b \\ V_c \\ i_1 \end{array}\right) = \left(\begin{array}{c} 0 \\ 0 \\ 0 \\ V \end{array}\right)$$ ✓

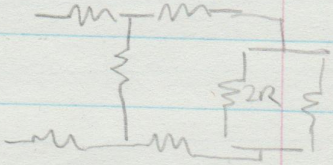
(15)



$\frac{R}{2}$ ✓

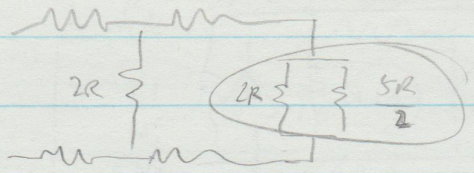


$\frac{5R}{2}$ ✓



$\frac{5R}{2}$ ✓

[-.5]



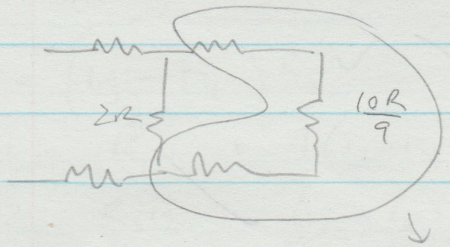
$$\rightarrow \frac{1}{\frac{1}{2R} + \frac{1}{\frac{5R}{2}}}$$

$$= \frac{1}{\frac{1}{2R} + \frac{2}{5R}}$$

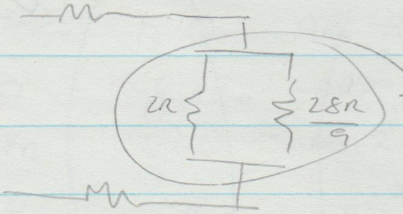
$$= \frac{1}{\frac{5}{10R} + \frac{4}{10R}}$$

$$= \frac{1}{\frac{9}{10R}}$$

$$= \frac{10R}{9}$$

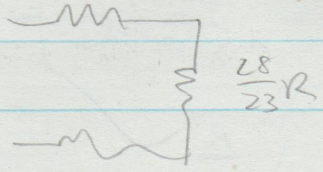


$\frac{28R}{9}$



$$\rightarrow R_{eq} = \frac{(2R)\left(\frac{28R}{9}\right)}{2R + \frac{28R}{9}}$$

$$= \frac{\frac{56R}{9}}{\frac{46R}{9}} = \frac{56}{46} = \frac{28}{23}$$



$$\frac{2(23) + 28}{23} R = \frac{46 + 28}{23} = \frac{74}{23} R$$

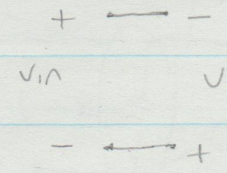
✓

Killing it so far

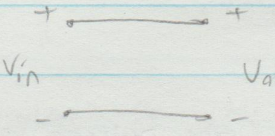
-.5

(16.) $\frac{V_{out}}{V_{in}} = \frac{2.5V}{5V} = \frac{1}{2}$ ✓

✓ (17)



Voltage gain $= \frac{-V_{in}}{V_{in}} = -1$.



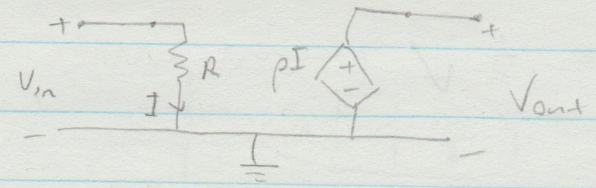
Voltage gain $= \frac{V_{in}}{V_{in}} = 1$.

It is not possible to create energy (and thus a larger voltage potential) using resistors (they only drop voltage) so largest voltage gain is 1 (where output is equal to input voltage). Conversely, since this is the largest possible magnitude for a voltage gain, flipping Vout gives the largest negative voltage gain of -1. Thus the range of voltage gains is [-1, 1]. Any value in between can be achieved with a voltage divider like in question 11, and a voltage gain of 0 can be created by connecting V_{out}^{+} to V_{out}^{-} with a S.C.

-1 ✓✓ (18) No, since passive elements can only drop (the magnitude of) power and voltage, thus only decreasing V_{in} and the voltage gain. not quite! v↑ i↓?

[-1.5]

(19)



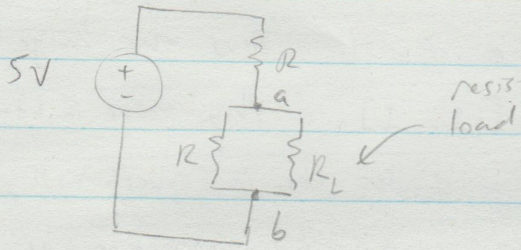
$$I = \frac{V_{in}}{R}, \quad \rho I = \rho\left(\frac{V_{in}}{R}\right) = V_{out}$$

✓ $\Rightarrow \frac{V_{out}}{V_{in}} = \frac{\rho}{R}$ is the voltage gain

✓ (20) NO, since there is an active element (the voltage source, which is able to supply voltage and thus increase V_{out} by an indeterminate amount, allowing voltage gain to exceed $[-1, 1]$.

-.5 (21) Yes.

wrong circuit!
Right answer for this one though



$$R_{eq} = R + \frac{RR_L}{R+R_L}$$

$$i = \frac{V}{R_{eq}} = \frac{V}{R + \frac{RR_L}{R+R_L}}$$

$$V_{ab} = iR_{RR_L} = \left(\frac{V}{R+\frac{RR_L}{R+R_L}}\right)\left(\frac{RR_L}{R+R_L}\right) = \frac{V(RR_L)}{R(R+R_L)+RR_L}$$

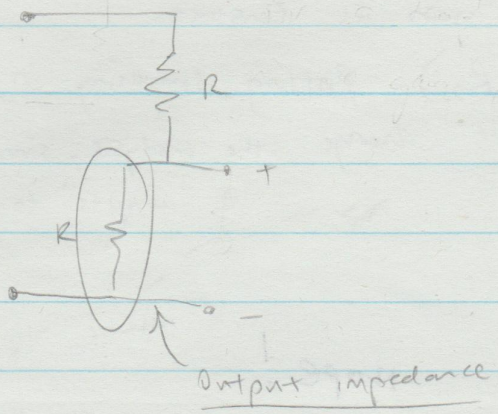
[-2]

$$= V\left(\frac{RR_L}{R^2+2RR_L}\right) < V\left(\frac{1}{2}\right), \text{ since}$$

$RR_L < \frac{1}{2}(R^2+2RR_L)$. Therefore V_{out} is lower than $\frac{1}{2}$, so the gain drops.

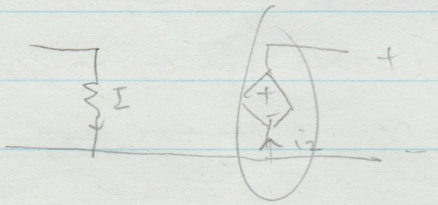
[2]

✓ (22)



Similar to HW #2, if the output impedance (R) is small, then the voltage gain remains close to the desired value. Therefore the output impedance should be small relative to the load.

-1 (23)



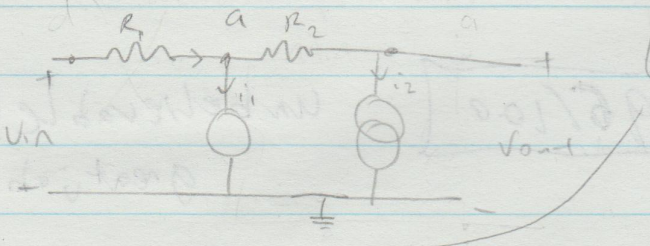
equivalent resistance across a dependent voltage source is

$R = \frac{V}{i_2} = \frac{pI}{i_2}$ by ohm's Law i_2 arbitrary. DP is zero

"output impedance" here

✓ (24)

yes!



$(V_{in} - V_a) G_1 = (V_a - V_{out}) G_2$

$-V_a = 0$

$(V_a - V_{out}) G_2 = i_2$

$\frac{G_1}{G_2}(V_{in} - V_a) = (V_a - V_{out}) \Rightarrow V_{out} = 0 - \frac{G_1}{G_2}(V_{in} - 0) = \frac{-G_1}{G_2}$

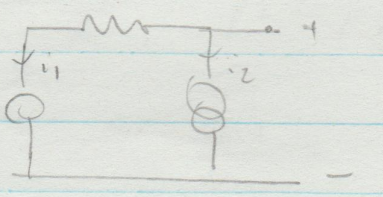
[3]

$\Rightarrow$ voltage gain $= \frac{\frac{-G_1}{G_2} V_{in}}{V_{in}} = \frac{-G_1}{G_2} = \frac{-R_2}{R_1}$

-3

✓ (25) No, since its voltage gain is constant (dependent only on R_1, R_2). Putting something in parallel to V_{out} wouldn't change the analysis, since the narrator "adapts" to the new resistance

✓1 (26)



nope!

$i_1 = 0$, $i_2 = -V_{out} G_1$

$R = \frac{V}{i} = \frac{V_{out}}{-V_{out} G_1} = -\frac{1}{G_1}$

↑ by ohm's Law

O again

✓ (27) Sign of gain represents whether output +/− is oriented same way as input +/−, i.e., if V_{in} and V_{out} have same sign, then sign of gain is positive; if 0, then $V_{out} = 0$; if negative, then V_{out} has opposite sign of V_{in}

✓ (28) I don't think so. A load always changes the constraints of a real system (e.g. equivalent resistance) and thus should change the circuit's parameters.

-4

raw score: 96/100

unbelievable!
great job

Jonathan Lam

Circuit Analysis Exam 2

Professor: Brian Frost

Fall 2019

1 Theoretical Questions (3 points each)

1. Write a brief argument using only Kirchoff's laws for the equivalent capacitance formula for N capacitors in parallel.
2. Write a brief argument using only Kirchoff's laws for the equivalent inductance formula for N inductors in parallel.
3. Suppose I have a complex linear two-terminal circuit with open circuit output voltage V. I load it with a resistor with resistance R, and measure a voltage V_0 across the resistor. What is the Thevenin resistance?
4. Consider a source-free RC circuit with $v(0) = V_0$. Derive the voltage as a function of time for $t > 0$.
5. As C grows large, what device does a capacitor behave like over short time intervals? Provide a brief explanation.
6. Consider the circuit in Figure 1. The voltage source has two clear components – a DC and AC component. Draw the two circuits corresponding to a superposition-based analysis of this circuit using a DC/AC breakdown of the voltage source.
7. Solve for the voltage across the resistor in the DC component circuit from the previous question. What role does the DC component play in the solution to the entire system?
8. Suppose a circuit contains an inductor which is known to have a periodic voltage across it. What will happen (over time) to the inductor current if the average voltage over a period is greater than 0? Equal to 0? Less than 0?
9. Describe the principal of maximum power transfer for two-terminal linear circuits – what is the maximum power transfered, and to what object is it transfered?
10. Provide one "pro" and one "con" about each of overdamped, underdamped and critically damped responses. That is, give me one reason each response is desirable and one reason each is not.
11. Consider an LC tank with initial current 0 and initial voltage V_0. Derive the frequency at which the device resonates. Will the system, theoretically, resonate forever? Realistically?

2 Computational Questions (33 points total)

12. (14 points) Consider the circuit in Figure 2. Find a differential equation which determines the current through inductor L_2, and come up with parameter regions in which the circuit is underdamped, critically damped and overdamped.

13. (14 points) Consider the circuit in Figure 3. The terminals across which V_{out} is defined are, naturally, the output terminals of this circuit. Using the matrix methods discussed in class, provide two matrix equations which will allow me to find V_{TH} and R_{TH}. Show me not just these equations, but how, once solved, I can find these Thevenin equivalent values.

14. (5 points) Consider the circuit in Figure 3 again. Draw the circuits corresponding to a superposition analyisis of this circuit and explain how I would use superposition to solve for the voltage across R_3 if the circuit is not loaded. Just explain – no need to write any matrix equations.

3 Expository Example (34 points total)

15. (3 points) In this problem we will discuss the buck converter – a DC-to-DC converter circuit. Imagine a circuit such as that shown in Figure 4, where the switch is open or closed dependent on a periodic function of time $f(t)$. Assume the period of the function is T. For $t \in (0, T_{ON})$, the switch is closed, and for $t \in (T_{ON}, T)$ the switch is open. Draw the current through the resistor as a function of time over two periods and label the maximum current with a value.

16. (3 points) The average power absorbed by a circuit element is defined by

$$P = \frac{1}{T}\int_0^T p(t)\,dt,$$

where $p(t)$ is the instantaneous power absorbed by the element. For the scenario in the previous question, find the average power consumed by the resistor. How does it differ from the average power absorbed by a resistor if the switch had been closed for all time? Write your answer in terms of the *duty cycle* $D = T_{ON}/T$.

17. (4 points) Now consider the significantly more complex circuit in Figure 5. This circuit is called a *buck converter*. The switch S1 will operate just as the switch in the previous problems, and switch S2 will behave in exactly the opposite way – that is, switch S2 is open for $t \in (0, T_{ON})$ and closed for $t \in (T_{ON}, T)$, and also periodic with period T. Look at this circuit! Let v_L be the voltage acros the inductor, and v_C be the voltage across the capacitor. Draw at least two periods of $v_C + v_L$. (hint: this requires no math)

18. (8 points) Consider now only the period from 0 to T. Draw the two circuits corresponding to the "on" time period and the "off" time period for switch S1. Each of these circuits are second order, but given that the time periods over which they are defined are finite, they will never reach their steady state conditions. Still, what are their steady state conditions? How do their initial and final (meaning at $t = 0$, T_{ON} and T) conditions relate?

19. (8 points) The intended operation of a buck converter is that the capacitor voltage v_C is approximately constant at a voltage V_0. Assume that is in fact the case. Given this fact, find and draw the voltage across the inductor in time for one period. Given some initial current I_0, find and sketch the current in time as well over one period.

20. (2 points) From your math and your drawing it should be easy to tell the amount that I changes during the "on" time period and "off" time period – call these values Δi_{ON} and Δi_{OFF}. If I tell you the inductor current is *also* periodic with period T (which it must be), what is the relationship between Δi_{ON} and Δi_{OFF}?

21. (3 points) Using this relationship, tell me what the relationship between V_0 and V is in terms of D and circuit component values only.

22. (3 points) Amazingly, this is independent of some parameters. This is principally due to our assumption that the output voltage is approximately constant. What does this assumption correspond to in terms of circuit component values?

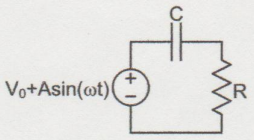


Figure 1: A sourced RC circuit.

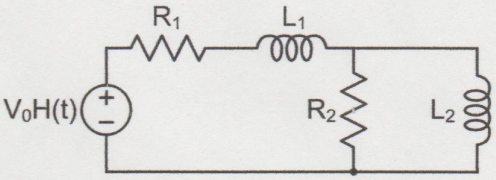


Figure 2: A second order circuit.

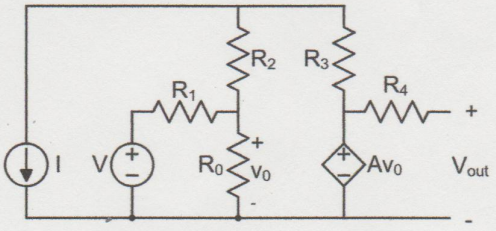


Figure 3: Linear two-terminal circuit.

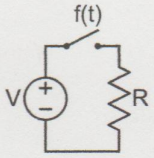


Figure 4: Resistive network with timed switching.

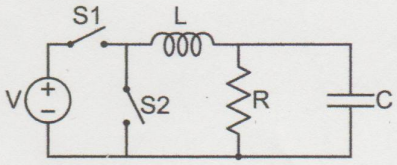
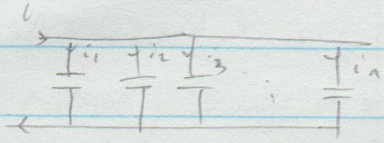


Figure 5: The buck converter.

(1)



all sure voltage b/t same nodes

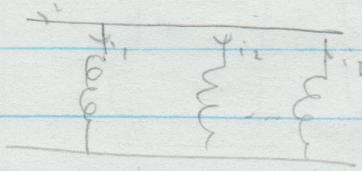
$i = i_1 + i_2 + \dots + i_n$

$= C\frac{dV}{dt} + C_2\frac{dV}{dt} + \dots + C_n\frac{dV}{dt}$

$= (C_1 + C_2 + \dots + C_n)\frac{dV}{dt}$ ✓

$= (C_{eq})\frac{dV}{dt} \Rightarrow C_{eq} = C_1 + C_2 + \dots + C_n$

(2)



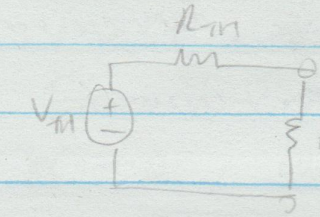
$i = i_1 + i_2 + \dots + i_n$

$= \frac{1}{L_1}\int V\,dt + \dots + \frac{1}{L_n}\int V\,dt$

$= \left(\frac{1}{L_1} + \frac{1}{L_2} + \dots + \frac{1}{L_n}\right)\int V\,dt$

$= \left(\frac{1}{L_{eq}}\right)\int V\,dt \Rightarrow L_{eq} = \frac{1}{\frac{1}{L_1} + \frac{1}{L_2} + \dots + \frac{1}{L_n}}$

(3)



$V_{TH} = V$

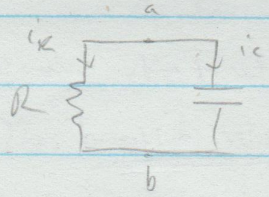
$V_{ab} = V_0$

voltage divider: $V\frac{R}{R+R_{TH}} = V_0$

$\frac{V_0}{V} = \frac{R}{R+R_{TH}} \Rightarrow (R+R_{TH}) = \frac{RV}{V_0}$

$\Rightarrow R_{TH} = R\left(\frac{V}{V_0} - 1\right)$ ✓

(4)



$i_R + i_C = 0$ (KCL)

$\frac{V_{ab}}{R} + C\int_0^t \frac{dV}{dt} = 0$

$\frac{dV}{dt} + \frac{1}{RC}V = 0 \Rightarrow \frac{dV}{dt} = -\frac{1}{RC}V \Rightarrow \frac{dV}{V} = -\frac{1}{RC}dt$

$\Rightarrow \int_0^t \frac{dV}{V} = \int_0^t -\frac{1}{RC}dt \Rightarrow \ln V = -\frac{t}{RC} + (-) \quad V = Ce^{-\frac{t}{RC}}$

$V(0) = Ce^0 = C \Rightarrow C = V_0$ ✓

$\Rightarrow V(t) = V_0 e^{-\frac{t}{RC}}$, for $t > 0$

-1

-1 (5) as C grows, its time constant grows, means slower recharge/charge times. It acts more like an open circuit over short time intervals b/c so little current flowing through it (in other words, since $V(t) = \frac{1}{C}\int i\,dt$, i small or C large means smaller changes in V for same time interval)

~ only kinda

mad like battery

(6) V_0 (DC) R $A\sin(\omega t)$ (AC) R ✓

✓ (7) voltage across resistor from DC component is 0, b/c open circuit from capacitor. This increases the voltage difference across the capacitor, but doesn't contribute to the current

(8) $i = \frac{1}{L}\int v\,dt \Rightarrow i_{avg} = \frac{1}{L}\int v_{avg}\,dt$.

-0.5 inductor current will increase if avg voltage > 0 ✓

avg " " stay the same " " " = 0

" " " decrease " " " < 0 ✓

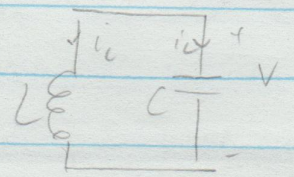
✓ (9) R_{TH}, a, V_{TH}, R_L, b

In a linear two-terminal circuit, the power transferred over a load (i.e., power dissipated over R_L) is maximum $\frac{V_{TH}^2}{4R_{TH}}$, and it is transferred if $R_{TH} = R_L$

-2.5

	Con	Pro
(10) Overdamped:	slower than C.D. to get to S.S.	practical, reaches S.S. and easier to produce than C.D. ✓
Critically damped:	hard to achieve	reaches S.S. fastest
Underdamped:	oscillatory behavior undesirable, doesn't die out quickly	Can provide useful / interesting circuits like LC tank w/ oscillations

(11)



$i_L(0) = 0, \quad V(0) = V_0$

KCL: $i_L + i_C = 0$

$$\frac{1}{L}\int V\,dt + C\frac{dV}{dt} = 0$$

$$\frac{1}{L}V + C\frac{d^2V}{dt^2} = 0$$

$$\frac{d^2V}{dt^2} + \frac{1}{LC}V = 0$$

$\alpha = 0, \quad \omega_0 = \frac{1}{\sqrt{LC}} \Rightarrow$ underdamped,

$\omega_d = \sqrt{\omega_0^2 - \alpha^2} = \omega_0 = \frac{1}{\sqrt{LC}}$, which is the resonating frequency

Theoretically, the complete response is in the form,

$$V(t) = V_0 + e^{-\alpha t}(A_1\cos(\omega_d t) + A_2\sin(\omega_d t)),$$

and since $\alpha = 0$,

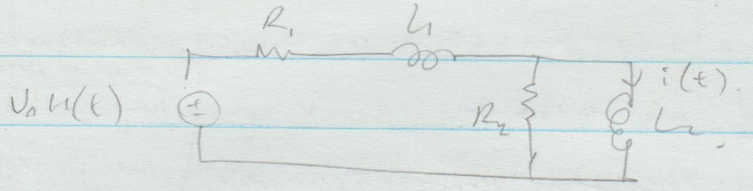
$$V(t) = V_0 + A_1\cos(\omega_d t) + A_2\sin(\omega_d t)$$

This should theoretically oscillate forever; however, there is always some resistance in a non-ideal LC tank (or any circuit) that leads to damping (not resonating forever). ✓

-2.5

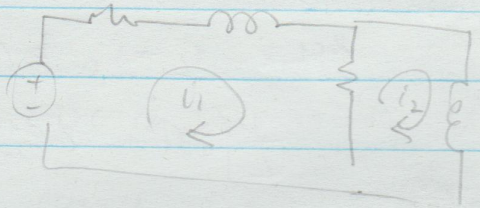
theor: $\frac{30.5}{33}$ good!

(12)



Assuming S.S reached before $t=0$, $i(t) = v(t)$ must be zero, b/c no sources and any current would die out (be dampened) through the resistors

after $t=0$:



i_2 is current through inductor L_2

KVL: $-V_0 + i_1R_1 + L_1\frac{di_1}{dt} + (i_1 - i_2)R_2 = 0$

KVL: $(i_2 - i_1)R_2 + L_2\frac{di_2}{dt} = 0$

$\Rightarrow i_2R_2 + L_2\frac{di_2}{dt} = i_1R_2$

$\Rightarrow i_1 = i_2 + \frac{L_2}{R_2}\left(\frac{di_2}{dt}\right)$ ← plug back into first KVL

$$V_0 = \left(i_2 + \frac{L_2}{R_2}\left(\frac{di_2}{dt}\right)\right)R_1 + L_1\frac{d}{dt}\left(i_2 + \frac{L_2}{R_2}\left(\frac{di_2}{dt}\right)\right) + \left(\left(i_2 + \frac{L_2}{R_2}\left(\frac{di_2}{dt}\right)\right) - i_2\right)R_2$$

$$V_0 = R_1i_2 + \left(\frac{L_2}{R_1R_2} + L_2\right)\left(\frac{di_2}{dt}\right) + L_1\left(\frac{di_2}{dt} + \frac{L_2}{R_2}\frac{d^2i_2}{dt^2}\right)$$

$$V_0 = i_2 + \left(L_1 + L_2 + R_1\frac{L_2}{R_2}\right)\frac{di_2}{dt} + \frac{L_1L_2}{R_2}\frac{d^2i_2}{dt^2}$$

consider transient response, ignore V_0 →

$$V_0 = \frac{d^2i_2}{dt^2} + \left(\frac{R_2}{L_2} + \frac{R_2}{L_1} + \frac{1}{R_1L_1}\right)\frac{di_2}{dt} + \left(\frac{R_2}{L_1L_2}\right)i_2$$

$$\alpha = \frac{1}{2\left(\frac{R_2}{L_2} + \frac{R_2}{L_1} + \frac{1}{R_1L_1}\right)}, \quad \omega_0 = \sqrt{\frac{L_1L_2}{R_2}}$$

(for quadratic characteristic eqn $x^2 + Ax + B = 0$, $\alpha = \frac{1}{2A}$, $\omega_0 = \frac{1}{\sqrt{B}}$) ✓

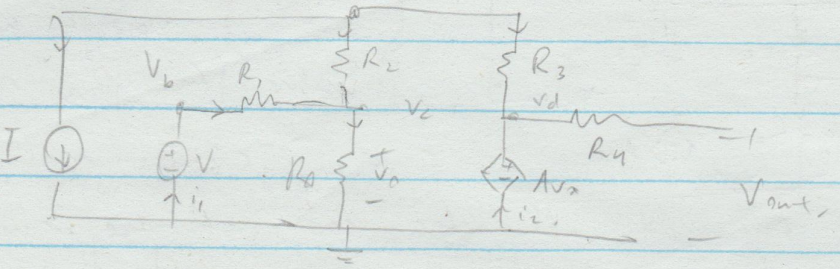
when $\alpha > \omega_0$, overdamped,

$\alpha = \omega_0$, critically damped,

$\alpha < \omega_0$, underdamped.

-2.5

(13)



V_{th}: find O.C. voltage V_{out}

MNA:

a: $I + (V_a - V_c)G_2 + (V_a - V_d)G_3 = 0$

b: $i_1 = (V_b - V_c)G_1$ ✓

c: $(V_b - V_c)G_1 + (V_a - V_c)G_2 = V_c G_0$ ✓

d: $(V_a - V_d)G_3 = -i_2$, way you drew it -.5

$V = V_b$ ✓

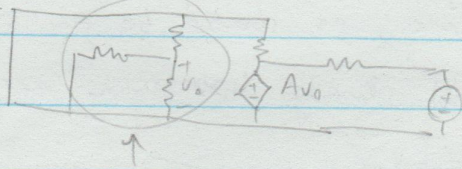
$AV_c = V_d$ ✓

$$\begin{pmatrix} G_2+G_3 & 0 & -G_2 & -G_3 & 0 & 0 \\ 0 & G_1 & -G_1 & 0 & -1 & 0 \\ G_2 & G_1 & -(G_0+G_1+G_2) & 0 & 0 & 0 \\ G_3 & 0 & 0 & -G_3 & 0 & -1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -A & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} V_a \\ V_b \\ V_c \\ V_d \\ i_1 \\ i_2 \end{pmatrix} = \begin{pmatrix} -I \\ 0 \\ 0 \\ 0 \\ V \\ 0 \end{pmatrix}$$

✓ $\boxed{V_d = V_{out} = V_{Th}}$ so solve for V_d.

To find R_{Th}: turn off indep. sources, use test voltage.

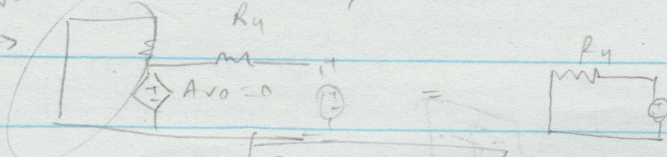
O.C.! →



-3

no current flowing through here, so this acts like an O.C., and $V_0 = 0$, so $AV_0 = 0$.

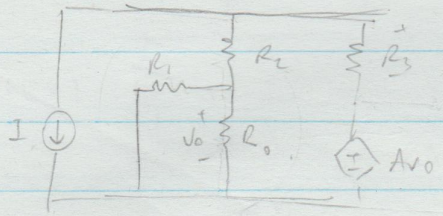
No current flowing through here →



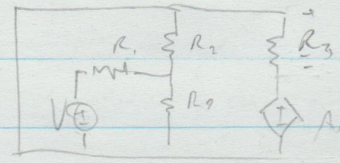
-6

$\boxed{R_{eq} = R_4}$ (don't need matrix equations)

14.



B.C. →

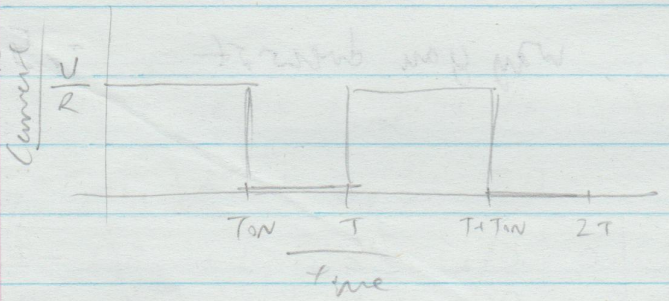


Voltage across

Solve for R_3 normally (using circuit analysis methods) for each of the circuits on the left, and add the two calculated voltages to get the voltage across R_3 in the original circuit (works b/c of linearity) -1

Comp: $\frac{28.5}{33}$

(15)



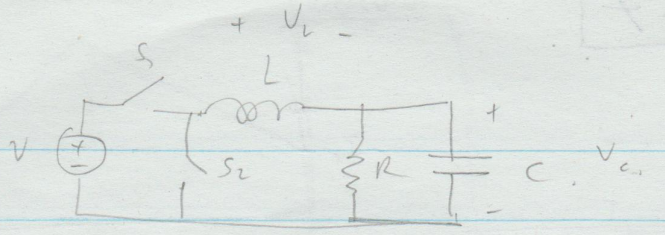
(16) $P_{avg} = \frac{1}{2T}\int_0^{2T} p(t)\,dt = \frac{1}{2T}\int_0^{2T} i^2(t)\,R\,dt.$

$= \frac{2}{2T}\int_0^{T_{ON}} \left(\frac{V}{R}\right)^2 R\,dt = \frac{2}{2T}\frac{V^2}{R}\int_0^{T_{ON}} dt = \frac{V^2 T_{ON}}{RT} = \frac{T_{ON}}{T}\left(\frac{V^2}{R}\right)$

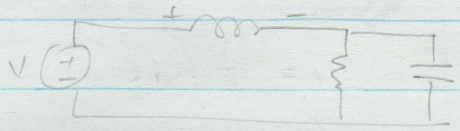
$= D\frac{V^2}{R}$

This is independent of time, so should be the same value throughout time (as long as starting and ending at the same part (phase?) of the period).

(17)

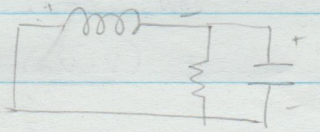


alternating between:



$V_L + V_c = V$ ✓

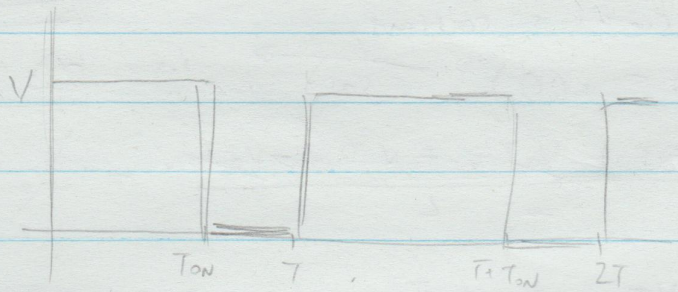
and:



parallel LRC,

$V_L = -V_c$,

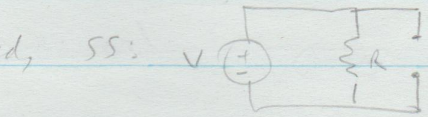
So $V_L + V_c = 0$. ✓



✓

(18) Sketches.

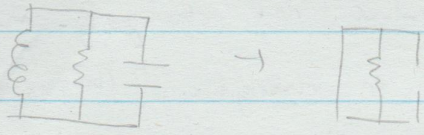
when S_1 closed, SS:



✓ $V_L = 0$, $V_c = V$

$i_L = \frac{V}{R}$, $i_c = 0$.

when S_2 closed SS:



(nothing flowing, no voltage) ✓

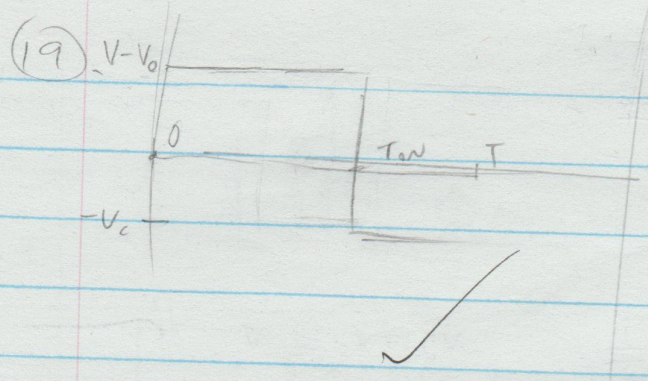
$V_L = 0$ $V_c = 0$

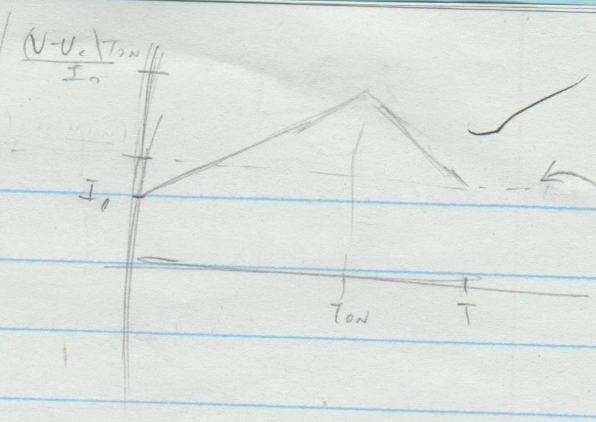
$i_L = 0$ $i_c = 0$.

✓

At each "switching point" ($t = 0$, T_{ON}, T, $T + T_{ON}$, $2T$, ...), the conditions of the two circuits must match, since the circuit switches instantaneously between these two circuits, and with both of these, the inductor and capacitor prevent an instantaneous change in current and voltage conditions (so conditions must be same to ensure continuity)

[-7]

(19) 



$i = \frac{1}{L}\int v\,dt$

(integrate one on the left).

at T_{ON}, $i = \frac{1}{L}\int_0^{T_{ON}} (V - V_0)\,dt + I_0$

$= \frac{(V-V_0)T_{ON}}{L} + I_0$

at T, $i = \frac{1}{L}\int_{T_{ON}}^{T} -V_0\,dt + I_{T_{ON}}$

$= -\frac{V_0(T-T_{ON})}{L} + \frac{(V-V_0)T_{ON}}{L} + I_0$

(20) if inductor current is constant,

(21) then $i(T) = i(0)$, so: i.e., $\Delta i_{ON} = \Delta i_{OFF}$

$I_0 + \frac{-V_0 T + V_0 T_{ON} + V T_{ON} - V_0 T_{ON}}{L} = I_0$

$\Rightarrow \quad V T_{ON} = V_0 T$

$\frac{T_{ON}}{T} = \frac{V_0}{V}$

so $\frac{V_0}{V} = D$. (no other circuit components necessary)

(22) Capacitor value not changing much means that it takes a long time to charge. This can be a combination of L, R, and/or C being large. (i.e., large time constants)

exp: 34/34 Wow!

total: 93/100

would've been nearly perfect if you realized ① → !

Basic DC Circuit Analysis Lab

Henry Son and Jonathan Lam

Professor Lisa Shay

ECE 291 Sophomore Projects

September 17, 2019

Table of Contents

I. Introduction

The objective of this lab is to become familiar with the process of analyzing simple DC circuits. It is an introductory lesson to the theoretical and practical sides of using two basic circuit elements, resistors and power sources, and two pieces of equipment, power supplies and digital multimeters (DMMs). The lab will begin with a review of resistor color bands, and then requires the use of Ohm's Law, the voltage divider rule, and the current divider rule to solve for V, I, and R in basic circuits. An understanding of KVL and KCL are also reinforced with some theoretical questions. The practical side is to build these circuits, to measure electrical aspects of the circuits, and to calculate other measures (and the percent errors) to determine how closely the theoretical laws hold true in practice.

II. Theory and Prelab

The following calculations were done in preparation for the lab.

Note: For the calculations in this theory section, a bar $\bar{x}$ indicates the last significant digit if more digits are shown than are significant. Only final results are rounded to the correct number of significant digits; intermediate steps may be shown rounded for concision.

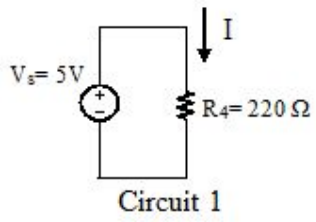
Resistor review

A review of resistor color bands and interpretations is shown in the below table.

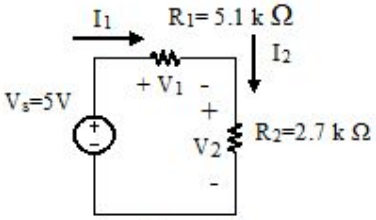
Table 1. Resistor colors and values

Color of band 1	Color of band 2	Color of band 3	Color of band 4	Resistor value
blue	red	red	silver	6200 ± 620 Ω
red	violet	brown	gold	270 ± 14 Ω
yellow	orange	black	gold	43 ± 22 Ω
brown	gray	green	gold	1800 ± 90k Ω
red	violet	orange	gold	27000 ± 1350 Ω
white	brown	brown	gold	910 ± 46 Ω
red	red	orange	gold	22 ± 1.1k Ω

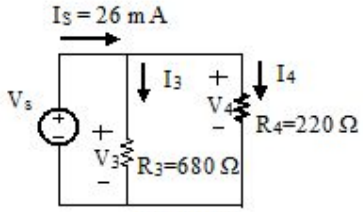
Circuit diagrams



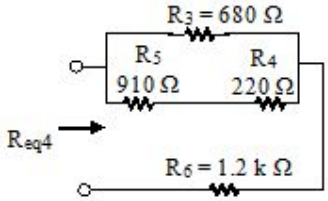
Circuit 1



Circuit 2



Circuit 3



Circuit 4

Circuit value calculations

These values were calculated to estimate results for the lab. They appear in the “pre-lab” section of the tables in Section IV: Results and Discussion, and use the desired resistor values from the circuit diagrams (which may differ slightly from the actual resistor values).

1. Ohm’s law

Solve for I in Circuit 1:

$I = \frac{V}{R} = \frac{5V}{220\Omega} = \bar{2}3mA$

2. KVL

Determine V_1 and V_2 for circuit 2, using the voltage divider rule.

$V_1 = 5V \times \frac{5.1k\Omega}{5.1k\Omega+2.7k\Omega} = \bar{3}.27V$

$V_2 = 5V \times \frac{2.7k\Omega}{5.1k\Omega+2.7k\Omega} = \bar{1}.73V$

Write a KVL equation for the circuit.

$-V_s + V_1 + V_2 = 0V$

$-5V + 3.27V + 1.73V = 0.V$

Calculate the currents I_1 and I_2 using Ohm’s Law and the voltages just computed

$I_1 = \frac{3.27V}{5.1k\Omega} = 0.\bar{6}4mA$
$I_2 = \frac{1.73V}{2.7k\Omega} = 0.\bar{6}4mA$
Their currents are the same, which makes sense because they form part of the same loop (in series).

3. KCL

Determine I_3 and I_4 for Circuit 3, using the current divider rule.
$I_3 = 26mA \times \frac{1/680\Omega}{1/680\Omega+1/220\Omega} = 6.\bar{3}6mA$
$I_4 = 26mA \times \frac{1/220\Omega}{1/680\Omega+1/220\Omega} = 19.\bar{6}mA$

Calculate the voltages V_3 and V_4 using Ohm's law and the currents just computed.
$V_3 = IR = 6.\bar{3}6mA \times 680\Omega = 4.3V$
$V_4 = IR = 19.\bar{6}mA \times 220\Omega = 4.3V$
The voltages are the same because they connect the same two nodes (i.e., act in parallel)

Write a KCL equation for the circuit.
$-I_s + I_3 + I_2 = 0$
$-26mA + 6.\bar{3}6mA + 19.\bar{6}mA = 0.mA$

Determine equivalent resistance seen by the voltage source for circuit 3 and then determine the value of the voltage source V_s needed to provide a supply current $I_s = 26mA$.
$R_{eq3} = \frac{R_3R_4}{R_3+R_4} = \frac{680\Omega \times 220\Omega}{680\Omega+220\Omega} = 166\Omega$
$V_s = IR = 0.026A \times 166\Omega = 4.3V$

4. Series and parallel resistances

Determine the equivalent resistance, R_{eq4}, for circuit 4.
$R_{eq4} = 1.2\Omega + \frac{1}{1/680\Omega+1/(910\Omega+220\Omega)} = 162.4\Omega$

III. Test procedure, equipment, and data

The test setup involved a tabletop digital multimeter, an adjustable power supply, and carbon resistors. All of the recorded measured values were measured with the multimeter, and all digits provided by the multimeter display are recorded.

The desired and measured resistor values are shown in Table 2.

Table 2. Desired and actual resistor values

	Desired resistance	**Components used** (gold tolerances)	**Measured resistance**
R_1	5.1kΩ	5.1kΩ	5.0620kΩ
R_2	2.7kΩ	2.2kΩ + 510Ω	2.6653kΩ
R_3	680Ω	560Ω + 100Ω	654.41Ω
R_4	220Ω	220Ω	218.08Ω
R_5	910Ω	910Ω	902.58Ω
R_6	1.2kΩ	1.2kΩ	1.17837kΩ

The voltage source voltage is shown in Table 3.

Table 3. Desired and actual voltage source voltage

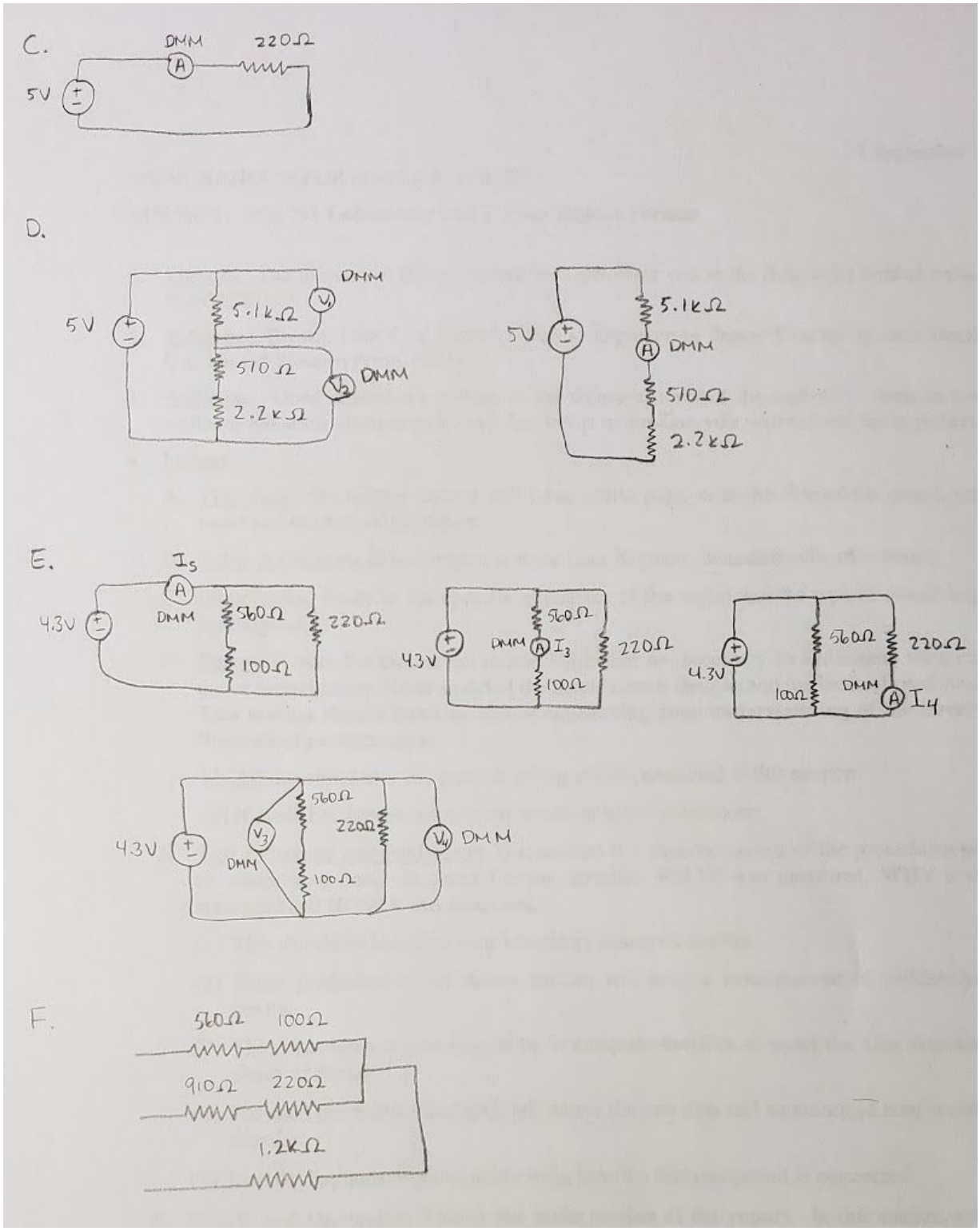
	Desired	**Measured**
$\mathbf{V_s}$	5V	4.96025V

The experimental setups for measuring current and voltage with the multimeter are shown in Figure 1. The measured current and voltage values are shown in Table 4.

Table 4. Measured voltages, currents, and resistances

		Measured value
Circuit 1	I	22.600mA
Circuit 2	V_1	3.2482V
	V_2	1.711V
	I_1	0.64128mA
	I_2	0.64125mA
Circuit 3	I_s	25.635mA
	I_3	6.4335mA
	I_4	19.2585mA
	V_3	4.2195V
	V_4	4.2193V
Circuit 4	R_{eq}	$1.59199k\Omega$

Figure 1. Equipment setup



Results and Discussion

Note: In the following sections, the percent error formula is $\%\ err = \frac{|experimental - theoretical|}{theoretical} \times 100\%$, and the calculation will be omitted.

Part A. Ohm's Law

The theoretical current and voltage are calculated with the following calculations using measured values.

$I = \frac{V_s}{R_4} = \frac{4.96025V}{218.08\Omega} = 22.745mA$

$V = I_{obs}R = 22.600mA \times 218.08\Omega = 4.9286V$

Table 5. Circuit 1 current and voltage values

Metric	Prelab	Theoretical	Experimental	Percent Error
I	$\bar{2}3mA$	22.745mA	22.600mA	0.638%
V	5V	4.9286V	4.96025V	0.638%

This result verifies Ohm's law, since it is used to calculate current given voltage and resistance.

Part B. KVL and voltage divider rule

The theoretical values for V_1, V_2 I_1, and I_2 are calculated for circuit 2 as follows using measured values:

$V_1 = 4.96025V \times \frac{5.0620k\Omega}{5.0620k\Omega + 2.6653k\Omega} = 3.2493V$ (voltage divider)

$V_2 = 4.96025V \times \frac{2.6653k\Omega}{5.0620k\Omega + 2.6653k\Omega} = 1.7109V$ (voltage divider)

$I_1 = \frac{3.2493V}{5.0320k\Omega} = 0.64573mA$

$I_2 = \frac{1.7109V}{2.6653k\Omega} = 0.64191mA$

Table 6. Circuit 2 current and voltage values

Metric	Prelab	Theoretical	Experimental	Percent Error
V_1	$\bar{3}.27V$	3.2493V	3.2482V	0.03385%
V_2	$\bar{1}.73V$	1.7109V	1.7110V	0.005849%
I_1	$0.\bar{6}4mA$	0.64573mA	0.64125mA	0.50021%
I_2	$0.\bar{6}4mA$	0.64191mA	0.64128mA	0.098145%

This result verifies the voltage divider rule, as it was used to calculate the voltage across two resistors in series. As for any circuit, it also demonstrates KCL, since the current across circuit elements in series are equal, and KVL, because the sum of the voltage drops of the resistors is equal to the voltage source potential.

Voltage must be measured *across* a circuit element because it is only defined as a relative energy potential, i.e., with reference to another node. In other words, voltage across a circuit element is only the difference in potential between the two ends of the circuit element.

Part C. KCL and current divider rule

Theoretical values for I_3, I_4, V_3, V_4, and i_s for Circuit 3 were calculated as follows using measured values:

$I_3 = 25.635mA \times \frac{1/654.41\Omega}{1/654.41\Omega+1/218.08\Omega} = 6.4075mA$ (current divider)

$I_4 = 25.635mA \times \frac{1/218.08\Omega}{1/654.41\Omega+1/218.08\Omega} = 19.227mA$ (current divider)

$V_1 = IR = 6.4075mA \times 654.41\Omega = 4.1931V$

$V_2 = IR = 19.227mA \times 218.08\Omega = 4.1930V$

$I_s = I_3 + I_4 = 6.4335mA + 19.2585mA = 25.6920$

Table 7. Circuit 3 current and voltage values

Metric	Prelab	Theoretical	Experimental	Percent Error
I_3	$6.\bar{3}6mA$	6.4075mA	6.4335mA	0.40577%
I_4	$1\bar{9}.6mA$	19.227mA	19.2585mA	0.1638%
V_1	4.3V	4.1931V	4.2195V	0.62961%
V_2	4.3V	4.1930V	4.2193V	0.69878%
I_s	26mA	25.6920mA	25.6350mA	0.22185%

This circuit analysis directly illustrates the use of the current divider rule, as it is used to calculate the current across multiple branches given the (inverse) resistances of each branch. It also demonstrates KCL, as the sum of the currents through the two branches is equal to the current entering the branches (I_s). As with Parts A and B, the percent errors are very small here, for likely the same reasons as before.

Here, as opposed to voltage, current must be measured by “breaking” the circuit because the current is calculated at a node, rather than as a difference between nodes. An ammeter can only measure the current at a node by having the node’s current pass through it, thus requiring “breaking” the circuit.

Part D. Equivalent Resistance

The theoretical equivalent resistance calculation is shown below using measured values, and a summary of the theoretical and measured values is shown in Table 7 below.

$$R_{eq4} = 1.17837k\Omega + \frac{1}{1/654.41\Omega+1/(902.58\Omega+218.08\Omega)} = 1591.52k\Omega$$

Table 8. Circuit 4 equivalent resistance

Metric	Prelab	Theoretical	Experimental	Percent Error
R_{eq}	1.624k Ω	$1.59152k\Omega$	$1.59199k\Omega$	0.029531%

The calculation uses the fact that the equivalent resistance of resistors in series is the sum of the resistances (a fact that can be derived from KCL), and the fact that the equivalent

resistance of resistors in parallel is the inverse sum of the inverses of the resistances (a fact that can be derived from KVL).

The power source must be removed when measuring resistance because the multimeter's method of testing resistance is to apply a voltage, measure the current, and calculate resistance using Ohm's law from these two known values. Applying an external voltage across the circuit adjusts this voltage, and thus the Ohm's law calculation is wrong.

Resistance is symmetric with probe polarity, since resistance is a physical property of a resistor and therefore independent of voltage or current passing through it. Voltage, however, is a directed difference between two potentials, so swapping probe polarities naturally reverses the sign of the difference. Similarly, current is a directed flow of electrons, so swapping probe polarities also reverses the sign of current.

Error analysis

The percent error of all of the theoretical calculations is very small. All of the percent errors were less than 0.6%, and most were within 0.01%, which all fall within the acceptable tolerance of within 3%.

The small error is likely to be due mostly to the measured resistance being incorrect, since the value was not extremely stable over time (a known physical property of resistors as they heat up), a modeling error with the assumption that the resistance of a resistor is constant. Another possibility in error is that voltage and resistance (or voltage and current) can be slightly nonlinear, a modeling error with Ohm's law. Again, both of these modeling errors are expected to be very small for practical applications, and the resulting error is therefore not large.

The error is less likely to be largely impacted by measurement (instrumental) error, since the instrumental precision is very high — from the measurements, the multimeter gave readings to the precision of ±100nA, ±10mΩ, and ±1000μV.

There should be no parametric error in the measured values from the lab, since the resistor and voltage source values were also measured (and therefore those measured values, and not standard values, were used in the calculations).

IV. Conclusions

Firstly, this lab reviews resistor band color-codes. This lab reinforces the fundamental relations between some basic measures of DC circuits: resistance, voltage, and current. Ohm's law creates a simple relation for a resistive load; for more complex circuits in series or parallel, the understanding of KVL (zero-sum voltage over a loop) and KCL (zero-sum current through a point) allows for the derivation and use of the voltage and current divider laws. Lastly, it is a good review on how to practically measure voltage and current in a circuit using a multimeter, and shows that the deviation of empirical values from from the theoretical values is very small (by the small percent errors), which means that the theory is a sound basis for prediction of resistance, voltage, and current in basic DC circuits.

V. Recommendations

This lab was useful in showing the basics of circuit analysis. While many of the sophomore electrical engineers had taken DLD, this is the first class in which the analysis method is paired with the practical side, so it does serve a useful purpose in the class.

A critique is that the lab report given to the students may be a little over-structured. It may be more educational if the students were told to solve for the same values in the circuits without explicitly mentioning what method to use (e.g., voltage and current divider laws). Or, perhaps it would be better to ask students to solve for a value multiple ways (e.g., using Ohm's law and the voltage divider law).

VI. References

Shay, Lisa A. "ECE 291 Laboratory and Project Report Format." *The Cooper Union*, 11 Sept. 2019.

Project Team Names: Jonathan Lam Victor Zhang (Zhihao)

Lab 2: Operational Amplifiers

I. Introduction.

In this lab you will build Operational Amplifier (OPAMP) circuits, characterize their transfer functions, and analyze their performance against ideal circuits. Complete Exercises 2-1 through 2-4 PRIOR to lab on November 7th. The completed lab is due at the start of lab on November 14th. There is no class on November 12th.

	$+V_{CC}$	$-V_{CC}$	R_9	R_{10}	R_{11}	P_3	C_8
Nominal	15V	-15V	1 MΩ	1 MΩ	1 kΩ	100 kΩ	1μF

II. Theory.

Op Amp Integrated Circuit. In this portion of the prelab, you will be analyzing the voltage follower and non-inverting amplifier. The OPAMP comes in several package styles. The version that we will use is the LF411 Op Amp Dual In-Line Package (DIP). A diagram of the chip with its pin assignments, along with a schematic symbol, is shown in Figure 1.

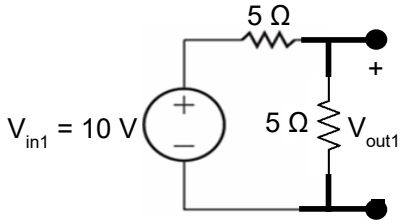
Exercise 2-1 Use the diagram to number the pins on the schematic symbol.

Figure 1. LF4111 Op Amp Dual In-Line Package Pin Connections and Schematic Symbol.

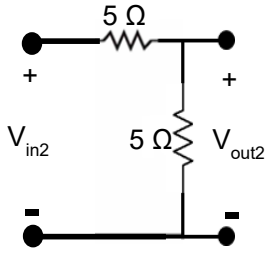
Exercise 2-2 Voltage Follower (Buffer).

1. Given the two circuits below, compute V_{out1} and V_{out2}. Assume $V_{in2} = V_{out1}$

Circuit 1:



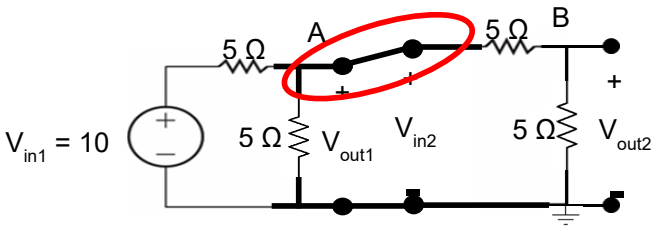
Circuit 2:



$$V_{out1} = 10V \times \frac{5\Omega}{5\Omega + 5\Omega} = 5V$$

$$V_{out2} = V_{out1} \times \frac{5\Omega}{5\Omega + 5\Omega} = 2.5V$$

2. Compute V_{out2} if the two circuits are directly connected:

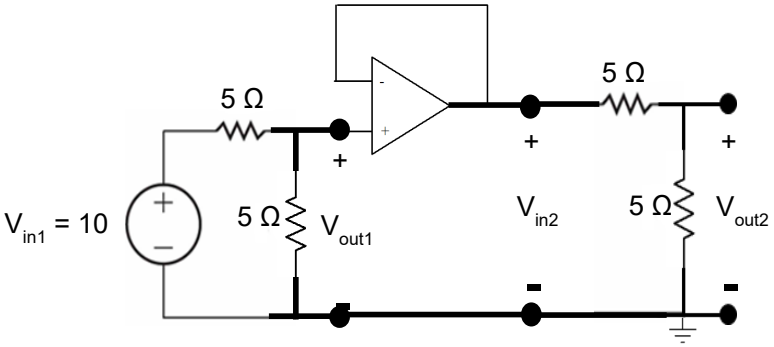


$$R_{eq} = 5\Omega \,||\, (5\Omega + 5\Omega) = \frac{10}{3}\Omega$$

$$V_{out1} = 10V \times \frac{\frac{10}{3}\Omega}{\frac{10}{3}\Omega + 5\Omega} = 4V$$

$$V_{out2} = V_{out1} \times \frac{5\Omega}{5\Omega + 5\Omega} = 2V$$

3. Now compute V_{out2} when the two stages are connected via a voltage follower buffer:



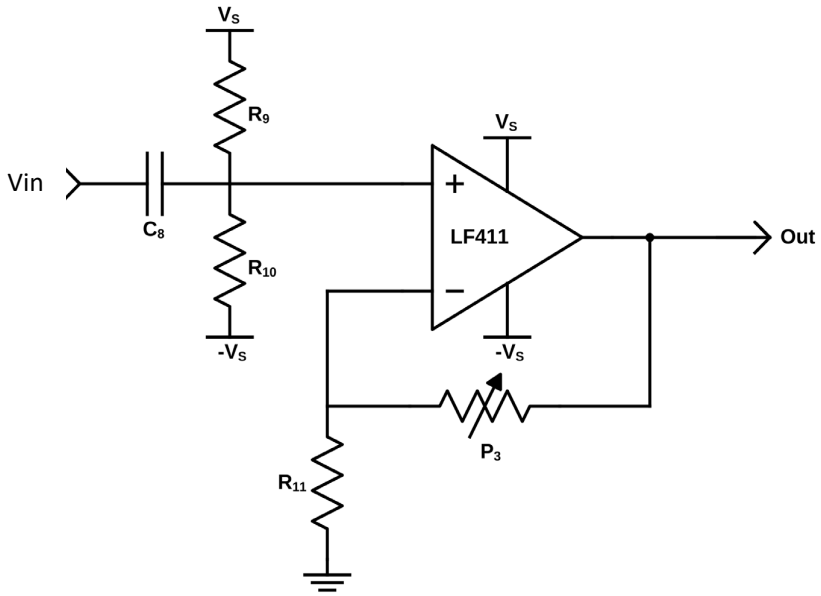
$$V_{out1} = 10V \times \frac{5\Omega}{5\Omega + 5\Omega} = 5V$$

$$V_{out2} = V_{out1} \times \frac{5\Omega}{5\Omega + 5\Omega} = 2.5V$$

4. What effect did the buffer have?

The buffer gives circuit V_{out1} impedance $R_{imp} = \infty\Omega$

<u>Exercise 2-3</u> Non-Inverting Amplifier. Number the pins on the schematic symbol for the non-inverting amplifier.



Compute the range of Voltage gain, Vout/Vin. For the low end, assume the potentiometer is set to 1kΩ. For the max gain, assume the potentiometer is set to 100kΩ:

$V_+ = V_{in} = V_-$

$V_{out} = V_{in} \times \frac{R_{11} + P_3}{R_{11}}$

$\frac{V_{out}}{V_{in}} = \frac{R_{11} + P_3}{R_{11}}$

$P_3 = 1k\Omega$

$\frac{V_{out}}{V_{in}} = \frac{1k\Omega + 1k\Omega}{1k\Omega} = 2$

$P_3 = 100k\Omega$

$\frac{V_{out}}{V_{in}} = \frac{100k\Omega + 1k\Omega}{1k\Omega} = 101$

<u>Exercise 2-4</u> Voltage Gain Graph for non-inverting amplifier. Choose a value of the non-inverting amplifier's voltage gain, A_v, within the range which you calculated in Exercise 2-3. Plot V_{out} versus V_{in}. Label both the saturation and linear regions. **Remember that a real-world OPAMP will reach saturation approximately 1.5V below the supply voltages, so show that here.**

$A_v = 5$

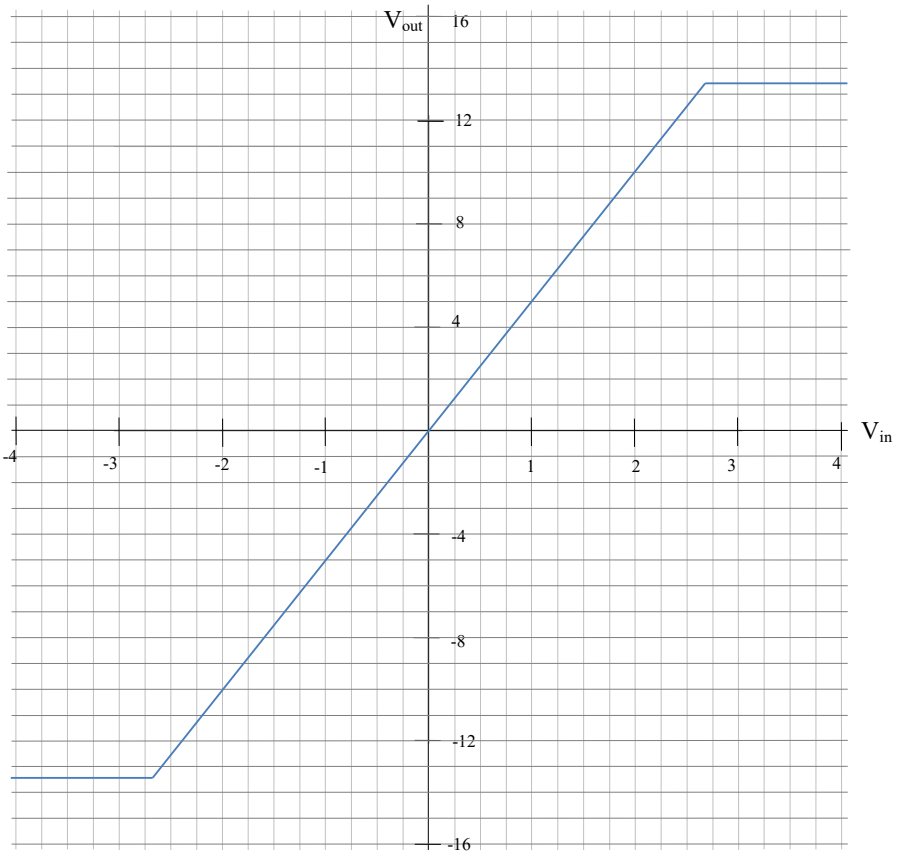


Figure 2. Plot of Av.

Based on your plot of A_v in Figure 2, what is the maximum positive input voltage that can be applied to the non-inverting amplifier before saturation occurs?

$V_{in\ max} = \dfrac{V_{out}}{A_v} = \dfrac{13.5V}{5} = 2.7V$

III. Test Procedure and Equipment

A. Measured component values. For each component measure the actual voltage, resistance or capacitance. These will be the measured values used in theoretical calculations. Enter the measured values in the table below:

	$+V_{CC}$	$-V_{CC}$	R_9	R_{10}	R_{11}	P_3	C_8
Nominal	15V	-15V	1 MΩ	1 MΩ	1 kΩ	100 kΩ	1μF
Measured	15.04V	-15.11V	1.002 MΩ	0.995MΩ	983Ω	103.4 kΩ	1.0031 μF

B. Non-Inverting Amplifier Gain and Phase.

- On the same protoboard as the rest of your Theremin, build the non-inverting OPAMP circuit from Exercise 2-3.
- DO NOT connect it to the rest of the Theremin.
- Set the potentiometer to 1kΩ (Record actual value : 988Ω)
- Apply a small (100mV) sinusoidal input at 100kHz.
- Record the amplitude of both the input and the output signals and their relative phase.
- Set the potentiometer to max value (~100kΩ, whatever you measured for P3 above)
- Record the amplitude of both the input and the output signals and their relative phase. If necessary, decrease the input to avoid saturation.
- Increase the input voltage amplitude until saturation occurs.
- Record the amplitude of both the input and the output signals and their relative phase.

Trial	Input Voltage	Output Voltage	Phase
1 : linear region, P_3 at ~1kΩ	330mV	740mV	0
2 : linear region, P_3 at ~100kΩ	290mV	3.86V	0.14π
3 : saturation	1.33V	25.7V	0.26π

C. Currents and Voltages. Measure and record the voltages at the inverting and noninverting terminals (v_- and v_+).

v_-	v_+
0.013V	0.176V

Measure and record the current flowing into each OPAMP terminal.

i_-	i_+
-0.028mA	0.001mA

V. Results and Discussion

Using the measured component values, calculate the **theoretical** voltage gain, A_v, of each amplifier you built, and enter it in the table below. Now record your **measured** gain for each amplifier and calculate your % error.

Reminder: $$\%Error = \left|\frac{Theoretical - Measured}{Theoretical}\right| \text{ x } 100\%$$

Voltage Gain	Theoretical A_v	Experimental A_v	%Error
Non-inverting Amp. linear	2.04	2.24	9.9%
Non-inverting Amp. saturated	N/A	19.32	N/A

Was your % error acceptable? Discuss the likely sources of error you encountered in this lab:

The % error is 9.9% when the potentiometer is set to minimum. Since the % error is bigger than 3%, the error is not acceptable. The sources of error might include measurement error caused by the P3 value and uncertainty of its position due to friction and modeling error of the amplifier (not ideal).

Do the voltage and current measurements in part IV C agree with ideal op-amp theory? Explain why or why not.

The ideal op-amp theory states that the voltage difference should be 0V and current through two input should be 0A. The voltage difference measured in part IV is 0.163V, the current is close to 0A. Hence, the measured value does not agree with ideal op-amp because the ideal op-amp has nullator between the two input pins which does not exist in real world.

ECE291 – Theremin

Victor Zhang, Jonathan Lam

December 19, 2019

Contents

1 Introduction

The goal of ECE291 is to produce a theremin (instrument), which uses the basic concepts of circuit analysis (ECE240) and operational amplifier circuits. The theremin is composed of two comparator-based square wave generators, whose outputs are low-pass filtered and combined to form a beat frequency. One wave generator uses a copper rod antenna for a variable frequency. The beat is collected by an envelope detector (an op-amp, diode, and RC circuit), and passively low-pass filtered. The resulting signal is then passed through a power amplifier into a speaker to produce music.

2 Theory

The schematic provided in lecture (see Fig. 1) did not provide a clean output signal. Much of the circuit is the same, however, and any deviations are discussed in detail in this section. For comparison, the schematic for the final design described in this report is displayed in Fig. 6.

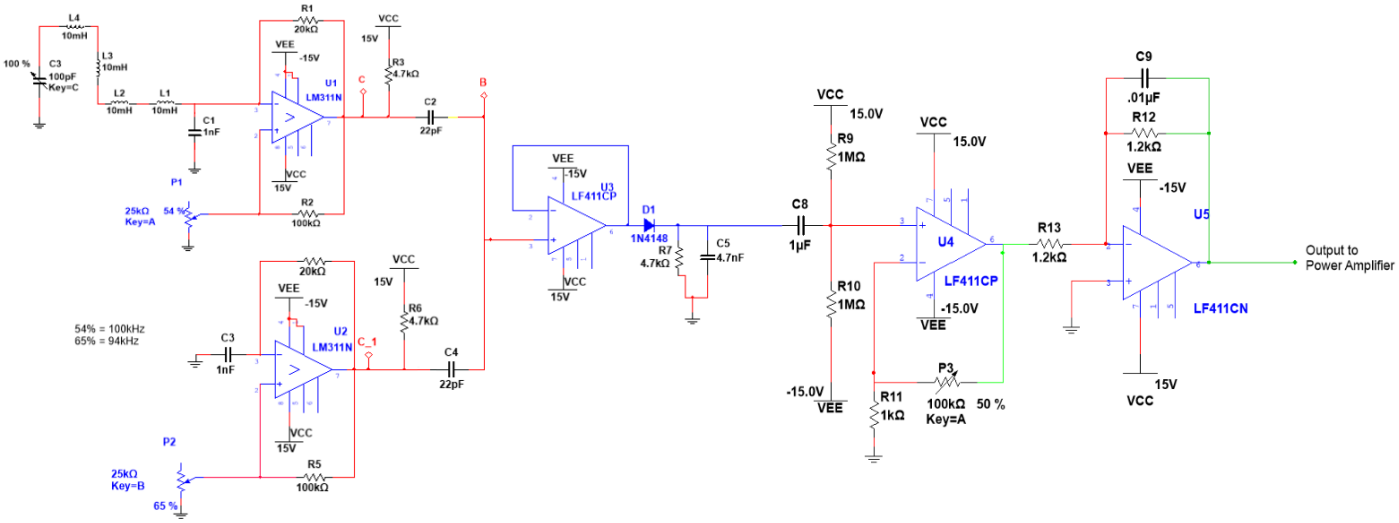


Figure 1: Overall provided block diagram of the Theremin. (Not accurate to the actualized project.)

2.1 Oscillator circuits

There are two square-wave oscillator circuits. Both use the LM311 comparator IC. This comparator is used to perform high-speed switching and desirable high-slew rate/low saturated power consumption (as opposed to the generic LF411 op-amp used for many of the other components in this project.) The variable oscillator circuit is shown in Fig 2, and a very schematic is used for the fixed oscillator circuit (namely, the lack of antenna and the replacement of the 470pF capacitor with a 1nF capacitor). This oscillator provides the desired square wave. However, we aimed to achieve a more symmetric, and larger-amplitude square wave output by replacing the 4.7kΩ pull-up resistor with a 1kΩ resistor. This reduces the changing time from roughly 400ns to 100ns, and increases the square wave's p-p amplitude from roughly 16V to roughly 19V (when VCC=10V, VEE= −10V); this is useful because filtering the square wave (discussed in the next section) greatly decreases the amplitude of the re-

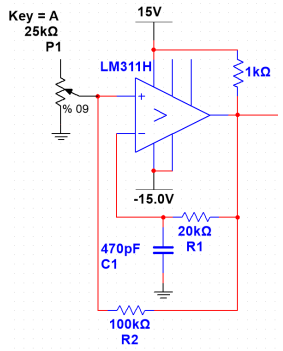


Figure 2: Square-wave (variable) oscillator circuit close-up

sulting wave. However, this did increase power consumption by each oscillator from roughly 0.01A to 0.02A.

Variable oscillator C_1 (pF)	470
Fixed oscillator C_1 (pF)	1000
R_1 (kΩ)	20
R_2 (kΩ)	100
R (kΩ)	1
P_1 (kΩ)	25
Emp. wave amp. (V p-p)	7.2
Emp. wave freq. (Hz)	100.40

Table 1: Square-wave oscillator values

2.1.1 Smoothing the oscillator output

Square waves do not produce desirable beats, but a sinusoid does. (See the Misc. Figures section.) "Smoothing" the square wave into a sinusoid was achieved by applying multiple low-pass filters in series to each oscillator output. The low-pass filters are tuned to have a cutoff frequency of roughly 100kHz, the target oscillator frequency. The chosen RC values are displayed in Table 2.

R (kΩ)	1.5
C (nF)	1

Table 2: Low-pass RC values

The cutoff frequency is calculated as follows.

$$f_c = \frac{1}{2\pi \mathrm{RC}} = \frac{1}{2\pi(1.5 \times 10^3 \Omega)(1 \times 10^{-9} \mathrm{F})} = 106\mathrm{kHz} \tag{1}$$

which is close to the desired 100kHz oscillation frequency. Four low-pass filters are used directly after the square wave oscillator to approximate a sinusoidal wave. This was then followed by an amplifier (Fig. 3), as each low-pass filter attenuated the oscillation's amplitude.

R_3 (kΩ)	1
R_F (kΩ)	15

Table 3: Inverting amplifier results

The amplification for the (inverting) amplifier is the known result shown below.

$$A_V = -\frac{R_F}{R_3} = -\frac{15\Omega}{1\Omega} = 15 \tag{2}$$

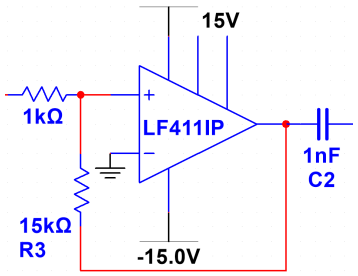


Figure 3: Inverted amplifier schematic close-up

2.1.2 Variable oscillator (with antenna)

A copper pipe, attached in series with four inductors, acted as a one-terminal capacitor, in series with the capacitor for one of the square wave oscillators. Its capacitance varies as the distance between the rod and other objects varies.

2.1.3 Other attempts to improve signal quality

We attempted to increase the carrier frequency to roughly 200kHz according to Prof. Frost's [2] advice in order to get a higher-quality envelope. However, this caused strange and unpredictable results with the LF411 op-amp circuits. Changing the higher carrier frequency back to 100kHz appeared to fix the anomalies.

We had multiple attempts to generate a sinusoidal wave in order to get a more defined beat frequency. One possibility we attempted was the Wien bridge oscillator [3], which is an op-amp circuit that uses a notch filter (high- and low-pass filter) on the positive feedback to force a self-reinforcing sinusoidal wave at its natural frequency. This design worked very well to generate a stable

sinusoidal wave, but it could not be used for the variable oscillator because it requires two capacitors with the same capacitance (one for each the high- and low-pass filter). Another sinusoidal oscillator we tested was a notch filter circuit. While this also produced nice sine waves, there was a very large amplitude modulation with a change in frequency. This amplitude modulation caused by the changes in frequency is still present in the current circuit design, but to lesser degrees as the low-pass and high-pass filters are better-tuned to the desired frequencies.

In another attempt to obtain a clearer beat frequency from two mixed square wave generators, a summing amplifier was prototyped and tested under the presumption that this might allow for truer beats by summing the two waves. However, this produced an output insignificantly different from the simple decoupling capacitor "mixer," so we stuck to the simpler capacitor design.

Another attempt to improve the symmetry of the square wave was to use a 555 timer chip. However, this did not perform well in the 100-200kHz frequency range, producing very triangular wave and contradicting our purpose for using them.

2.2 Oscillator mixer

The two oscillator circuits are joined using an capacitor "mixer." The inputs are driven with buffers, so we can expect the output to act like two capacitors driven by two sinusoidal ideal voltage sources. This output should have a beat frequency, according to the well-known result for beat frequencies, shown in Equation 3:

$$\cos\omega_1 t + \cos\omega_2 t = 2\cos\left(\frac{\omega_1+\omega_2}{2}t\right)\cos\left(\frac{\omega_1-\omega_2}{2}t\right) \tag{3}$$

which has the "fast" carrier modulation of $\frac{1}{2}(\omega_1+\omega_2)$ and "beat frequency" of $\frac{1}{2}(\omega_1-\omega_2)$. The values for the mixer are shown in Table 4.

C_2 (nF)	1

Table 4: Capacitor "mixer" values

2.3 Envelope Detector

The output of the oscillator mixer (with the beats) is fed into a voltage follower and then an AM (envelope) detector. This involves a diode (D_1) and a capacitor and resistor in parallel to ground, with the recommended values (Fig. 4).

2.4 Passive filtering and amplifier

The final low-pass filter (Fig. 5) is used to attenuate any high frequencies left behind by the carrier signal. Since we only wish to hear noises in the human

R_4 (kΩ)	4.7
C_3 (nF)	4.7

Table 5: Envelope detector values

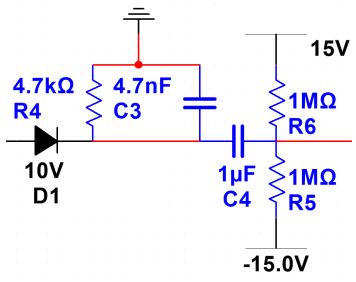


Figure 4: Envelope detector schematic close-up

hearing range (i.e., frequencies above 20kHz are squeaky, annoying, and may cause damage to the ear), the low-pass filter is tuned with a cutoff frequency of 20kHz. Additionally, we (as with many other project groups) noticed an ambient, small 60Hz overall modulation that appears after the diode, so there is also one high-pass filter with an identical cutoff frequency, and resistor/capacitor values (due to the similar nature of high- and low-pass RC filters). This creates a small "frequency notch"; i.e., the amplitude of the output frequency is most strongly transmitted at 20kHz, strongly attenuated at higher frequencies, and weakly attenuated at lower frequencies.

R (kΩ)	8.2
C (nF)	10

Table 6: Final high- and low-pass RC filter values

The cutoff frequency is calculated as follows.

$$f_c = \frac{1}{2\pi \mathrm{RC}} = \frac{1}{2\pi(8.2 \times 10^3 \mathrm{k\Omega})(10 \times 10^{-9} \mathrm{F})} = 19.4 \mathrm{kHz} \tag{4}$$

Six passive low-pass filter circuits are connected in series to attempt to smooth the envelope to approach a sinusoidal wave, similar to the filtering on the square wave oscillator. This replaces the final amplifier and active low-pass filter from the original recommended schematic. The (inverting) amplifier uses a variable resistor to amplify and vary the output amplitude. As with the other inverting amplifiers, the gain is given by:

$$A_V = -\frac{P_2}{R_3} \Rightarrow 1 \leq |A_V| \leq 10 \tag{5}$$

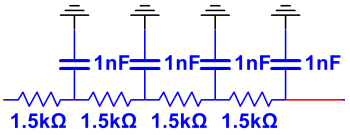


Figure 5: Post-AM detector passive low-pass filter close-up. Note that the first filter is a high-pass filter designed to filter out the 60Hz noise

R_7 (kΩ)	1
P_2 (kΩ)	10

Table 7: Final inverting amplifier values

This amplitude modulation serves as a tunable audio amplitude (i.e., volume) control.

3 Schematic and setup

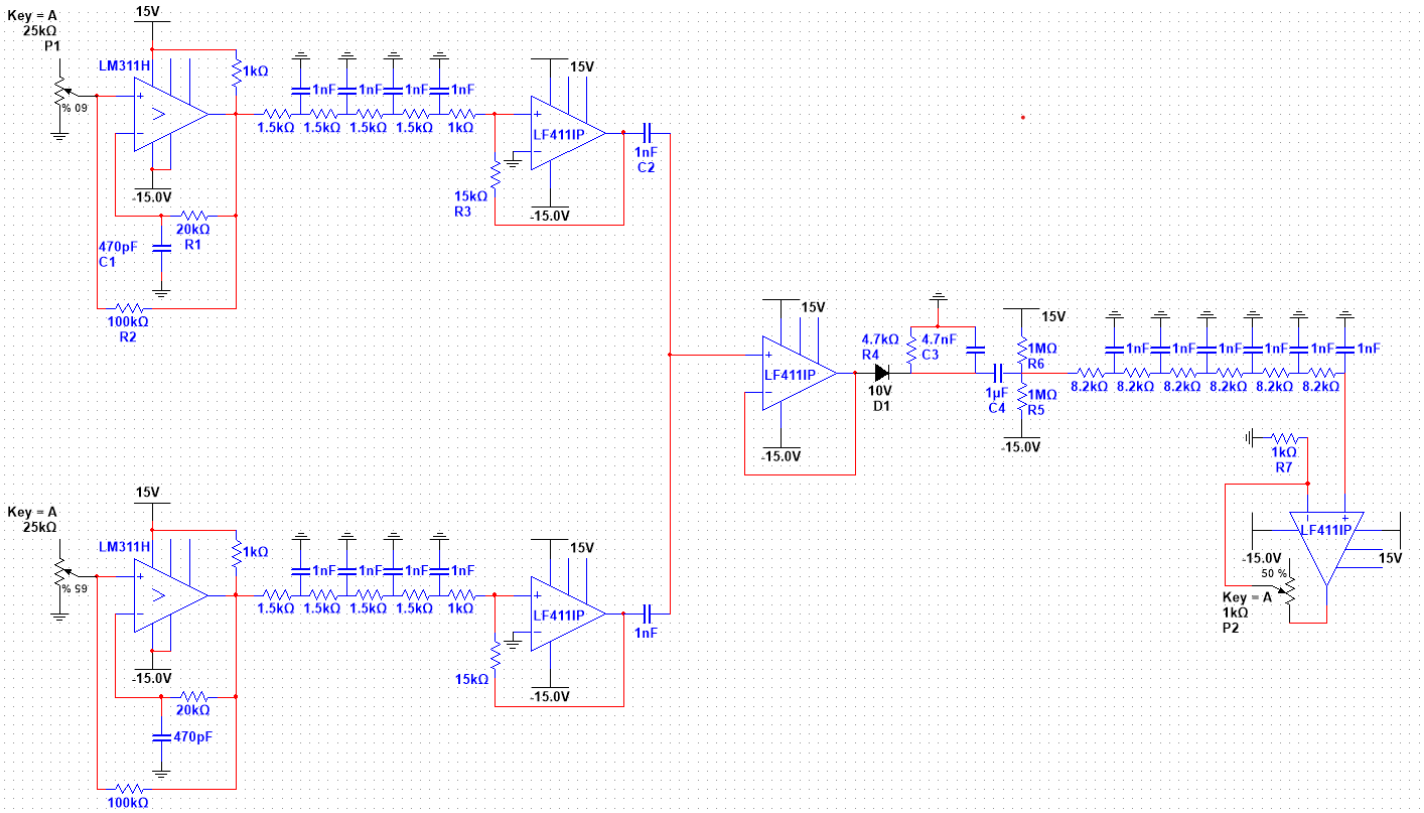


Figure 6: Main schematic

There are a few clearly visually distinguishing factors from Fig. 1, the originally recommended schematic. The low-pass filters between the square-wave oscillators and the capacitor mixer, and the low-pass filters between the envelope detector and the final amplifier are the most pronounced and the major deviation in our circuit design from the original.

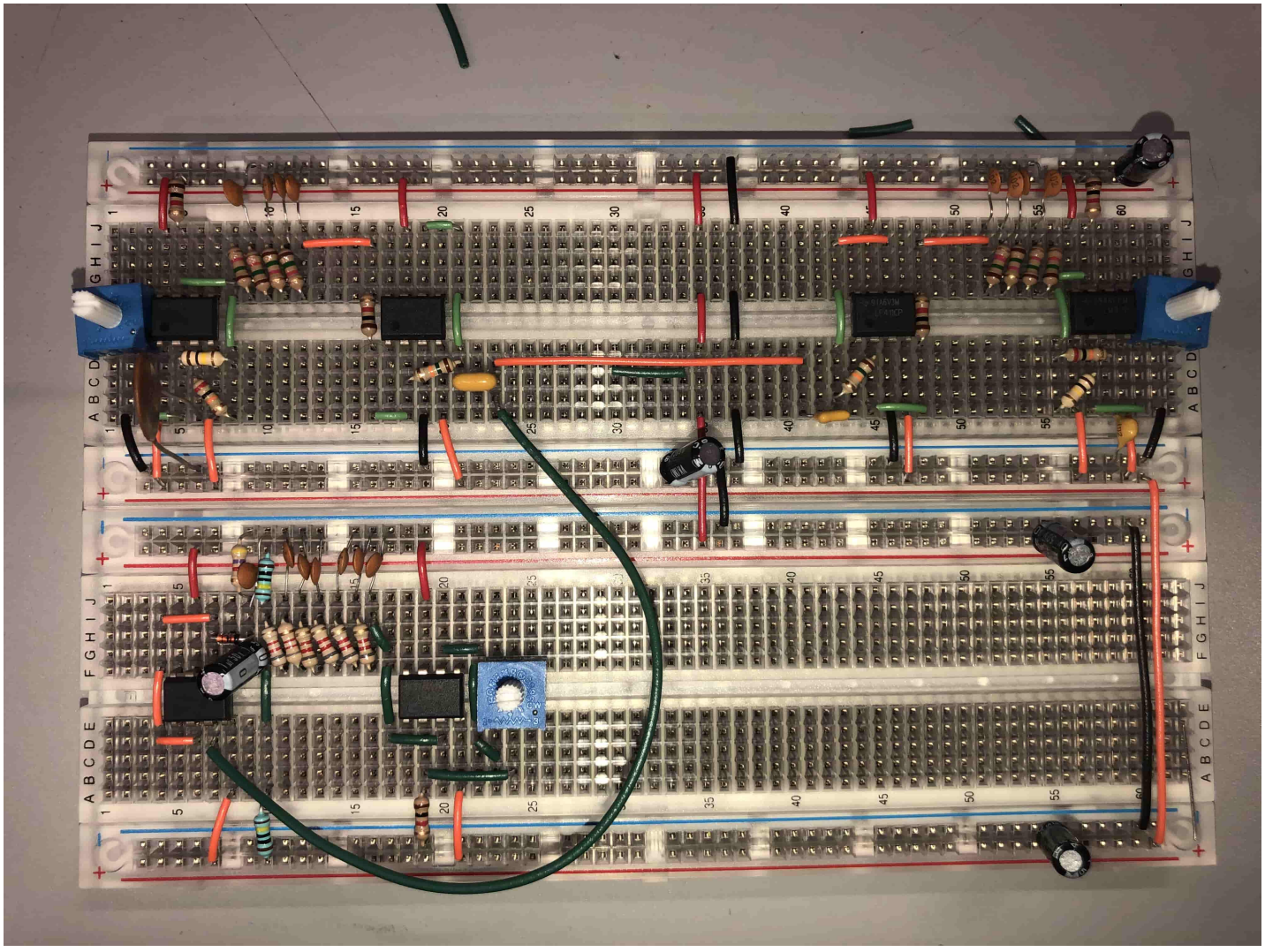
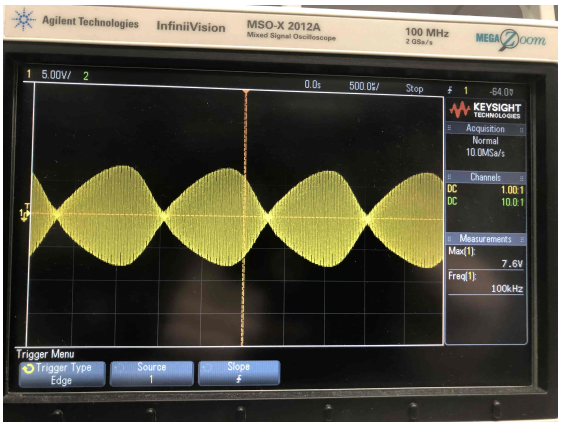


Figure 7: Experimental setup

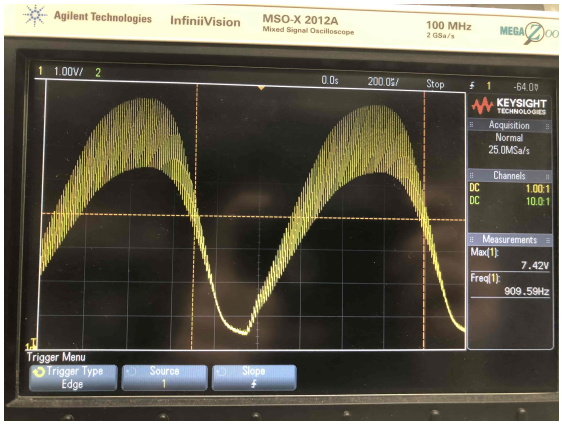
4 Results

The evolution of the signal through the circuit is very pronounced. The square wave generators produce nearly 100kHz, nearly-square waves (Fig. 8.a). The oscillator inputs are mixed to produce beats (Fig. 8.b). (Not shown is the low-pass filtering of the square waves before mixing.) The AM detector picks up the signal (Fig. 8.c), which is filtered sinusoid (Fig. 8.d). The carrier signal used was 100kHz, and the beat frequencies were roughly on the scale of 1-20kHz.

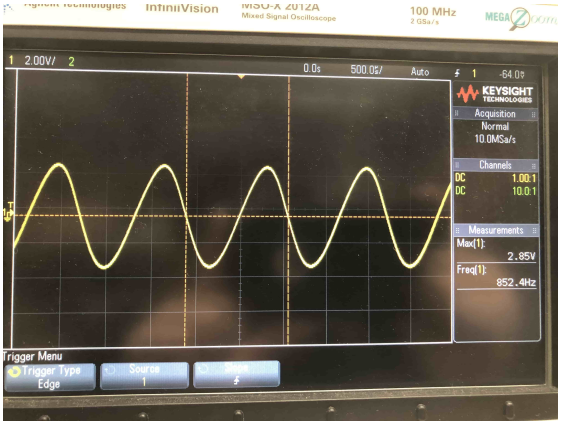
(a) Square wave oscillator input



(b) Oscillator beats



(c) Detected envelope



(d) Audio output

Figure 8: Signal evolution through circuit

4.1 Subjective description of theremin instrument

This iteration of the theremin tended to produce clean sinusoidal audio signals. The beat (and output) frequency tended to be stable in a range of roughly 300Hz to 20kHz. Subjectively, the most pleasing tones were in the 800-5000Hz range, where higher frequencies tended to be squeaky and aggravating. As mentioned in the Theory section, there tended to be some amplitude modulation that accompanied the frequency modulation, which we postulate to be due to the low-pass filters' frequency-dependent attenuation. The capacitor was fairly responsive to changes in distance, and was able to noticeably change pitch as people moved as much as a meter away from the antenna.

We were unsuccessful in playing the Cooper Union theme song on this instrument =(.

5 Conclusions

The theremin project introduces the use of the op-amp and comparator, and provides an opportunity to work with AC signal which magnifies the common modeling error in circuit analysis. Students learnt to reduce or avoid unwanted circuit behavior which is critical for implementing electrical engineering in real world. The theremin works as expected and produces relatively clean sound wave. Multiple prototyping iterations were gone through to optimize the circuit. The design process used can be significantly improved by doing more thorough modelling and testing. The students who conducted this experiment learnt how to generate high-frequency signals, filter (high and low frequencies), amplify, voltage-forward, invert, DC-block, and detect AM using various op-amp implementations. There was also useful learning in practical circuit-building and debugging skills, as well as learning how to deal with noisy, high-frequency, and audio signals.

References

[1] Shay, Lisa.

[2] Frost, Brian.

[3] "Wien Bridge Oscillator Tutorial and Theory." *Basic Electronics Tutorials*, Aspencore, 21 Feb. 2018, https://electronics-tutorials.ws/oscillator/wien_bridge.html.

PSET 1

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

Problem 1: What is a system call and what is not?

name	syscall?	triggers?	triggers what?
read	Y	-	-
fputc	N	M - only on buffer flush	write
strcpy	N	N	-
sqrt	N	N	-
malloc	N	M - if program runs out of heap space	sbrk, sometimes brk/mmap
fopen	N	Y	open/openat
strerror	N	N	-
isalpha	N	N	-
atoi	N	N	-
scanf	N	Y	read
return	N	M - if returning from main (ending the process)	_exit

Problem 2: Error messages

A) close is called with a parameter of -1
EBADF Bad file descriptor

B) write is made to a file which resides on a disk that is completely full
ENOSPC No space left on device

C) open is called with its first parameter referring to a non-existent file and second parameter of O_RDONLY
ENOENT No such file or directory

D) write is made with the second parameter of 0, a first parameter which refers to a valid fd open for writing, and a third parameter >0
EFAULT Bad address

Problem 3: Use of system calls in a simple concatenation program

kitty.c

```
#include <fcntl.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
```

PSET 1

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

```
#define BUF_SIZE 4096
int main(int argc, char **argv) {
  int ofd, ifd, flen, rlen, wlen, is_bin, rwcnt;
  char buf[BUF_SIZE], *bufp, *fnames[argc], **fnamep, *out_file = NULL,
       *errop, *errctx, *errctm;
  memset(fnames, 0, argc*sizeof(char *));

  // parse args
  for(fnamep = fnames, ++argv; --argc; ++argv)
    if(!strcmp(*argv, "-o")) {
      if(!--argc) {
        errop = "parsing", errctx = "args", errctm = "No output file after -o";
        goto fail;
      }
      out_file = *++argv;
    } else
      *fnamep++ = *argv;

  // open out_file using creat (open w/ flags O_CREAT|O_WRONLY|O_TRUNC)
  if(out_file) {
    if((ofd = creat(out_file, 0666)) == -1) {
      errop = "creating (for writing)", errctx = out_file;
      goto fail;
    }
  } else
    ofd = 1;

  // if no input file specified (fnames empty), add std. input to input list
  if(!*fnames)
    *fnames = "-";

  // loop through and open input files, concatenate to output
  for(fnamep = fnames; *fnamep; fnamep++) {
    if(strcmp(*fnamep, "-")) {
      if((ifd = open(*fnamep, O_RDONLY)) == -1) {
        errop = "opening (for reading)", errctx = *fnamep;
        goto fail;
      }
    } else
      ifd = 0;

    // attempt reading file
    flen = is_bin = rwcnt = 0;
    while(rlen = read(ifd, buf, BUF_SIZE)) {
      if(rlen == -1) {
        errop = "reading of", errctx = *fnamep;
```

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

```
        goto fail;
      }

      // write to output file
      if((wlen = write(ofd, buf, rlen)) == -1) {
        errop = "writing to", errctx = out_file;
        goto fail;
      }

      // account for partial write scenario
      if(wlen != rlen) {
        errop = "writing to", errctx = out_file, errctm = "Partial write";
        goto fail;
      }

      // add to total length (in bytes) and read/write count
      flen += rlen;
      ++rwcnt;

      // check if file includes binary chars
      if(!is_bin)
        for(bufp = buf; bufp-buf < rlen; bufp++)
          if((*bufp < 32 || *bufp >= 127) && !(*bufp >= 9 && *bufp <= 13)) {
            is_bin = 1;
            break;
          }
    }

    // report bytes transferred for file
    fprintf(stdout, "%s%s: %d bytes transferred. %d read/write call(s).\n",
            ifd ? *fnamep : "<standard input>", is_bin ? " [BINARY]" : "",
            flen, rwcnt);
  }

  return 0;

fail:
  fprintf(stderr, "Error: %s %s: %s\n",
          errop, errctx, errno ? strerror(errno) : errctm);
  return -1;
}
```

kitty.c sample runs

```
(base) [jon@archijon programs]$ ./kitty -o file1
Hello, world!
```

PSET 1

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

```
This is file1

<standard input>: 29 bytes transferred. 3 read/write call(s).
(base) [jon@archijon programs]$ echo 'This is file2' > file2
(base) [jon@archijon programs]$ echo -e 'file3\nfile3\nfile3' > file3
(base) [jon@archijon programs]$ ./kitty file1 file2 file3
Hello, world!
This is file1

file1: 29 bytes transferred. 1 read/write call(s).
This is file2
file2: 14 bytes transferred. 1 read/write call(s).
file3
file3
file3
file3: 18 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ ./kitty file1 file2 file3 -o file4
file1: 29 bytes transferred. 1 read/write call(s).
file2: 14 bytes transferred. 1 read/write call(s).
file3: 18 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ ./kitty file4
Hello, world!
This is file1

This is file2
file3
file3
file3
file4: 61 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ dd if=/dev/urandom of=rand1 bs=1M count=50
50+0 records in
50+0 records out
52428800 bytes (52 MB, 50 MiB) copied, 0.310224 s, 169 MB/s
(base) [jon@archijon programs]$ ./kitty rand1 -o rand2
rand1 [BINARY]: 52428800 bytes transferred. 12800 read/write call(s).
(base) [jon@archijon programs]$ sha256sum rand1 rand2
f5c0c772512b1b177fa7144e92d637c0a7d608b75e22601646f9b7e15c5d9870  rand1
f5c0c772512b1b177fa7144e92d637c0a7d608b75e22601646f9b7e15c5d9870  rand2
(base) [jon@archijon programs]$ ./kitty -o file5 -
This is to go in file5
Hello, world!
<standard input>: 37 bytes transferred. 2 read/write call(s).
(base) [jon@archijon programs]$ ./kitty - - file5 - -o file6
This is to
go in
<standard input>: 17 bytes transferred. 2 read/write call(s).
file6:
```

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

```
<standard input>: 7 bytes transferred. 1 read/write call(s).
file5: 37 bytes transferred. 1 read/write call(s).
End of file6.
<standard input>: 14 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ ./kitty file6
This is to
go in
file6:
This is to go in file5
Hello, world!
End of file6.
file6: 75 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ ./kitty kitty.c
/**
  * kitty - concatenate and copy files
  *
                                    [TRUNCATED]
          errop, errctx, errno ? strerror(errno) : errctm);
  return -1;
}
kitty.c: 2843 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ ./kitty kitty
ELF>��;@8
          @@@@h���    ?-?=?=???-?=?=????DDP?td? ? ? 44Q?tdR?td?-?=?=/lib64/ld-linux-
                                    [TRUNCATED]
?  ?? ? 4? ! !??????=?-??????@?h@h?@x00 ?0x0??0^??7??:kitty [BINARY]: 17112 bytes
transferred. 5 read/write call(s).
(base) [jon@archijon programs]$ ./kitty kitty kitty.c /usr/bin/cat -o kittykittycat
kitty [BINARY]: 17112 bytes transferred. 5 read/write call(s).
kitty.c: 2843 bytes transferred. 1 read/write call(s).
/usr/bin/cat [BINARY]: 38952 bytes transferred. 10 read/write call(s).
(base) [jon@archijon programs]$ ./kitty -o -

This is to go inside the file "-". This can be kittied using ./-
<standard input>: 66 bytes transferred. 2 read/write call(s).
(base) [jon@archijon programs]$ ./kitty ./-

This is to go inside the file "-". This can be kittied using ./-
./-: 66 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ ./kitty kitty.c -o
Error: parsing args: No output file after -o
(base) [jon@archijon programs]$ ./kitty nonexistentfile.txt
Error: opening (for reading) nonexistentfile.txt: No such file or directory
(base) [jon@archijon programs]$ touch badpriv
(base) [jon@archijon programs]$ chmod 000 badpriv
(base) [jon@archijon programs]$ ./kitty badpriv
Error: opening (for reading) badpriv: Permission denied
```

PSET 1

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

kitty.c

```
#include <fcntl.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>

#define BUF_SIZE 4096
#define MAX_PAR_WRITE_RETRY 256

int main(int argc, char **argv) {
  int ofd = -1, ifd = -1, flen, rlen, wlen, is_bin, rcnt, wcnt, par_retry_count;
  char buf[BUF_SIZE], *bufp, *fnames[argc+1], **fnamep, *out_file = NULL,
       *errop, *errctx, *errctm;
  memset(fnames, 0, (argc+1)*sizeof(char *));

  // parse args; only handles -o argument (handles others as filenames)
  for(fnamep = fnames, ++argv; --argc; ++argv)
    if(!strcmp(*argv, "-o")) {
      if(!--argc) {
        errop = "parsing", errctx = "args", errctm = "No output file after -o";
        goto fail;
      }
      out_file = *++argv;
    } else
      *fnamep++ = *argv;

  // open out_file using creat (open w/ flags O_CREAT|O_WRONLY|O_TRUNC)
  if(out_file) {
    if((ofd = creat(out_file, 0666)) == -1) {
      errop = "creating (for writing)", errctx = out_file;
      goto fail;
    }
  } else
    ofd = 1;

  // if no input file specified (fnames empty), add std. input to input list
  if(!*fnames)
    *fnames = "-";

  // loop through and open input files, concatenate to output
  for(fnamep = fnames; *fnamep; fnamep++) {
    if(strcmp(*fnamep, "-")) {
      if((ifd = open(*fnamep, O_RDONLY)) == -1) {
        errop = "opening (for reading)", errctx = *fnamep;
        goto fail;
```

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

```
      }
    } else
      ifd = 0;

    // attempt reading file
    flen = is_bin = rcnt = wcnt = 0;
    while(rcnt++, rlen = read(ifd, buf, BUF_SIZE)) {
      if(rlen == -1) {
        errop = "reading of", errctx = *fnamep;
        goto fail;
      }

      // write to output file
      // account for partial write scenario; most likely due to a pipe/socket
      // with a small buffer; keep retrying until exceeded maximum tries or
      // write complete; while loop breaks when buffer successfully written
      wlen = 0, par_retry_count = 0;
      while(wcnt++, (wlen += write(ofd, buf+wlen, rlen-wlen)) != rlen) {
        if(++par_retry_count == MAX_PAR_WRITE_RETRY) {
          errop = "writing to", errctx = out_file, errctm = "Partial write";
          goto fail;
        }

        // write error
        if(wlen == -1) {
          errop = "writing to", errctx = out_file;
          goto fail;
        }
      }

      // add to total length (in bytes) and read/write count
      flen += rlen;

      // check if file includes binary chars
      if(!is_bin)
        for(bufp = buf; bufp-buf < rlen; bufp++)
          if((*bufp < 32 || *bufp >= 127) && !(*bufp >= 9 && *bufp <= 13)) {
            is_bin = 1;
            break;
          }
    }

    // close input file
    if(ifd > 2 && close(ifd) == -1) {
      errop = "closing", errctx = *fnamep;
      goto fail;
    }
```

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

```
    // report bytes transferred for file to stderr
    fprintf(stderr, "%s%s: %d bytes transferred. %d read / %d write call(s).\n",
            ifd ? *fnamep : "<standard input>", is_bin ? " [BINARY]" : "",
            flen, rcnt, wcnt);
  }

  // close output file and exit
  if(ofd > 2 && close(ofd) == -1) {
    errop = "closing", errctx = out_file, ofd = -1;
    goto fail;
  }
  return 0;

fail:
  fprintf(stderr, "Error: %s %s: %s\n",
          errop, errctx, errno ? strerror(errno) : errctm);

  // attempt to close input/output files
  // silently fail here because files will automatically be closed anyway
  // and to avoid extra errors printed to screen
  if(ofd != -1)
    close(ofd);
  if(ifd != -1)
    close(ifd);
  return -1;
}
```

PSET 1

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

Example output

```
(base) [jon@archijon programs]$ echo -e 'Hello, world!\nThis is file1\n\ntesting' >
file1
(base) [jon@archijon programs]$ ./kitty -o file2
This is file2
<standard input>: 14 bytes transferred. 2 read / 1 write call(s).
(base) [jon@archijon programs]$ cat > file3
file3, file3, file3
(base) [jon@archijon programs]$ ./kitty file1 file2 file3 -o file4 && ./kitty file4
file1: 37 bytes transferred. 2 read / 1 write call(s).
file2: 14 bytes transferred. 2 read / 1 write call(s).
file3: 20 bytes transferred. 2 read / 1 write call(s).
Hello, world!
This is file1

testing
This is file2
file3, file3, file3
file4: 71 bytes transferred. 2 read / 1 write call(s).
(base) [jon@archijon programs]$ dd if=/dev/urandom of=rand bs=1M count=50
50+0 records in
50+0 records out
52428800 bytes (52 MB, 50 MiB) copied, 0.307862 s, 170 MB/s
(base) [jon@archijon programs]$ ./kitty rand -o rand2
rand [BINARY]: 52428800 bytes transferred. 12801 read / 12800 write call(s).
(base) [jon@archijon programs]$ cat rand > rand3
(base) [jon@archijon programs]$ sha256sum rand rand2 rand3
901f72e6755ab3186fa0f1c80dc9773c19ec44aa7d53ae8543fff03276da2e86  rand
901f72e6755ab3186fa0f1c80dc9773c19ec44aa7d53ae8543fff03276da2e86  rand2
901f72e6755ab3186fa0f1c80dc9773c19ec44aa7d53ae8543fff03276da2e86  rand3
(base) [jon@archijon programs]$ ./kitty -o file5 -
This is to go in file5
Hello, world!
<standard input>: 37 bytes transferred. 3 read / 2 write call(s).
(base) [jon@archijon programs]$ ./kitty - - file5 - -o file6
This is to
go in
<standard input>: 17 bytes transferred. 3 read / 2 write call(s).
file6
<standard input>: 6 bytes transferred. 2 read / 1 write call(s).
file5: 37 bytes transferred. 2 read / 1 write call(s).
End of file6
<standard input>: 13 bytes transferred. 2 read / 1 write call(s).
(base) [jon@archijon programs]$ ./kitty file6
This is to
go in
```

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

```
file6
This is to go in file5
Hello, world!
End of file6
file6: 73 bytes transferred. 2 read / 1 write call(s).
(base) [jon@archijon programs]$ ./kitty -o file7 kitty.c kitty
kitty.c: 3511 bytes transferred. 2 read / 1 write call(s).
kitty [BINARY]: 17120 bytes transferred. 6 read / 5 write call(s).
(base) [jon@archijon programs]$ cat kitty.c kitty > file8
(base) [jon@archijon programs]$ sha256sum file7 file8
a8d8e491f40d9812b3d9f502e3a5e07f247d0ae51aacfde2a9c042d1389adf8e  file7
a8d8e491f40d9812b3d9f502e3a5e07f247d0ae51aacfde2a9c042d1389adf8e  file8
(base) [jon@archijon programs]$ ./kitty file7
#include <fcntl.h>
#include <stdio.h>
#include <string.h>
```

[TRUNCATED]

```
tag.gnu.hash.dynsym.dynstr.gnu.version.gnu.version_r.rela.dyn.rela.plt.init.text.fi
ni.rodata.eh_frame_hdr.eh_frame.init_array.fini_array.dynamic.got.got.plt.data.bss.
comment@#@@$6@@ D@@No
@  @@ @ 4@(!(!@@@@@@=@@@?@@@@p@p@@@0 @0@0@0@@o@@@7@@:file7 [BINARY]: 20631 bytes
transferred. 7 read / 6 write call(s).
(base) [jon@archijon programs]$ ./kitty -o ./-
This is the file "-"
<standard input>: 21 bytes transferred. 2 read / 1 write call(s).
(base) [jon@archijon programs]$ ./kitty ./-
This is the file "-"
./-: 21 bytes transferred. 2 read / 1 write call(s).
(base) [jon@archijon programs]$ ./kitty -o
Error: parsing args: No output file after -o
(base) [jon@archijon programs]$ ./kitty nonexistentfile.txt
Error: opening (for reading) nonexistentfile.txt: No such file or directory
(base) [jon@archijon programs]$ touch badpriv
(base) [jon@archijon programs]$ chmod 000 badpriv
(base) [jon@archijon programs]$ ./kitty badpriv
Error: opening (for reading) badpriv: Permission denied
(base) [jon@archijon programs]$ ./kitty
hello
hello
world
world
!
!
<standard input>: 14 bytes transferred. 4 read / 3 write call(s).
```

PSET 2

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/25/19

Problem 1: A simple hypothetical filesystem

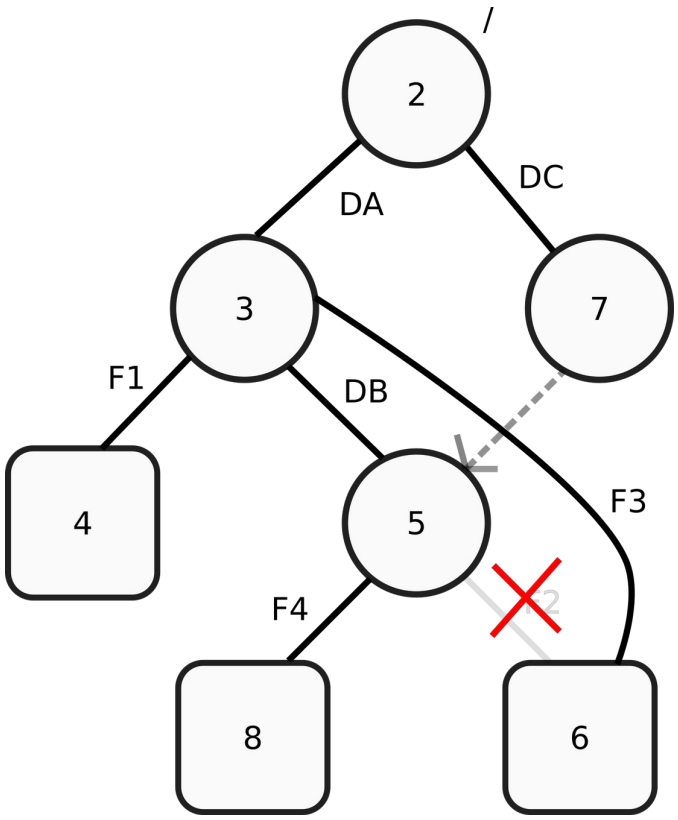
Inode tables:

<table>
<tr><th>#</th><th>type</th><th>nlink</th><th>direntry table (if dir)</th></tr>
<tr><td>2</td><td>dir</td><td>3</td><td><table><tr><th>inode</th><th>name</th></tr><tr><td>2</td><td>.</td></tr><tr><td>2</td><td>..</td></tr><tr><td>3</td><td>DA</td></tr><tr><td>7</td><td>DC</td></tr></table></td></tr>
<tr><td>3</td><td>dir</td><td>3</td><td><table><tr><th>inode</th><th>name</th></tr><tr><td>3</td><td>.</td></tr><tr><td>2</td><td>..</td></tr><tr><td>4</td><td>F1</td></tr><tr><td>5</td><td>DB</td></tr><tr><td>6</td><td>F3</td></tr></table></td></tr>
<tr><td>4</td><td>file</td><td>1</td><td>–</td></tr>
<tr><td>5</td><td>dir</td><td>2</td><td><table><tr><th>inode</th><th>name</th></tr><tr><td>5</td><td>.</td></tr><tr><td>3</td><td>..</td></tr><tr><td>8</td><td>F4</td></tr></table></td></tr>
<tr><td>6</td><td>file</td><td>1</td><td>–</td></tr>
<tr><td>7</td><td>symlink</td><td>1</td><td>–</td></tr>
<tr><td>8</td><td>file</td><td>1</td><td>–</td></tr>
</table>

PSET 2

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/25/19

Filesystem tree (/ is mounted at /mnt of root fs)



Problem 2: Exploratory questions

A) Data corruption without journaling

What program will need to be run before the volume can be mounted?
fsck is the command to check for data corruption (inconsistencies on the disk).

What sort of issues do we expect this program will find with the volume? Give a specific example of at least one issue.
An example of data corruption fsck can notice is incomplete file deletion or insertion caused by the abrupt power off. What fsck does is recursively traverse the filesystem tree from the root, checking inode and free block maps, and then checking for orphaned inodes and free blocks. For example, during a file creation, if power was cut after creating an inode but before it is inserted into a directory's data block, fsck will detect the orphaned inode. Similarly, during file deletion, if the inode is deleted before the directory entry is removed (or vice

versa, depending on the implementation of remove), then fsck should find this inconsistency as well.

If this is a 2TB volume with 1,000,000 allocated inodes, will it take a long time? Why or why not?
Given that fsck has to traverse the entire filesystem tree and scan the entire inode table in order to check for inconsistencies. Since fsck has to go through the data blocks starting from root and check associated inodes, and it has to scan through each data block (~2TB/block size, a very large number) to make sure that free data blocks are not claimed by any files, this will take a long time.

B) EACCESS when deleting /dir/foo
If the user doesn't have write permissions to dir, then deleting the file (i.e., changing the directory name-inode table) is not allowed. Also, if the user doesn't have execute/traverse permissions to dir or root, then they cannot traverse root (/) or dir (/dir) (and therefore do not have permissions to do any action on foo since it traverses dir.

C) Possible factors:
Metadata/indirect blocks: The size of a volume includes more than only file contents (i.e., data blocks): it also contains the other parts of the unix filesystem, such as the inode table and indirect blocks in ext3 (or extend descriptors in ext4 data blocks).

Definition of 4TB (marketing scamminess): this can be a base 10 number (4*10^12) or a base 2 number (2^42). 4000 videos that are each 2^30 bytes long would have a total size (in data blocks alone) of 4.29*10^12 bytes, which is significantly larger than the base 10 version, while the base 2 version has 4.39*10^12 bytes (which would be sufficient and was what the user was looking for).

Reserve factor (possibly): for performance, it helps to leave free blocks abundant so files can be extended into adjacent free blocks (to avoid more disk activity and therefore slower times in mechanical hard drives), i.e., to reduce fragmentation. This may mean leaving some space that cannot be allocated, therefore effectively shrinking hard drive size from the actual capacity.

D) Causes of different move speeds
It is possible that F1 and F2 are on the same volume, but F2 and F3 are not on the same volume. This is possible because it is possible that Z is mounted to a different volume (but looks like an ordinary directory because of the VFS), and therefore the mv command copies the files instead of linking/unlinking (since hard links cannot be made across volumes). Copying a large file is much slower than linking/unlinking.

PSET 2

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
10/2/19

PROGRAM SOURCE (rls.c)

```
#include <ctype.h>
#include <dirent.h>
#include <errno.h>
#include <grp.h>
#include <pwd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/sysmacros.h>
#include <sys/types.h>
#include <time.h>
#include <unistd.h>

// logging to stderr
#define ERROR_NON_FATAL(action, ctx, msg) \
  fprintf(stderr, "rls: %s \"%s\": %s\n", action, ctx, msg);

#define ERROR_FATAL(action, ctx, msg) {\
    fprintf(stderr, "rls: ERROR: %s \"%s\": %s\n", action, ctx, msg);\
    exit(EXIT_FAILURE);\
  }

// conditions for printing; used twice hence the macro
#define should_print()\
      (!optm || ((optm > 0 && stime - f_stat.st_mtime > optm)\
             || (optm < 0 && stime - f_stat.st_mtime <= -optm)))\
       && rperm(&f_stat, u_info)

// struct to store user/group info for optu
struct u_info {
  int optu;
  uid_t uid;
  gid_t *groups;
  int ngroups;
};

// get whether or not a user has read perms
int rperm(struct stat *f_stat, struct u_info *u_info) {
  // if optu not set or root, ignore
  if(!u_info->optu || !u_info->uid)
    return 1;

  // replicate unix perm checking; u->g->o
  if(f_stat->st_uid == u_info->uid)
    return f_stat->st_mode & S_IRUSR;
  for(int i = 0; i < u_info->ngroups; i++)
    if(u_info->groups[i] == f_stat->st_gid)
      return f_stat->st_mode & S_IRGRP;
  if(f_stat->st_mode & S_IROTH)
    return 1;
```

```
  return 0;
}

// print directory entry; handles decoding of stat info
void print_dirent(struct stat *f_stat, char *f_path) {
  // maximum path length in ext4: https://unix.stackexchange.com/a/32834/307410
  // maximum username length: https://serverfault.com/a/294122/332775
  char f_uname[33], f_gname[32], f_size[32], rl_buf[4097], f_link[4101],
       f_mtime[16], f_mode[11];
  struct passwd *f_user;
  struct group *f_group;
  int rl_len, f_blocks;

  // decode mode
  // get file type
  switch(f_stat->st_mode & S_IFMT) {
    case S_IFBLK: *f_mode = 'b'; break;
    case S_IFCHR: *f_mode = 'c'; break;
    case S_IFDIR: *f_mode = 'd'; break;
    case S_IFIFO: *f_mode = 'p'; break;
    case S_IFLNK: *f_mode = 'l'; break;
    case S_IFSOCK:*f_mode = 's'; break;
    case S_IFREG: *f_mode = '-'; break;
    default:      *f_mode = '?';
  }

  // permission bits
  f_mode[1] = f_stat->st_mode & S_IRUSR ? 'r' : '-';
  f_mode[2] = f_stat->st_mode & S_IWUSR ? 'w' : '-';
  f_mode[3] = f_stat->st_mode & S_IXUSR
            ? f_stat->st_mode & S_ISUID ? 's' : 'x'
            : f_stat->st_mode & S_ISUID ? 'S' : '-';
  f_mode[4] = f_stat->st_mode & S_IRGRP ? 'r' : '-';
  f_mode[5] = f_stat->st_mode & S_IWGRP ? 'w' : '-';
  f_mode[6] = f_stat->st_mode & S_IXGRP
            ? f_stat->st_mode & S_ISGID ? 's' : 'x'
            : f_stat->st_mode & S_ISGID ? 'S' : '-';
  f_mode[7] = f_stat->st_mode & S_IROTH ? 'r' : '-';
  f_mode[8] = f_stat->st_mode & S_IWOTH ? 'w' : '-';
  f_mode[9] = f_stat->st_mode & S_IXOTH
            ? f_stat->st_mode & S_ISVTX ? 't' : 'x'
            : f_stat->st_mode & S_ISVTX ? 'T' : '-';
  f_mode[10] = '\0';

  // get user info
  if(f_user = getpwuid(f_stat->st_uid))
    strcpy(f_uname, f_user->pw_name);
  else
    sprintf(f_uname, "%d", f_stat->st_uid);

  // get group info
  if(f_group = getgrgid(f_stat->st_gid))
    strcpy(f_gname, f_group->gr_name);
  else
    sprintf(f_gname, "%d", f_stat->st_gid);

  // if not char/block dev, get size in bytes
```

```
  // else get dev major, minor number
  if(!S_ISBLK(f_stat->st_mode) && !S_ISCHR(f_stat->st_mode))
    sprintf(f_size, "%ld", f_stat->st_size);
  else
    sprintf(f_size, "%d,%d", major(f_stat->st_rdev), minor(f_stat->st_rdev));

  // format date; if within the last year, do date/time (mon day time)
  // if longer do date/year (mon day year)
  // (1 year = 365*24*60*60 seconds = 31536000s)
  if(time(NULL) - f_stat->st_mtime < 31536000)
    strftime(f_mtime, 16, "%b %e %H:%M", localtime(&f_stat->st_mtime));
  else
    strftime(f_mtime, 16, "%b %e  %Y", localtime(&f_stat->st_mtime));

  // if symlink, get what it points to; otherwise, clear string
  if(S_ISLNK(f_stat->st_mode)) {
    if((rl_len = readlink(f_path, rl_buf, sizeof rl_buf - 1)) > 0)
      rl_buf[rl_len] = '\0';
    else
      ERROR_NON_FATAL("readlink", f_path, strerror(errno));

    sprintf(f_link, " -> %s", rl_buf);
  } else
    *f_link = '\0';

  // get number of blocks; contribution by Dave Kwong in the case
  // that fs has weird blocksize (adds 1 if not even multiple of 512 bytes)
  f_blocks = f_stat->st_blocks/2 + f_stat->st_blocks%2;

  fprintf(stdout, " %8ld %6d %s %3d %-8s %-8s %8s %s %s%s\n",
          f_stat->st_ino, f_blocks, f_mode, f_stat->st_nlink,
          f_uname, f_gname, f_size, f_mtime, f_path, f_link);
  return;
};

// recursively walk fs and print; handles tree walking and deciding
// which inode entries should be printed
void rls(char *basedir, int optm, int optv,
         dev_t start_vol, struct u_info *u_info, time_t stime) {
  DIR *dir;
  struct dirent *dirent;
  // max path length is 4096: https://unix.stackexchange.com/a/32834/307410
  char f_path[4097];
  struct stat f_stat;

  if(!(dir = opendir(basedir)))
    // if no access to directory, exit here
    if(errno == EACCES) {
      ERROR_NON_FATAL("opening directory", basedir, strerror(errno));
      return;
    } else
      ERROR_FATAL("opening directory", basedir, strerror(errno));

  // this happens on the first call to rls (by main() driver)
  if(start_vol == -1) {
    if(stat(basedir, &f_stat) < 0)
      ERROR_FATAL("stat directory", basedir, strerror(errno));
```

```
    start_vol = f_stat.st_dev;

    // print if applicable
    if(should_print())
      print_dirent(&f_stat, basedir);
  }

  errno = 0;
  while(dirent = readdir(dir)) {
    // ignore ., ..
    if(!strcmp(dirent->d_name, ".") || !strcmp(dirent->d_name, ".."))
      continue;

    // stat inode
    sprintf(f_path, "%s%s%s", basedir,
            basedir[strlen(basedir)-1] == '/' ? "" : "/", dirent->d_name);
    if(lstat(f_path, &f_stat) < 0) {
      ERROR_NON_FATAL("stat", basedir, strerror(errno));
      if(errno == EACCES)
        return;
    }

    // print if applicable
    if(should_print())
      print_dirent(&f_stat, f_path);
    errno = 0;

    // if directory, recursively print filenames
    if(S_ISDIR(f_stat.st_mode))
      if(optv && f_stat.st_dev != start_vol) {
        ERROR_NON_FATAL("mount point", f_path, "Not crossing mount point");
      } else
        rls(f_path, optm, optv, start_vol, u_info, stime);
    errno = 0;
  }
  // error reading directory in above loop
  if(errno)
    ERROR_NON_FATAL("reading directory", basedir, strerror(errno));

  // close dir
  if(closedir(dir) < 0)
    ERROR_FATAL("closing directory", basedir, strerror(errno));

  return;
}

// driver for rls function
int main(int argc, char **argv) {
  int optv = 0, optm = 0, opt;
  long optu = -1;
  char *startdir = "./";
  struct passwd *optu_passwd;
  struct u_info u_info = { .optu = 0 };
  time_t stime;

  // parse args
  while((opt = getopt(argc, argv, "m:u:v")) != -1) {
```

```
    switch(opt) {
      case 'm':
        // silently fail if invalid number; atoi will return 0 if invalid
        optm = atoi(optarg);
        break;
      case 'u':
        errno = 0;
        if((isdigit(*optarg) && (optu_passwd = getpwuid(atoi(optarg))))
          || (!isdigit(*optarg) && (optu_passwd = getpwnam(optarg)))) {
          // get grouplist; once to get ngroups, second time to retrieve gl
          u_info.ngroups = 0;
          getgrouplist(optu_passwd->pw_name, optu_passwd->pw_gid,
                       NULL, &u_info.ngroups);
          u_info.groups = (gid_t *) malloc(u_info.ngroups * sizeof(gid_t *));
          getgrouplist(optu_passwd->pw_name, optu_passwd->pw_gid,
                       u_info.groups, &u_info.ngroups);
          u_info.optu = 1;
          u_info.uid = optu_passwd->pw_uid;
        }
        if(errno)
          ERROR_FATAL("(getpwuid/getpwnam) processing parameter -u", optarg,
                      strerror(errno));

        // user not found; non-fatal error
        if(!optu_passwd)
          ERROR_NON_FATAL("processing parameter -u", optarg, "User not found");

        break;
      case 'v':
        optv = 1;
        break;
    }
  }

  // iterate thu args if given; else rls on cwd
  stime = time(NULL);
  if(optind == argc)
    rls(".", optm, optv, -1, &u_info, stime);
  else
    while(optind++ != argc)
      rls(argv[optind-1], optm, optv, -1, &u_info, stime);

  // free dynamically allocated memory and exit
  if(u_info.optu)
    free(u_info.groups);
  exit(EXIT_SUCCESS);
}
/**
  * Notes about this program:
  * - Formatting (padding) is not dynamic, but not the point of this project.
  * - Repeated -m and -u opts overwrite one another -- again, not the point
  *   of this project and could be implemented with more time.
  * - Unlike find, this doesn't escape some chars (e.g., backslash and space),
  *   so those show up on the diff.
  * - Some of the output messages were changed a little to be more consistent
  *   and similar to the output of find.
  */
```

EXAMPLE OUTPUT

```
(base) [jon@archijon testfs]$ # generating some directory structure
(base) [jon@archijon testfs]$ # demonstrate symlinks, hard links (same inode #s)
(base) [jon@archijon testfs]$ whoami
jon
(base) [jon@archijon testfs]$ mkdir dir1 dir2 dir3
(base) [jon@archijon testfs]$ touch dir1/f1 dir1/f2 dir2/f3
(base) [jon@archijon testfs]$ ln -s dir1/f1 dir3/f6
(base) [jon@archijon testfs]$ ln dir1/f2 dir3/f7
(base) [jon@archijon testfs]$
(base) [jon@archijon testfs]$ # show rls on some directories, invalid dir
(base) [jon@archijon testfs]$ find -ls
  5140990      4 drwxr-xr-x   5 jon      jon          4096 Oct  2 15:31 .
  6034871      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir2
  6034876      0 -rw-r--r--   1 jon      jon             0 Oct  2 15:31 ./dir2/f3
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir1
  6034875      0 -rw-r--r--   2 jon      jon             0 Oct  2 15:31 ./dir1/f2
  6034874      0 -rw-r--r--   1 jon      jon             0 Oct  2 15:31 ./dir1/f1
  6034873      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir3
  6034875      0 -rw-r--r--   2 jon      jon             0 Oct  2 15:31 ./dir3/f7
  6034877      0 lrwxrwxrwx   1 jon      jon             7 Oct  2 15:31 ./dir3/f6 -> dir1/f1
  5140991     24 -rwxr-xr-x   1 jon      jon         22512 Oct  2 15:31 ./rls
(base) [jon@archijon testfs]$ ./rls
  5140990      4 drwxr-xr-x   5 jon      jon          4096 Oct  2 15:31 .
  6034871      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir2
  6034876      0 -rw-r--r--   1 jon      jon             0 Oct  2 15:31 ./dir2/f3
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir1
  6034875      0 -rw-r--r--   2 jon      jon             0 Oct  2 15:31 ./dir1/f2
  6034874      0 -rw-r--r--   1 jon      jon             0 Oct  2 15:31 ./dir1/f1
  6034873      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir3
  6034875      0 -rw-r--r--   2 jon      jon             0 Oct  2 15:31 ./dir3/f7
  6034877      0 lrwxrwxrwx   1 jon      jon             7 Oct  2 15:31 ./dir3/f6 -> dir1/f1
  5140991     24 -rwxr-xr-x   1 jon      jon         22512 Oct  2 15:31 ./rls
(base) [jon@archijon testfs]$ ./rls .. | tail -n 5
  6034877      0 lrwxrwxrwx   1 jon      jon             7 Oct  2 15:31 ../testfs/dir3/f6 -> dir1/f1
  5140991     24 -rwxr-xr-x   1 jon      jon         22512 Oct  2 15:31 ../testfs/rls
  1979747      0 lrwxrwxrwx   1 jon      jon             4 Sep 27 22:14 ../test2 -> test
  1978833     24 -rwxr-xr-x   1 jon      jon         22512 Oct  2 15:26 ../rls
  1979582     44 -rw-r--r--   1 jon      jon         41979 Sep 30 22:28 ../t1
(base) [jon@archijon testfs]$ ./rls dir3/../../testfs/dir2
  6034871      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 dir3/../../testfs/dir2
  6034876      0 -rw-r--r--   1 jon      jon             0 Oct  2 15:31 dir3/../../testfs/dir2/f3
```

```
(base) [jon@archijon testfs]$ ./rls mkldmd
rls: ERROR: opening directory "mkldmd": No such file or directory
(base) [jon@archijon testfs]$
(base) [jon@archijon testfs]$ # demonstrate permissions
(base) [jon@archijon testfs]$ mkdir perms perms/forbidden perms/notraverse && cd perms
(base) [jon@archijon perms]$ touch 0777 0744 0000 1050 7000 4000 1000 7011 forbidden/test notraverse/test
(base) [jon@archijon perms]$ chmod 0777 0777
(base) [jon@archijon perms]$ chmod 0744 0744
(base) [jon@archijon perms]$ chmod 0000 0000
(base) [jon@archijon perms]$ chmod 1050 1050
(base) [jon@archijon perms]$ chmod 7000 7000
(base) [jon@archijon perms]$ chmod 4000 4000
(base) [jon@archijon perms]$ chmod 1000 1000
(base) [jon@archijon perms]$ chmod 7011 7011
(base) [jon@archijon perms]$ chmod 7666 forbidden
(base) [jon@archijon perms]$ chmod 7333 notraverse
(base) [jon@archijon perms]$ cd .. && ./rls perms
  6034878      4 drwxr-xr-x   4 jon      jon          4096 Oct  2 15:31 perms
  6034985      0 ---------T   1 jon      jon             0 Oct  2 15:31 perms/1000
  6034986      0 ---S--s--t   1 jon      jon             0 Oct  2 15:31 perms/7011
  6034883      0 ----------   1 jon      jon             0 Oct  2 15:31 perms/0000
  6034885      0 ---S--S--T   1 jon      jon             0 Oct  2 15:31 perms/7000
  6034879      4 drwSrwSrwT   2 jon      jon          4096 Oct  2 15:31 perms/forbidden
rls: stat "perms/forbidden": Permission denied
  6034881      0 -rwxrwxrwx   1 jon      jon             0 Oct  2 15:31 perms/0777
  6034880      4 d-ws-ws-wt   2 jon      jon          4096 Oct  2 15:31 perms/notraverse
rls: opening directory "perms/notraverse": Permission denied
  6034886      0 ---S------   1 jon      jon             0 Oct  2 15:31 perms/4000
  6034882      0 -rwxr--r--   1 jon      jon             0 Oct  2 15:31 perms/0744
  6034884      0 ----r-x--T   1 jon      jon             0 Oct  2 15:31 perms/1050
(base) [jon@archijon testfs]$
(base) [jon@archijon testfs]$ # demonstrate -v
(base) [jon@archijon testfs]$ mkisofs -o d1iso.iso dir1
Setting input-charset to 'UTF-8' from locale.
Total translation table size: 0
Total rockridge attributes bytes: 0
Total directory bytes: 0
Path table size(bytes): 10
Max brk space used 0
174 extents written (0 MB)
(base) [jon@archijon testfs]$ mkdir mnt
(base) [jon@archijon testfs]$ sudo mount -o loop d1iso.iso mnt
```

```
mount: /home/jon/Documents/coursework/ece357/hw/programs/testfs/mnt: WARNING: device write-protected, mounted read-only.
(base) [jon@archijon testfs]$ ./rls
  5140990      4 drwxr-xr-x   7 jon      jon          4096 Oct  2 15:31 .
  6034871      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir2
  6034876      0 -rw-r--r--   1 jon      jon             0 Oct  2 15:31 ./dir2/f3
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir1
  6034875      0 -rw-r--r--   2 jon      jon             0 Oct  2 15:31 ./dir1/f2
  6034874      0 -rw-r--r--   1 jon      jon             0 Oct  2 15:31 ./dir1/f1
     1472      2 dr-xr-xr-x   1 root     root         2048 Oct  2 15:31 ./mnt
     1474      0 -r-xr-xr-x   1 root     root            0 Oct  2 15:31 ./mnt/f1
     1475      0 -r-xr-xr-x   1 root     root            0 Oct  2 15:31 ./mnt/f2
  6034873      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir3
  6034875      0 -rw-r--r--   2 jon      jon             0 Oct  2 15:31 ./dir3/f7
  6034877      0 lrwxrwxrwx   1 jon      jon             7 Oct  2 15:31 ./dir3/f6 -> dir1/f1
  5140992    348 -rw-r--r--   1 jon      jon        356352 Oct  2 15:31 ./d1iso.iso
  6034878      4 drwxr-xr-x   4 jon      jon          4096 Oct  2 15:31 ./perms
  6034985      0 ---------T   1 jon      jon             0 Oct  2 15:31 ./perms/1000
  6034986      0 ---S--s--t   1 jon      jon             0 Oct  2 15:31 ./perms/7011
  6034883      0 ----------   1 jon      jon             0 Oct  2 15:31 ./perms/0000
  6034885      0 ---S--S--T   1 jon      jon             0 Oct  2 15:31 ./perms/7000
  6034879      4 drwSrwSrwT   2 jon      jon          4096 Oct  2 15:31 ./perms/forbidden
rls: stat "./perms/forbidden": Permission denied
  6034881      0 -rwxrwxrwx   1 jon      jon             0 Oct  2 15:31 ./perms/0777
  6034880      4 d-ws-ws-wt   2 jon      jon          4096 Oct  2 15:31 ./perms/notraverse
rls: opening directory "./perms/notraverse": Permission denied
  6034886      0 ---S------   1 jon      jon             0 Oct  2 15:31 ./perms/4000
  6034882      0 -rwxr--r--   1 jon      jon             0 Oct  2 15:31 ./perms/0744
  6034884      0 ----r-x--T   1 jon      jon             0 Oct  2 15:31 ./perms/1050
  5140991     24 -rwxr-xr-x   1 jon      jon         22512 Oct  2 15:31 ./rls
(base) [jon@archijon testfs]$ ./rls -v
  5140990      4 drwxr-xr-x   7 jon      jon          4096 Oct  2 15:31 .
  6034871      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir2
  6034876      0 -rw-r--r--   1 jon      jon             0 Oct  2 15:31 ./dir2/f3
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir1
  6034875      0 -rw-r--r--   2 jon      jon             0 Oct  2 15:31 ./dir1/f2
  6034874      0 -rw-r--r--   1 jon      jon             0 Oct  2 15:31 ./dir1/f1
     1472      2 dr-xr-xr-x   1 root     root         2048 Oct  2 15:31 ./mnt
rls: mount point "./mnt": Not crossing mount point
  6034873      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir3
  6034875      0 -rw-r--r--   2 jon      jon             0 Oct  2 15:31 ./dir3/f7
  6034877      0 lrwxrwxrwx   1 jon      jon             7 Oct  2 15:31 ./dir3/f6 -> dir1/f1
  5140992    348 -rw-r--r--   1 jon      jon        356352 Oct  2 15:31 ./d1iso.iso
```

```
  6034878      4 drwxr-xr-x   4 jon      jon          4096 Oct  2 15:31 ./perms
  6034985      0 ---------T   1 jon      jon             0 Oct  2 15:31 ./perms/1000
  6034986      0 ---S--s--t   1 jon      jon             0 Oct  2 15:31 ./perms/7011
  6034883      0 ----------   1 jon      jon             0 Oct  2 15:31 ./perms/0000
  6034885      0 ---S--S--T   1 jon      jon             0 Oct  2 15:31 ./perms/7000
  6034879      4 drwSrwSrwT   2 jon      jon          4096 Oct  2 15:31 ./perms/forbidden
rls: stat "./perms/forbidden": Permission denied
  6034881      0 -rwxrwxrwx   1 jon      jon             0 Oct  2 15:31 ./perms/0777
  6034880      4 d-ws-ws-wt   2 jon      jon          4096 Oct  2 15:31 ./perms/notraverse
rls: opening directory "./perms/notraverse": Permission denied
  6034886      0 ---S------   1 jon      jon             0 Oct  2 15:31 ./perms/4000
  6034882      0 -rwxr--r--   1 jon      jon             0 Oct  2 15:31 ./perms/0744
  6034884      0 ----r-x--T   1 jon      jon             0 Oct  2 15:31 ./perms/1050
  5140991     24 -rwxr-xr-x   1 jon      jon         22512 Oct  2 15:31 ./rls
(base) [jon@archijon testfs]$
(base) [jon@archijon testfs]$ # demonstrate -m
(base) [jon@archijon testfs]$ # I run this all as a script, so pretty much the created
(base) [jon@archijon testfs]$ # directory structure and processes should show up here
(base) [jon@archijon testfs]$ sleep 2
(base) [jon@archijon testfs]$ echo test > dir1/f1
(base) [jon@archijon testfs]$ sleep 2
(base) [jon@archijon testfs]$ echo test > dir1/f2
(base) [jon@archijon testfs]$ sleep 2
(base) [jon@archijon testfs]$ ./rls -m -3
  6034875      4 -rw-r--r--   2 jon      jon             5 Oct  2 15:31 ./dir1/f2
  6034875      4 -rw-r--r--   2 jon      jon             5 Oct  2 15:31 ./dir3/f7
rls: stat "./perms/forbidden": Permission denied
rls: opening directory "./perms/notraverse": Permission denied
(base) [jon@archijon testfs]$ ./rls -m -5
  6034875      4 -rw-r--r--   2 jon      jon             5 Oct  2 15:31 ./dir1/f2
  6034874      4 -rw-r--r--   1 jon      jon             5 Oct  2 15:31 ./dir1/f1
  6034875      4 -rw-r--r--   2 jon      jon             5 Oct  2 15:31 ./dir3/f7
rls: stat "./perms/forbidden": Permission denied
rls: opening directory "./perms/notraverse": Permission denied
(base) [jon@archijon testfs]$ ./rls -m 5
  5140990      4 drwxr-xr-x   7 jon      jon          4096 Oct  2 15:31 .
  6034871      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir2
  6034876      0 -rw-r--r--   1 jon      jon             0 Oct  2 15:31 ./dir2/f3
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir1
     1472      2 dr-xr-xr-x   1 root     root         2048 Oct  2 15:31 ./mnt
     1474      0 -r-xr-xr-x   1 root     root            0 Oct  2 15:31 ./mnt/f1
     1475      0 -r-xr-xr-x   1 root     root            0 Oct  2 15:31 ./mnt/f2
```

```
  6034873      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir3
  6034877      0 lrwxrwxrwx   1 jon      jon             7 Oct  2 15:31 ./dir3/f6 -> dir1/f1
  5140992    348 -rw-r--r--   1 jon      jon        356352 Oct  2 15:31 ./d1iso.iso
  6034878      4 drwxr-xr-x   4 jon      jon          4096 Oct  2 15:31 ./perms
  6034985      0 ---------T   1 jon      jon             0 Oct  2 15:31 ./perms/1000
  6034986      0 ---S--s--t   1 jon      jon             0 Oct  2 15:31 ./perms/7011
  6034883      0 ----------   1 jon      jon             0 Oct  2 15:31 ./perms/0000
  6034885      0 ---S--S--T   1 jon      jon             0 Oct  2 15:31 ./perms/7000
  6034879      4 drwSrwSrwT   2 jon      jon          4096 Oct  2 15:31 ./perms/forbidden
rls: stat "./perms/forbidden": Permission denied
  6034881      0 -rwxrwxrwx   1 jon      jon             0 Oct  2 15:31 ./perms/0777
  6034880      4 d-ws-ws-wt   2 jon      jon          4096 Oct  2 15:31 ./perms/notraverse
rls: opening directory "./perms/notraverse": Permission denied
  6034886      0 ---S------   1 jon      jon             0 Oct  2 15:31 ./perms/4000
  6034882      0 -rwxr--r--   1 jon      jon             0 Oct  2 15:31 ./perms/0744
  6034884      0 ----r-x--T   1 jon      jon             0 Oct  2 15:31 ./perms/1050
  5140991     24 -rwxr-xr-x   1 jon      jon         22512 Oct  2 15:31 ./rls
(base) [jon@archijon testfs]$ # show that invalid -m is ignored:
(base) [jon@archijon testfs]$ ./rls -m amdk
  5140990      4 drwxr-xr-x   7 jon      jon          4096 Oct  2 15:31 .
  6034871      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir2
  6034876      0 -rw-r--r--   1 jon      jon             0 Oct  2 15:31 ./dir2/f3
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir1
  6034875      4 -rw-r--r--   2 jon      jon             5 Oct  2 15:31 ./dir1/f2
  6034874      4 -rw-r--r--   1 jon      jon             5 Oct  2 15:31 ./dir1/f1
     1472      2 dr-xr-xr-x   1 root     root         2048 Oct  2 15:31 ./mnt
     1474      0 -r-xr-xr-x   1 root     root            0 Oct  2 15:31 ./mnt/f1
     1475      0 -r-xr-xr-x   1 root     root            0 Oct  2 15:31 ./mnt/f2
  6034873      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 ./dir3
  6034875      4 -rw-r--r--   2 jon      jon             5 Oct  2 15:31 ./dir3/f7
  6034877      0 lrwxrwxrwx   1 jon      jon             7 Oct  2 15:31 ./dir3/f6 -> dir1/f1
  5140992    348 -rw-r--r--   1 jon      jon        356352 Oct  2 15:31 ./d1iso.iso
  6034878      4 drwxr-xr-x   4 jon      jon          4096 Oct  2 15:31 ./perms
  6034985      0 ---------T   1 jon      jon             0 Oct  2 15:31 ./perms/1000
  6034986      0 ---S--s--t   1 jon      jon             0 Oct  2 15:31 ./perms/7011
  6034883      0 ----------   1 jon      jon             0 Oct  2 15:31 ./perms/0000
  6034885      0 ---S--S--T   1 jon      jon             0 Oct  2 15:31 ./perms/7000
  6034879      4 drwSrwSrwT   2 jon      jon          4096 Oct  2 15:31 ./perms/forbidden
rls: stat "./perms/forbidden": Permission denied
  6034881      0 -rwxrwxrwx   1 jon      jon             0 Oct  2 15:31 ./perms/0777
  6034880      4 d-ws-ws-wt   2 jon      jon          4096 Oct  2 15:31 ./perms/notraverse
rls: opening directory "./perms/notraverse": Permission denied
```

```
  6034886      0 ---S------   1 jon      jon             0 Oct  2 15:31 ./perms/4000
  6034882      0 -rwxr--r--   1 jon      jon             0 Oct  2 15:31 ./perms/0744
  6034884      0 ----r-x--T   1 jon      jon             0 Oct  2 15:31 ./perms/1050
  5140991     24 -rwxr-xr-x   1 jon      jon         22512 Oct  2 15:31 ./rls
(base) [jon@archijon testfs]$
(base) [jon@archijon testfs]$ # demonstrate -u
(base) [jon@archijon testfs]$ chmod 0244 dir1/f1
(base) [jon@archijon testfs]$ sudo chown alice dir1/f1
(base) [jon@archijon testfs]$ chmod 0040 dir1/f2
(base) [jon@archijon testfs]$ sudo chown bob dir1/f2
(base) [jon@archijon testfs]$ ./rls dir1
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 dir1
  6034875      4 ----r-----   2 bob      jon             5 Oct  2 15:31 dir1/f2
  6034874      4 --w-r--r--   1 alice    jon             5 Oct  2 15:31 dir1/f1
(base) [jon@archijon testfs]$ ./rls -u jon dir1
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 dir1
  6034875      4 ----r-----   2 bob      jon             5 Oct  2 15:31 dir1/f2
  6034874      4 --w-r--r--   1 alice    jon             5 Oct  2 15:31 dir1/f1
(base) [jon@archijon testfs]$ ./rls -u 1000 dir1
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 dir1
  6034875      4 ----r-----   2 bob      jon             5 Oct  2 15:31 dir1/f2
  6034874      4 --w-r--r--   1 alice    jon             5 Oct  2 15:31 dir1/f1
(base) [jon@archijon testfs]$ ./rls -u bob dir1
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 dir1
  6034874      4 --w-r--r--   1 alice    jon             5 Oct  2 15:31 dir1/f1
(base) [jon@archijon testfs]$ ./rls -u alice dir1
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 dir1
(base) [jon@archijon testfs]$ ./rls -u root dir1
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 dir1
  6034875      4 ----r-----   2 bob      jon             5 Oct  2 15:31 dir1/f2
  6034874      4 --w-r--r--   1 alice    jon             5 Oct  2 15:31 dir1/f1
(base) [jon@archijon testfs]$ # show invalid -u
(base) [jon@archijon testfs]$ ./rls -u askdmsad dir1
rls: processing parameter -u "askdmsad": User not found
  6034870      4 drwxr-xr-x   2 jon      jon          4096 Oct  2 15:31 dir1
  6034875      4 ----r-----   2 bob      jon             5 Oct  2 15:31 dir1/f2
  6034874      4 --w-r--r--   1 alice    jon             5 Oct  2 15:31 dir1/f1
(base) [jon@archijon testfs]$
(base) [jon@archijon testfs]$ # show that its output is similar to find -ls (except whitespace)
(base) [jon@archijon testfs]$ # show block, char devices
(base) [jon@archijon testfs]$ ./rls /dev | tail -n 20
     3094      0 crw--w----   1 jon      tty           4,2 Oct  2 14:09 /dev/tty2
```

```
      3093      0 crw-------   1 jon      tty           4,1 Oct  2 15:22 /dev/tty1
      3092      0 crw-rw----   1 root     tty         7,129 Oct  2 13:21 /dev/vcsa1
      3091      0 crw-rw----   1 root     tty          7,65 Oct  2 13:21 /dev/vcsu1
      3090      0 crw-rw----   1 root     tty           7,1 Oct  2 13:21 /dev/vcs1
      3089      0 crw-rw----   1 root     tty         7,128 Oct  2 13:21 /dev/vcsa
      3088      0 crw-rw----   1 root     tty          7,64 Oct  2 13:21 /dev/vcsu
      3087      0 crw-rw----   1 root     tty           7,0 Oct  2 13:21 /dev/vcs
      3086      0 crw--w----   1 root     tty           4,0 Oct  2 13:21 /dev/tty0
      3085      0 crw-------   1 root     root          5,1 Oct  2 13:21 /dev/console
      3084      0 crw-rw-rw-   1 root     tty           5,0 Oct  2 15:27 /dev/tty
      3083      0 crw-r--r--   1 root     root         1,11 Oct  2 13:21 /dev/kmsg
      3082      0 crw-rw-rw-   1 root     root          1,9 Oct  2 13:21 /dev/urandom
      3081      0 crw-rw-rw-   1 root     root          1,8 Oct  2 13:21 /dev/random
      3080      0 crw-rw-rw-   1 root     root          1,7 Oct  2 13:21 /dev/full
      3079      0 crw-rw-rw-   1 root     root          1,5 Oct  2 13:21 /dev/zero
      3078      0 crw-r-----   1 root     kmem          1,4 Oct  2 13:21 /dev/port
      3077      0 crw-rw-rw-   1 root     root          1,3 Oct  2 13:21 /dev/null
      3076      0 crw-r-----   1 root     kmem          1,1 Oct  2 13:21 /dev/mem
      2074      0 crw-------   1 root     root        10,63 Oct  2 13:21 /dev/vga_arbiter
(base) [jon@archijon testfs]$ diff -w <(./rls /dev) <(find /dev -ls)
(base) [jon@archijon testfs]$ # (since diff showed nothing, is same except whitespace)
```

PSET 3

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
10/23/19

Problem 1: Shell Script Invocation

1. The child of the interactive shell execs /bin/sh (from the shebang).

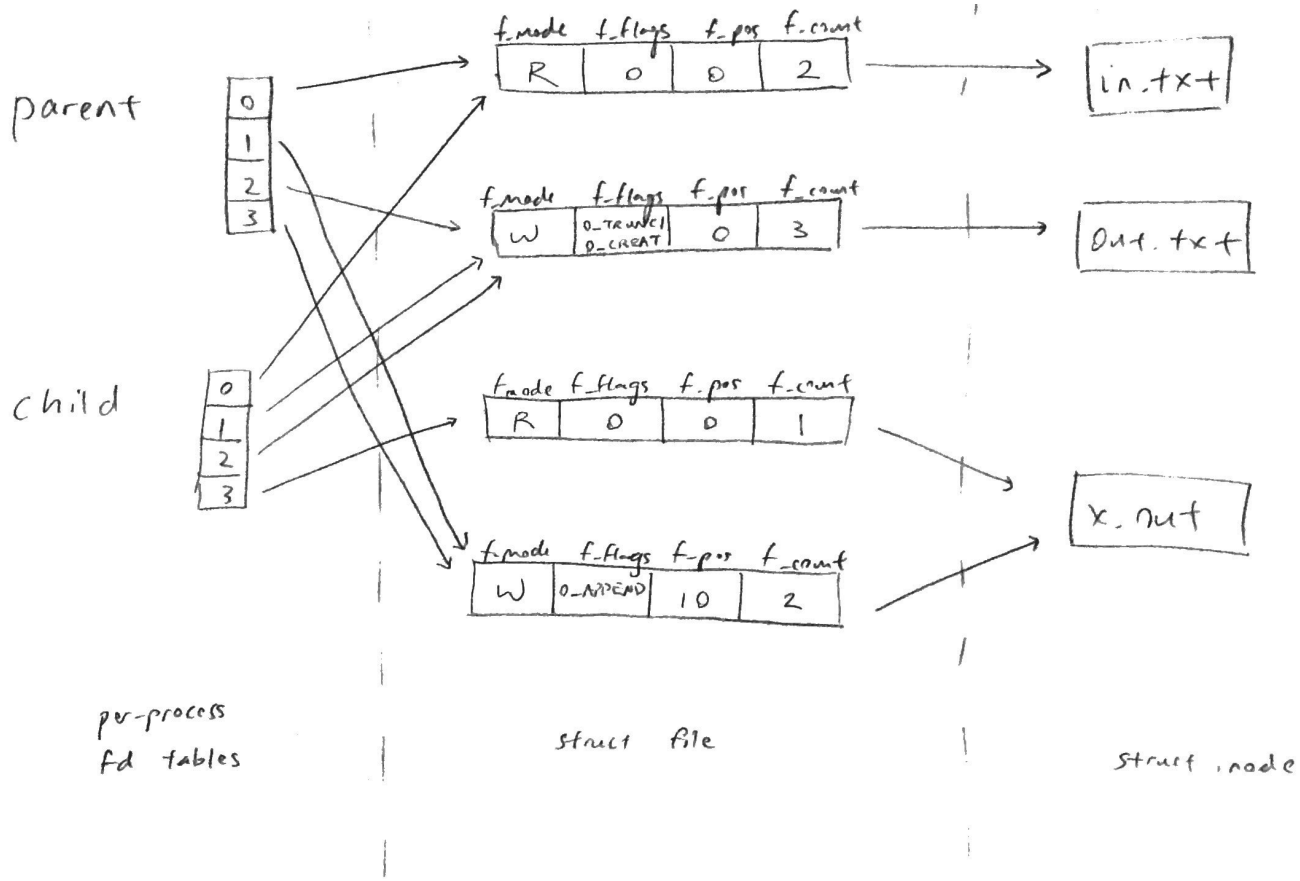
2. argc: 5
   argv: {“sh”, “./script.sh”, “f2.c”, “f3.c”, “f4.c”, NULL}

The sh shell expands “f[2-4].c” into separate arguments if the files exist, which they do. The argv is always NULL-terminated. It also seems that argv[1] is not the fully qualified path (as the lecture notes indicate), but the path of the script passed to the command.

3. Calls one of the wait syscalls to wait for the child to finish executing, and to get its return code.

4. Looking at man 1 ls, ls returns 1 for minor errors (such as not being able to access subdirs) and 2 for major errors (such as not being able to access the command-line argument). Since foobar is unreachable, this falls into the second category, so 2 is printed to the terminal.

Problem 2: File Descriptor Tables



Problem 3: Simple Shell Program

```
#include <ctype.h>
#include <errno.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/resource.h>
#include <sys/wait.h>
#include <unistd.h>

#define SHELL "jsh"
#define PROMPT SHELL "$ "

#define ERR_FAT(prog, op, ctx, msg) {\
    dprintf(2, "%s: ERROR: %s \"%s\": %s\n", prog, op, ctx, msg);\
    exit(EXIT_FAILURE);\
  }
#define WARN(prog, op, ctx, msg)\
  dprintf(2, "%s: %s \"%s\": %s\n", prog, op, ctx, msg)

// structure parsed line
struct rd_out {
  // flag: 1 for append, 0 for trunc
  int append;
  char *file;
};
struct cmd_parse {
  char *cmd, **argv, *rd_in;
  struct rd_out rd_out, rd_err;
  int argc;
};

// global last status
int laststatus = 0;

// manage i/o redirection in file
void io_rd(char *file, int flags, int fd, char *rd_stream) {
  int rd_fd;
  char warn[40];
  if((rd_fd = open(file, flags, 0666)) < 0)
    sprintf(warn, "i/o redirection of (open) to %s", rd_stream);
  else if(dup2(rd_fd, fd) < 0)
    sprintf(warn, "i/o redirection of (dup2) to %s", rd_stream);
  else if(close(rd_fd) < 0)
    sprintf(warn, "unclean fd environment from i/o redirection (close) to %s",
            rd_stream);
  else
    return;
  WARN(SHELL, warn, file, strerror(errno));
  exit(1);
}

// handling line parsing; cmd_src is the source of cmds (stdin or interpreter)
void parse_line(char *line, FILE *cmd_src) {
```

```
char *token, path_buf[4097];
int argc_cap = 4, rd_fd;
struct cmd_parse cmd_parse = {
  .cmd = NULL,
  .argv = (char **) malloc(argc_cap * sizeof(char *)),
  .rd_in = NULL,
  .rd_out = { 0, NULL },
  .rd_err = { 0, NULL },
  .argc = 1
};
struct rusage rusage;
pid_t cpid;
struct timeval cp_start, cp_end;
int wstatus;

// very basic comments: ignore lines starting with "#"
if(*line == '#')
  return;

// very basic tokenizing by whitespace
token = strtok(line, " \t\n");
if(!token)
  return;

cmd_parse.cmd = token;
cmd_parse.argv[0] = token;
while(token = strtok(NULL, " \t\n")) {
  if(*token == '>') {
    cmd_parse.rd_out.append = *(token+1) == '>';
    cmd_parse.rd_out.file = token+1+cmd_parse.rd_out.append;
  } else if(*token == '2' && *(token+1) == '>') {
    cmd_parse.rd_err.append = *(token+2) == '>';
    cmd_parse.rd_err.file = token+2+cmd_parse.rd_err.append;
  } else if(*token == '<') {
    cmd_parse.rd_in = token+1;
  } else {
    if(cmd_parse.argc == argc_cap)
      if(!(cmd_parse.argv = (char **)
           realloc(cmd_parse.argv, (argc_cap*=2) * sizeof(char *))))
        WARN(SHELL, "allocating memory for argument parsing (realloc)",
             token, strerror(errno));
    cmd_parse.argv[cmd_parse.argc++] = token;
  }
}
// terminate argv with np
if(cmd_parse.argc == argc_cap)
  if(!(cmd_parse.argv = (char **)
       realloc(cmd_parse.argv, (argc_cap+1) * sizeof(char *))))
    WARN(SHELL, "allocating memory for argument parsing (realloc)",
         "end token (NULL)", strerror(errno));
cmd_parse.argv[cmd_parse.argc] = NULL;

// shell built-ins
if(!strcmp(cmd_parse.cmd, "pwd")) {
  if(!getcwd(path_buf, 4097)) {
    WARN("pwd", "getcwd", "", strerror(errno));
    laststatus = errno;
```

```
  } else
    dprintf(1, "%s\n", path_buf);
} else if(!strcmp(cmd_parse.cmd, "cd")) {
  if(chdir(cmd_parse.argc == 1 ? getenv("HOME") : cmd_parse.argv[1]) < 0) {
    WARN("cd", "chdir", cmd_parse.argc == 1 ? "" : cmd_parse.argv[1],
         strerror(errno));
    laststatus = errno;
  }
} else if(!strcmp(cmd_parse.cmd, "exit")) {
  // if invalid error code, return 2
  // bash does this; see https://askubuntu.com/a/892605/433872
  if(cmd_parse.argc > 1)
    for(char *c = cmd_parse.argv[1]; *c; c++)
      if(!(isdigit(*c) || (*c == '-' && c == cmd_parse.argv[1]))) {
        WARN(SHELL, "exit", cmd_parse.argv[1], "Numeric argument required");
        exit(2);
      }
  exit(cmd_parse.argc > 1 ? atoi(cmd_parse.argv[1]) : laststatus);
}

// fork, exec other programs
else {
  gettimeofday(&cp_start, NULL);
  switch(cpid = fork()) {
    case -1:
      WARN(SHELL, cmd_parse.cmd, "fork", strerror(errno));
      break;

    // child
    case 0:
      // if not interactive mode, close interpreted script fd
      if(cmd_src != stdin) {
        if(fclose(cmd_int)) {
          WARN(SHELL, "closing script fd to initiate clean child fd env",
               "close", strerror(errno));
        }
      }

      // i/o redirection
      if(cmd_parse.rd_in)
        io_rd(cmd_parse.rd_in, O_RDONLY, 0, "standard input");
      if(cmd_parse.rd_out.file)
        io_rd(cmd_parse.rd_out.file,
              O_WRONLY|O_CREAT|(cmd_parse.rd_out.append?O_APPEND:O_TRUNC),
              1, "standard output");
      if(cmd_parse.rd_err.file)
        io_rd(cmd_parse.rd_err.file,
              O_WRONLY|O_CREAT|(cmd_parse.rd_err.append?O_APPEND:O_TRUNC),
              2, "standard error");

      // exec; if unsuccessful, following lines to report error
      execvp(cmd_parse.cmd, cmd_parse.argv);
      WARN(SHELL, "exec", cmd_parse.cmd, strerror(errno));
      exit(127);

    // parent
    default:
```

```
        wait4(cpid, &wstatus, 0, &rusage);
        // return status from exit (laststatus) is the return value if normally
        // exited, and the whole status value if terminated with signal
        // (same behavior as bash)
        laststatus = WIFSIGNALED(wstatus) ? wstatus : WEXITSTATUS(wstatus);
        gettimeofday(&cp_end, NULL);
        dprintf(2, "%s: Child process %d exited ", SHELL, cpid);
        if(wstatus && !WIFSIGNALED(wstatus))
          dprintf(2, "with return value %d\n", WEXITSTATUS(wstatus));
        else if(wstatus)
          dprintf(2, "with signal %d (%s)\n",
                  WTERMSIG(wstatus), strsignal(WTERMSIG(wstatus)));
        else
          dprintf(2, "normally\n");
        dprintf(2, "%s: Real: %fs User: %fs Sys: %fs\n",
                SHELL,
                cp_end.tv_sec-cp_start.tv_sec+(cp_end.tv_usec-cp_start.tv_usec)
                  /1e6,
                rusage.ru_utime.tv_sec+rusage.ru_utime.tv_usec/1e6,
                rusage.ru_stime.tv_sec+rusage.ru_stime.tv_usec/1e6);
        free(cmd_parse.argv);
    }
  }
}

// driver function: handle interpreted scripts and start parse loop
int main(int argc, char **argv) {
  char *line_buf = NULL;
  size_t line_len = 0;
  FILE *cmd_in = stdin;

  // open command inputs as fd 3; will be closed to children after forking
  // expects first argument to be a fname since no other args are defined;
  // this is the same behavior as bash for non-option arguments
  if(argc > 1) {
    if(!(cmd_in = fopen(argv[1], "r")))
      ERR_FAT(SHELL, "Opening interpreter file", argv[1], strerror(errno));
  }

  // read, parse, execute command
  // prints out prompt if not reading from script file
  errno = 0;
  while(cmd_in == stdin && dprintf(1, PROMPT),
        getline(&line_buf, &line_len, cmd_in) != -1)
    parse_line(line_buf, cmd_in);
  if(errno)
    ERR_FAT(SHELL, "Reading line (getline)", line_buf, strerror(errno));

  // cleanup and exit
  // this will only be called if EOF from interpreter
  free(line_buf);
  dprintf(1, "\nEOF read, exiting shell with exit code %d\n", laststatus);
  exit(EXIT_SUCCESS);
}
```

Test cases

```
(base) [jon@archijon prog3]$ ./jsh
jsh$ # testing builtins
jsh$ cd
jsh$ pwd
/home/jon
jsh$ cd Documents/coursework/ece^C
(base) [jon@archijon prog3]$ ./jsh
jsh$ # testing builtins
jsh$ pwd
/home/jon/Documents/coursework/ece357/hw/programs/prog3
jsh$ cd
jsh$ pwd
/home/jon
jsh$ cd Documents/coursework/ece357/hw/programs/prog3/././/////.
jsh$ pwd
/home/jon/Documents/coursework/ece357/hw/programs/prog3
jsh$ exit 52
(base) [jon@archijon prog3]$ echo $?
52
(base) [jon@archijon prog3]$ ./jsh
jsh$ # test redirection
jsh$ ls -l >ls.out
jsh: Child process 8009 exited normally
jsh: Real: 0.014720s User: 0.013427s Sys: 0.000000s
jsh$ cat ls.out
total 68
-rw-r--r-- 1 jon jon    36 Oct 23 13:50 input.txt
-rwxr-xr-x 1 jon jon 17992 Oct 23 13:41 jsh
-rw-r--r-- 1 jon jon  7326 Oct 23 13:41 jsh.c
-rw-r--r-- 1 jon jon     0 Oct 23 14:30 ls.out
-rwxr-xr-x 1 jon jon 16536 Oct 12 02:24 sigsegv
-rw-r--r-- 1 jon jon    54 Oct 12 02:20 sigsegv.c
-rwxr-xr-x 1 jon jon    27 Oct 23 13:49 test2.jsh
-rw-r--r-- 1 jon jon   920 Oct 23 14:30 testcommands
-rwxr-xr-x 1 jon jon    42 Oct 23 13:48 testme.jsh
jsh: Child process 8010 exited normally
jsh: Real: 0.003175s User: 0.002788s Sys: 0.000000s
jsh$ ../prog2/rls / -v 2>rls.err >rls.out
jsh: Child process 8013 exited normally
jsh: Real: 33.907177s User: 8.273273s Sys: 12.156793s
jsh$ tail rls.err
rls: opening directory "/opt/anaconda/.cph_tmpxk03b1af": Permission denied
rls: opening directory "/opt/anaconda/.cph_tmpwh81ttt3": Permission denied
rls: opening directory "/opt/anaconda/.cph_tmpsdh_1e38": Permission denied
rls: opening directory "/opt/anaconda/.cph_tmp5p78hp0x": Permission denied
rls: opening directory "/opt/anaconda/.cph_tmpwvqtbno8": Permission denied
rls: opening directory "/opt/anaconda/.cph_tmpmnzy858x": Permission denied
rls: opening directory "/opt/anaconda/.cph_tmpxpla5b29": Permission denied
rls: opening directory "/opt/anaconda/.cph_tmpeb0kxuah": Permission denied
rls: mount point "/dev": Not crossing mount point
rls: opening directory
"/home/jon/Documents/coursework/ece357/hw/playground/testdir": Permission denied
jsh: Child process 8032 exited normally
jsh: Real: 0.002997s User: 0.000000s Sys: 0.002584s
jsh$ tail rls.out
```

```
 13406311    293 -rw-r--r--   1 jon      jon        296213 Oct 13 18:25
/home/jon/Downloads/diff guide.pdf
 13413185    309 -rw-r--r--   1 jon      jon        314946 Oct  5 13:05
/home/jon/Downloads/Studio Ghibli Medley (Animenz)_v2 (1).pdf
 13404302    133 -rw-r--r--   1 jon      jon        134675 Sep 16 10:11
/home/jon/Downloads/lr_template (1).pdf
 13418131    613 -rw-r--r--   1 jon      jon        624375 Sep 25 12:45
/home/jon/Downloads/richard-bib-out.txt
  6034852      5 drwxr-xr-x   2 root     root         4096 Oct  2 14:21 /home/alice
  2621441      5 drwxr-xr-x   4 root     root         4096 Jul 30 08:50 /srv
  2621443      5 drwxr-xr-x   2 root     root         4096 Oct  6 13:06 /srv/http
  2621442      5 dr-xr-xr-x   2 root     ftp          4096 May 23 10:18 /srv/ftp
       14      0 lrwxrwxrwx   1 root     root            7 Oct  6 11:44 /lib64 ->
usr/lib
       15      0 lrwxrwxrwx   1 root     root            7 Oct  6 11:44 /sbin ->
usr/bin
jsh: Child process 8035 exited normally
jsh: Real: 0.002904s User: 0.002589s Sys: 0.000000s
jsh$ echo helloworld >>rls.out
jsh: Child process 8038 exited normally
jsh: Real: 0.002749s User: 0.002341s Sys: 0.000000s
jsh$ tail rls.out
 13413185    309 -rw-r--r--   1 jon      jon        314946 Oct  5 13:05
/home/jon/Downloads/Studio Ghibli Medley (Animenz)_v2 (1).pdf
 13404302    133 -rw-r--r--   1 jon      jon        134675 Sep 16 10:11
/home/jon/Downloads/lr_template (1).pdf
 13418131    613 -rw-r--r--   1 jon      jon        624375 Sep 25 12:45
/home/jon/Downloads/richard-bib-out.txt
  6034852      5 drwxr-xr-x   2 root     root         4096 Oct  2 14:21 /home/alice
  2621441      5 drwxr-xr-x   4 root     root         4096 Jul 30 08:50 /srv
  2621443      5 drwxr-xr-x   2 root     root         4096 Oct  6 13:06 /srv/http
  2621442      5 dr-xr-xr-x   2 root     ftp          4096 May 23 10:18 /srv/ftp
       14      0 lrwxrwxrwx   1 root     root            7 Oct  6 11:44 /lib64 ->
usr/lib
       15      0 lrwxrwxrwx   1 root     root            7 Oct  6 11:44 /sbin ->
usr/bin
helloworld
jsh: Child process 8041 exited normally
jsh: Real: 0.002795s User: 0.002467s Sys: 0.000000s
jsh$
jsh$ # test error checking
jsh$ ls amskdlmaskdm
ls: cannot access 'amskdlmaskdm': No such file or directory
jsh: Child process 8043 exited with return value 2
jsh: Real: 0.003643s User: 0.000000s Sys: 0.003322s
jsh$ cd askdlmasldm
cd: chdir "askdlmasldm": No such file or directory
jsh$ ./sigsegv
jsh: Child process 8050 exited with signal 11 (Segmentation fault)
jsh: Real: 0.220205s User: 0.000000s Sys: 0.001683s
jsh$
EOF read, exiting shell with exit code 139
(base) [jon@archijon prog3]$ echo $?
139
(base) [jon@archijon prog3]$ cat testme.jsh
#!./jsh
cat >cat.out
```

```
cat cat.out
exit 123
(base) [jon@archijon prog3]$ ./testme.jsh
hello, world!
here is some text
1
1 1
1 1 1
1 1 1 1 1
1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1
jsh: Child process 8288 exited normally
jsh: Real: 23.773219s User: 0.000000s Sys: 0.003095s
hello, world!
here is some text
1
1 1
1 1 1
1 1 1 1 1
1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1
jsh: Child process 8298 exited normally
jsh: Real: 0.002143s User: 0.001872s Sys: 0.000000s
(base) [jon@archijon prog3]$ echo $?
123
(base) [jon@archijon prog3]$ cat test2.jsh
#!./jsh
cat >cat2.out
exit
(base) [jon@archijon prog3]$ cat input.txt
Hello, world!
This is in input.txt!
(base) [jon@archijon prog3]$ ./test2.jsh <input.txt
jsh: Child process 8316 exited normally
jsh: Real: 0.004048s User: 0.002154s Sys: 0.000000s
(base) [jon@archijon prog3]$ echo $?
0
(base) [jon@archijon prog3]$ cat cat2.out
Hello, world!
This is in input.txt!
```

PSET 4

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
10/30/19

Problem 1 - Signal Numbers & Behavior

The answers below apply to Linux 5.3.7 x86_64.

a) Which signal is generated for all foreground processes attached to a terminal session when that terminal receives a Ctrl+Z character?

SIGTSTP (i.e., signal number 20)

b) If we wanted to terminate a program running the foreground AND cause it to dump core (if possible), what key sequence do we use?

Ctrl+\

(This is SIGQUIT, found using stty -a.)

c) What UNIX command would I use to make Ctrl+I the character which causes SIGINT to be sent?

stty intr ^I

d) I send a certain process signal #60 on a Linux system, using the kill syscall, 32 times in a row. At the time, that process has signal #60 in its blocked signals mask, and has a handler function established. After some period of time has elapsed, the target process unblocks signal #60.

di) How? Include a code snippet to unblock signal #60.

```
sigset_t set;
sigemptyset(&set);
sigaddset(&set, 60);
sigprocmask(SIG_UNBLOCK, &set, NULL);
```

dii) How many times does the handler function run? If instead of signal #60, we had been talking about signal #2, how many times would the handler function run?

Signal 60 is one of the real-time signals, so it would queue. Therefore, we can expect the handler function to be called 32 times.

However, the a standard signal like signal 2 (SIGINT) doesn't queue, so it would only be called once (on top of the fact that it would exit the program if its handler was not changed from the default terminate action). This is because the pending signals mask is only a bitmap, and doesn't hold any additional information about a signal (e.g., quantity) except that that one or more is pending.

Problem 2 - Interrupted and Restarted System Calls

a) You should see that the “pipe short write” condition comes up, not necessarily with each and every write. Why is this happening?

> This message is printed only if there is the SIGUSR1 signal sent to the process after write has written some bytes to the pipe but has not finished writing (returned). If the write syscall is interrupted before it has written any bytes, or if the write syscall is not interrupted by a SIGUSR1 signal at all, then the message is not written. It’s erratic because of the asynchronity of the message sending and the read/write/sleep cycles of the child processes.

b) Do you notice any pattern to the integer reported with the “pipe short write” messages? Explain?

> All of the integers are integer multiples of 4096, greater than 0 and less than 65536. This makes sense, since write is buffered to 4K for files (including pipes, which act like files).

c) What happens if you change sa_flags to 0? Why?

> This would remove the SA_RESTART flag, which causes an automatic restart if a signal was received before any data was written to the pipe (whereas SA_RESTART causes write to silently fail and “retry” if a signal is received before writing any data, since there shouldn’t be any negative effects).
>
> This change causes write to return -1 and set errno to EINTR if 0 bytes have been written, which causes “EINTR” to be written to stderr.

d) Why signal(SIGCHILD,SIG_IGN)? What happens if not?

> This causes the program to never terminate. This is because by default, a child creates a zombie after termination. However, the signal(SIGCHILD,SIG_IGN) causes no zombie to be created in Linux. The only way for the parent to exit is when the kill syscall fails. If the zombie doesn’t exist (as in the original program code), then the kill syscall fails; however, if the zombie does exist, then the kill continues to send the signal to the zombie without error (as the manpage says that a zombie that has not been recalled with the wait syscall is a valid existing process), so the parent never terminates.

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
11/6/19

```
// catgrepless.c -- uses less b/c less is more
// (and b/c less always goes into int. mode and doesn't "skip" short files)

#include <errno.h>
#include <fcntl.h>
#include <setjmp.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>

#define BUF_SIZE 4096

#define ERR_FAT(prog, op, ctx, msg) {\
    dprintf(2, "%s: ERROR: %s \"%s\": %s\n", prog, op, ctx, msg);\
    exit(EXIT_FAILURE);\
  }

// handle broken pipe (SIGPIPE) in grep process (when more exits first);
// when encountered, write debug data and end this process
jmp_buf jmp_env;
void sigpipe_hand(int sig) { siglongjmp(jmp_env, 1); }
void sigint_hand(int sig) { siglongjmp(jmp_env, 2); }

int main(int argc, char **argv) {

  char buf[BUF_SIZE], *pattern, **infile, **gargv, **margv;
  int gpid, mpid, wstatus, ftg[2], gtm[2], // file-to-grep, grep-to-more pipes
      rlen, wlen, wtot, ifd, ofd, bp, jstatus; // jump status

  // assume input of "catgrepmore pattern infile1 [...infile2...]"
  if(argc < 3)
    ERR_FAT(argv[0], "parsing arguments", "[too few arguments]",
            "Usage:\n\tcatgrepmore pattern infile1 [...infile2...]");
  pattern = argv[1];
  infile = argv+2;

  // sets up args for execing children
  // actually uses less instead of more
  gargv = (char *[3]) {"grep", pattern, NULL};
  margv = (char *[2]) {"less", NULL};

  // initialize bytes/files processed, ifd/ofd to -1 to indicate not open
```

```
bp = 0, ifd = ofd = -1;

// handle SIGPIPE, SIGINT: should only be called after loop below begins
// SIGPIPE may occur if more exists before grep has finished writing; for
// this, simply continue to the next file. For SIGINT, do the same but also
// print out number of bytes/files processed. Also make sure all ifd/ofd
// are closed (if opened).
if((jstatus = sigsetjmp(jmp_env, 1)) != 0) {
  // make sure children are dead; this is not necessary for more, but less
  // persists for SIGINT
  if(kill(mpid, SIGTERM) < 0)
    ERR_FAT(argv[0], "kill: more after signal", *infile, strerror(errno));
  waitpid(mpid, &wstatus, 0);
  waitpid(gpid, &wstatus, 0);

  // make sure fds closed (if opened)
  if(ifd != -1 && close(ifd) < 0)
    ERR_FAT(argv[0], "close: input file", *infile, strerror(errno));
  if(ofd != -1 && close(ofd) < 0)
    ERR_FAT(argv[0], "close: output pipe from parent for input file",
            *infile, strerror(errno));

  // print message if Ctrl+C
  if(jstatus == 2)
    dprintf(2, "Warning: SIGINT encountered. %d bytes/%d files processed.\n",
            bp, infile-argv-1);
  infile++;
}
signal(SIGINT, sigint_hand);
signal(SIGPIPE, sigpipe_hand);

for(; *infile; infile++) {

  // pipe 1 is from this program to grep; pipe 2 is from grep to more
  // pipe 2 is created first because to involve fewer file closings
  if(pipe(gtm) < 0)
    ERR_FAT(argv[0], "pipe: creating pipe 2", *infile, strerror(errno));
  switch(mpid = fork()) {
    case -1:
      ERR_FAT(argv[0], "fork: to more child", *infile, strerror(errno));
      break;
    case 0:
      signal(SIGINT, SIG_DFL);
      signal(SIGPIPE, SIG_DFL);

      if(dup2(gtm[0], 0) < 0)
        ERR_FAT(argv[0], "dup2: pipe 2 read to more child in", *infile,
                strerror(errno));
      if(close(gtm[0]) < 0)
        ERR_FAT(argv[0], "close: pipe 2 read in more child", *infile,
                strerror(errno));
```

```
    // this is not used, close
    if(close(gtm[1]) < 0)
      ERR_FAT(argv[0], "close: pipe 2 write in more child", *infile,
              strerror(errno));

    if(execvp(*margv, margv) < 0)
      ERR_FAT(argv[0], "execvp: more", *infile, strerror(errno));
  default:
    if(close(gtm[0]) < 0)
      ERR_FAT(argv[0], "close: pipe 2 read in parent", *infile,
              strerror(errno));
}

if(pipe(ftg) < 0)
  ERR_FAT(argv[0], "pipe: creating pipe 1", *infile, strerror(errno));
ofd = ftg[1];
switch(gpid = fork()) {
  case -1:
    ERR_FAT(argv[0], "fork: to grep child", *infile, strerror(errno));
    break;
  case 0:
    signal(SIGINT, SIG_DFL);
    signal(SIGPIPE, SIG_DFL);

    if(dup2(ftg[0], 0) < 0)
      ERR_FAT(argv[0], "dup2: pipe 1 read to grep child in", *infile,
              strerror(errno));
    if(dup2(gtm[1], 1) < 0)
      ERR_FAT(argv[0], "dup2: pipe 2 write to grep child out", *infile,
              strerror(errno));

    if(close(ftg[0]) < 0)
      ERR_FAT(argv[0], "close: pipe 1 read in grep child", *infile,
              strerror(errno));
    if(close(ftg[1]) < 0)
      ERR_FAT(argv[0], "close: pipe 1 write in grep child", *infile,
              strerror(errno));
    if(close(gtm[1]) < 0)
      ERR_FAT(argv[0], "close: pipe 2 write in grep child", *infile,
              strerror(errno));

    if(execvp(*gargv, gargv) < 0)
      ERR_FAT(argv[0], "execvp: grep", pattern, strerror(errno));
  default:
    if(close(ftg[0]) < 0)
      ERR_FAT(argv[0], "close: pipe 1 read in parent", *infile,
              strerror(errno));
    if(close(gtm[1]) < 0)
      ERR_FAT(argv[0], "close: pipe 2 write in parent", *infile,
              strerror(errno));
```

```
    }

    if((ifd = open(*infile, O_RDONLY)) < 0)
      ERR_FAT(argv[0], "open: file for reading", *infile, strerror(errno));

    while(rlen = read(ifd, buf, BUF_SIZE)) {
      if(rlen < 0)
        ERR_FAT(argv[0], "reading of input file", *infile, strerror(errno));

      wtot = wlen = 0;
      while((wtot += wlen) < rlen) {
        if((wlen = write(ofd, buf+wtot, rlen-wtot)) < 0)
          ERR_FAT(argv[0], "writing of input file to pipe to grep", *infile,
                  strerror(errno));
        bp += wlen;
      }
    }

    if(close(ofd) < 0)
      ERR_FAT(argv[0], "closing output pipe", *infile, strerror(errno));
    if(close(ifd) < 0)
      ERR_FAT(argv[0], "closing input file", *infile, strerror(errno));
    ifd = ofd = -1;

    // no need to handle wait status
    waitpid(gpid, &wstatus, 0);
    waitpid(mpid, &wstatus, 0);
  }

  exit(EXIT_SUCCESS);
}
```

Example test case: ((arch)linux 5.3.8)

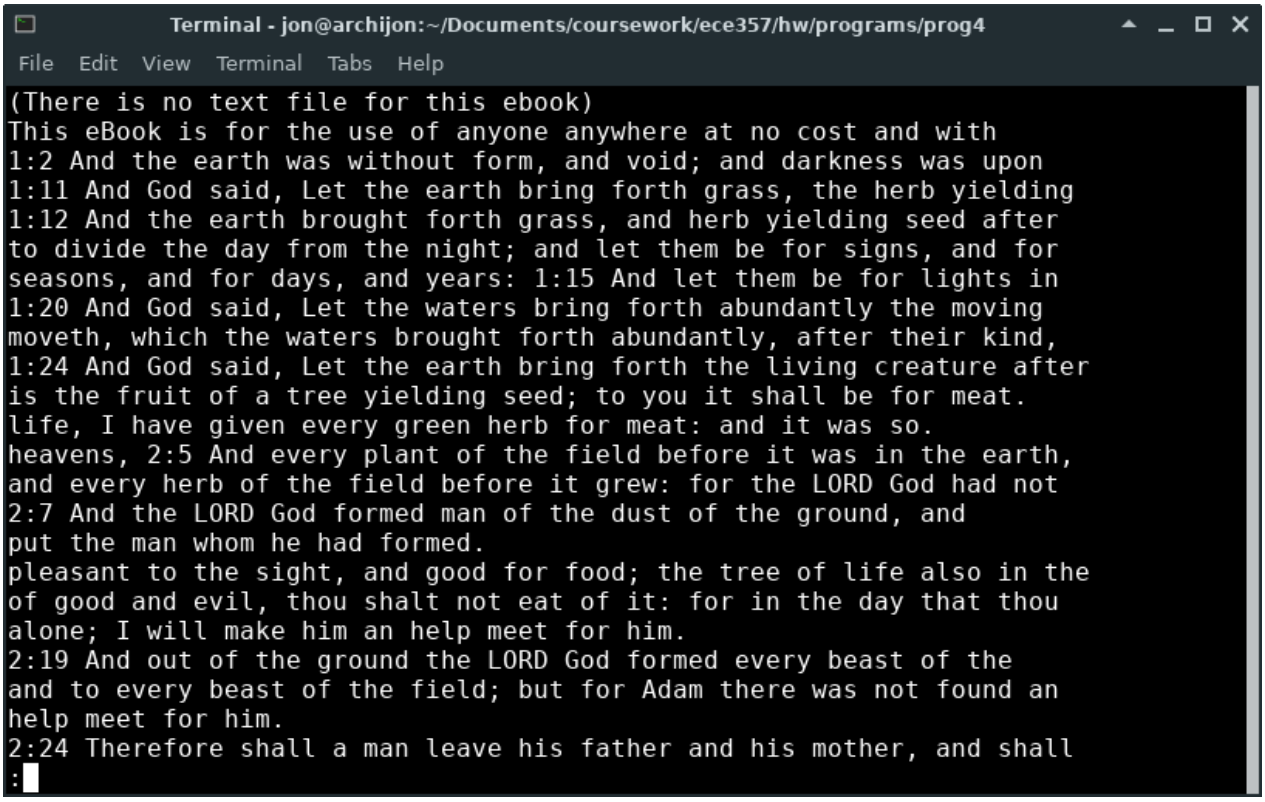
shell command:

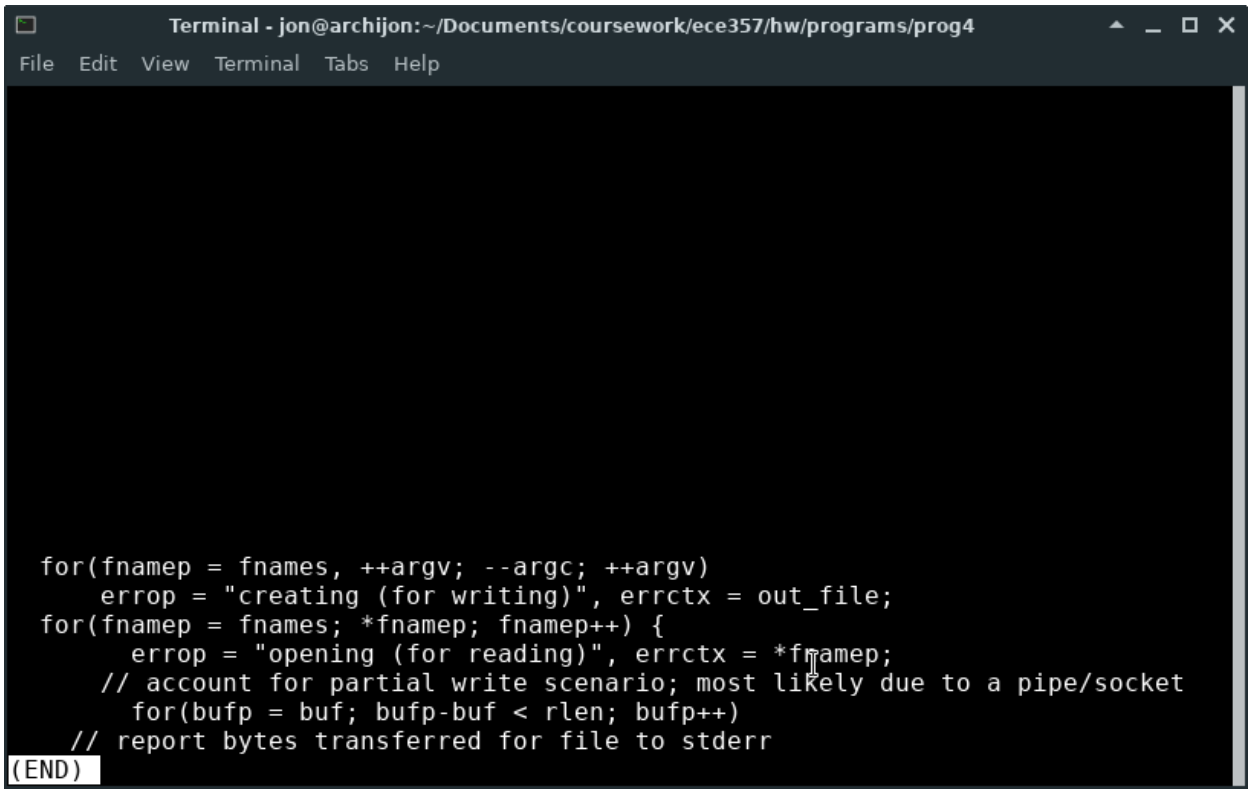
```
[jon@archijon prog4]$ ./catgrepless for bible.txt kitty.c catgrepless.c
```

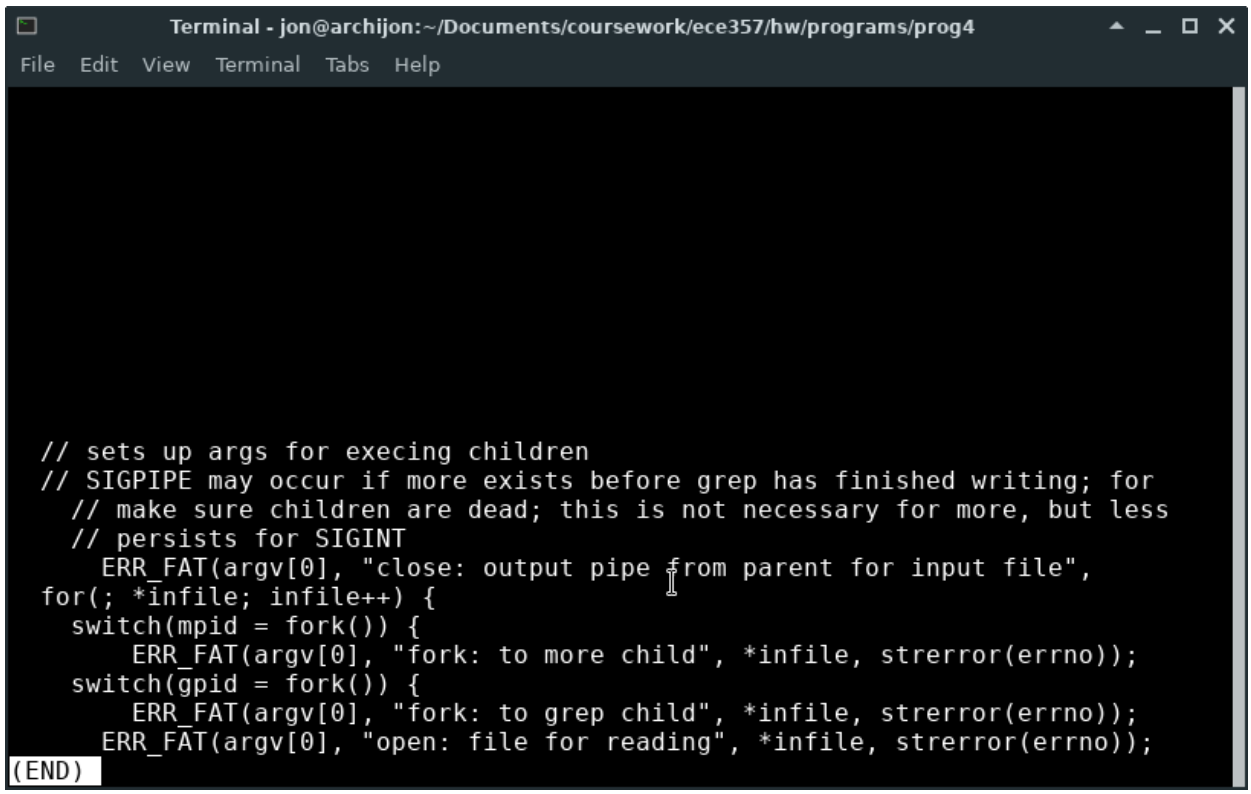
press:

q, Ctrl+C, q

screenshots: (i.e., this is just less working normally, quitting after each q or Ctrl+C)







stderr:

Warning: SIGINT encountered. 531895 bytes/2 files processed.

PSET 5

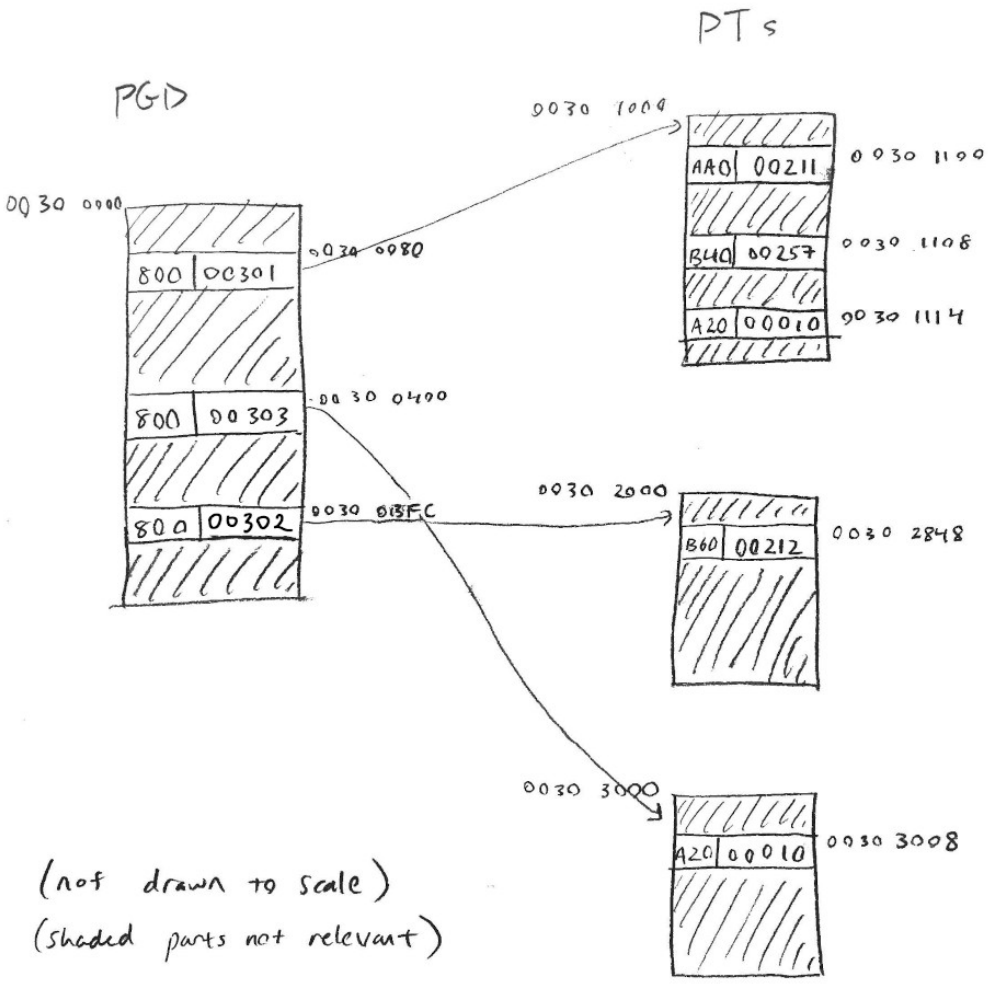
Jonathan Lam
Prof. Hakner
ECE 357
Computer OS
11/20/19

Problem 1 – Page Tables

a) Identfy the type of each memory region, based on the clues.

1) text (clues: map r/x/private, contains opcodes)
2) data (clues: map r/w/private, not empty)
3) bss (clues: map r/w/private, zero-filled)
4) bss – mmapped in (clues: map r/w/shared, zero-filled)
5) stack (clues: high address, map r/w/private/anon/growsdown, filled at end)

b) Draw out the PGD and any PTs that the kernel would have allocated.



c) Compose a brief program to reach the above state.

Note: From the diagram, we only know that these PTEs are marked not present, which can mean either that they have never been accessed (read from/written to/executed) or were paged in at some point and are currently paged out. This program (and the following answers) assumes that none of the non-present, mmap-ed in virtual pages (i.e., 0x40001000, 0x40003000, and 0x08044000) have never been accessed.

```
1      // assume compiler doesn't optimize out useless statements/functions
2
3      // .data
4      unsigned long i = 1, j = 1;
5
6      // two-page .bss
7      char l[8192];
8
9      int main(void) {
10       // keep accessing this to keep .stack accessed
11       int k;
12       char *m;
13
14       // create mmap-ed region
15       // assume 0x40001000 was chosen by mmap, not chosen explicitly
16       m = (char *) mmap(NULL, 12288, PROT_READ|PROT_WRITE,
17                         MAP_ANONYMOUS|MAP_SHARED, -1, 0);
18
19       // dirty .data; assume little-endian (shows correctly in core dump)
20       i = 0x45334543, j = 0x00003537;
21
22       // rapidly access both .bss and .stack regions so that PFRA doesn't
23       // try to reclaim it and access bit stays set
24       while(1) {
25         l[4096];
26         m[4096];
27         k;
28       }
29     }
```

d) Explain each page fault incurred by the program and resolution.

First, there is the minor page fault to create space for the PGD. Next, there is the mmap of the .text, .data, .bss, and .stack regions. These mmaps set up the valid memory regions but do not incur page faults.

Program execution begins in the .text region, so this region gets read, leading into the minor page fault to create the PT (0x00301000) and a major page fault (reading from a.out mapped to VA 0x08040000). The .text region has an instruction to add main to the stack, which causes a minor page fault for the PT (0x00302000) and a minor page fault in the stack, writing to a virgin anonymous page mapped to VA 0xBFFFF000.

In the program in part (c) above, next is a mmap to region 4 (no page fault). After that, there is a major page fault after writing to the data region (reading from a.out mapped to VA 0x08042000). On accessing (reading) the bss region, a minor page fault occurs mapping VA 0x08045FFF to the dedicated kernel page of all zeros (for COW optimization). For the mmap-ed region, a minor page fault occurs for the PT (0x00303000) and a minor page fault to read the data with VA 0x40002000 (similar to above, mapped to dedicated kernel page of all zeros).

This is a total of 2 major and 7 minor page faults.

e) What evidence from the diagram above shows that the PFRA has been recently active? Which page frame(s) might be a candidate for paging-out and reclamation?

Region 2 has been written to (dirty) and is not shared, but not accessed, which means that something (which can only be the PFRA) cleared the accessed bit. The PFRA must have run recently since it would claim pages for paging out if they have the accessed bit left unchecked for too long. Because this page is the only one with the accessed bit unset, it is the most likely candidate for being paged out.

There are no clear candidates for page reclamation just based on this diagram alone. However, if the valid memory regions that are not mapped to any page table have been swapped out (contrary to the assumption made in part (c)), these may be candidates for reclamation.

Problem 2 – mmap test programs

Test program (mtest.c)

```
 1      #include <errno.h>
 2      #include <fcntl.h>
 3      #include <signal.h>
 4      #include <stdio.h>
 5      #include <stdlib.h>
 6      #include <string.h>
 7      #include <sys/mman.h>
 8      #include <unistd.h>
 9
10      // test program vars; chosen arbitrarily and tweakable
11      #define TMP_PATH "/tmp/mtest.tmp"
12      #define TST_IND 3
13      #define TST_LEN 324
14      #define TST_CHR 'A'
15
16      // less strict error reporting, since these errors shouldn't happen
17      int err_chk(int res, char *cmd_name) {
18        if(res<0) {
19          dprintf(2, "./mtest: ERROR: failed on %s: %s.\n",
20                  cmd_name, strerror(errno));
21          exit(EXIT_FAILURE);
```

```
22        }
23        dprintf(2, "./mtest: Performing %s.\n", cmd_name);
24        return res;
25      }
26
27      // auxiliary fns
28      void sig_trap(int sig_num) {
29        dprintf(2, "./mtest: Received signal %d: %s. Exiting.\n",
30                sig_num, strsignal(sig_num));
31        exit(sig_num);
32      }
33      void trap_all_sigs() {
34        for(int i=1; i<=31; i++)
35          signal(i, sig_trap);
36      }
37      int create_tmp_file() {
38        int fd;
39        char buf[] = "Hello, world!";
40
41        // create and fill with some (known) bytes
42        fd = err_chk(open(TMP_PATH, O_CREAT|O_RDWR|O_TRUNC), "creat tmpfile");
43        err_chk(write(fd, buf, sizeof buf), "fill some data to tmpfile");
44        err_chk(unlink(TMP_PATH), "unlink tmpfile");
45
46        return fd;
47      }
48
49      // test 1: PROT_READ violation
50      void test_1() {
51        char *p, prev, post;
52
53        trap_all_sigs();
54        p = (char *) mmap(NULL, 4096, PROT_READ, MAP_ANON|MAP_SHARED, -1, 0);
55        err_chk(-(p==MAP_FAILED), "mmap tmpfile with MAP_ANON|MAP_SHARED");
56
57        prev = p[TST_IND];
58        dprintf(2, "./mtest: p[TST_IND]==%x.\n", prev);
59        dprintf(2, "./mtest: attempting write p[TST_IND]=%x.\n", prev+1);
60        post = ++p[TST_IND];
61
62        // diagnostic message and cleanup
63        err_chk(munmap(p, 4096), "munmap tmpfile");
64        dprintf(2, "./mtest: test byte: pre-write: %x post-write %x.\n", prev, post);
65        exit(-(prev==post));
66      }
67
68      // tests 2/3: writing to MAP_SHARED/MAP_PRIVATE
69      void test_23(int is_shared) {
70        int fd;
71        char *p, prev, post;
72
73        fd = create_tmp_file();
```

```
74        p = (char *) mmap(NULL, 4096, PROT_READ|PROT_WRITE,
75                          MAP_FILE|(is_shared?MAP_SHARED:MAP_PRIVATE), fd, 0);
76        err_chk(-(p==MAP_FAILED), "mmap tmpfile with MAP_SHARED or MAP_PRIVATE");
77
78        prev = p[TST_IND];
79        post = ++p[TST_IND];
80
81        // see if regular read call works
82        err_chk(lseek(fd, TST_IND, SEEK_SET), "lseek in tmpfile");
83        err_chk(read(fd, &post, 1), "read in tmpfile");
84
85        // diagnostic messae and cleanup
86        err_chk(munmap(p, 4096), "munmap tmpfile");
87        err_chk(close(fd), "close tmpfile");
88        dprintf(2, "./mtest: test byte: pre-write: %x post-write %x.\n", prev, post);
89        exit(prev==post);
90      }
91
92      // test 4: writing into a hole
93      void test_4() {
94        int fd;
95        char *p, post;
96
97        fd = create_tmp_file();
98        p = (char *) mmap(NULL, TST_LEN, PROT_READ|PROT_WRITE, MAP_FILE|MAP_SHARED,
99                          fd, 0);
100       err_chk(-(p==MAP_FAILED), "mmap tmpfile with MAP_SHARED");
101
102       p[TST_LEN] = TST_CHR;
103       err_chk(lseek(fd, TST_LEN+16, SEEK_SET), "lseek to TST_LEN+16 in tmpfile");
104       err_chk(write(fd, " ", 1), "write to TST_LEN+16 in tmpfile");
105       err_chk(lseek(fd, TST_LEN, SEEK_SET), "lseek to TST_LEN in tmpfile");
106       err_chk(read(fd, &post, 1), "read in tmpfile");
107
108       // diagnostic messae and cleanup
109       err_chk(munmap(p, TST_LEN), "munmap tmpfile");
110       err_chk(close(fd), "close tmpfile");
111       dprintf(2,"./mtest:test byte: %x; byte at pos TST_LEN+1:%x.\n",TST_CHR,post);
112       exit(1-(TST_CHR==post));
113     }
114
115     // usage: ./a.out [TEST_NUM]
116     int main(int argc, char **argv) {
117       if(argc > 1)
118         switch(**(argv+1)) {
119         case '1': test_1(); break;
120         case '2': test_23(1); break;
121         case '3': test_23(0); break;
122         case '4': test_4(); break;
123         }
124       exit(EXIT_FAILURE);
125     }
```

Sample Output

```
(base) [jon@archijon prog5]$ ./mtest 1
./mtest: Performing mmap tmpfile with MAP_ANON|MAP_SHARED.
./mtest: p[TST_IND]==0.
./mtest: attempting write p[TST_IND]=1.
./mtest: Received signal 11: Segmentation fault. Exiting.
(base) [jon@archijon prog5]$ echo $?
11
(base) [jon@archijon prog5]$ ./mtest 2
./mtest: Performing creat tmpfile.
./mtest: Performing fill some data to tmpfile.
./mtest: Performing unlink tmpfile.
./mtest: Performing mmap tmpfile with MAP_SHARED or MAP_PRIVATE.
./mtest: Performing lseek in tmpfile.
./mtest: Performing read in tmpfile.
./mtest: Performing munmap tmpfile.
./mtest: Performing close tmpfile.
./mtest: test byte: pre-write: 6c post-write 6d.
(base) [jon@archijon prog5]$ echo $?
0
(base) [jon@archijon prog5]$ ./mtest 3
./mtest: Performing creat tmpfile.
./mtest: Performing fill some data to tmpfile.
./mtest: Performing unlink tmpfile.
./mtest: Performing mmap tmpfile with MAP_SHARED or MAP_PRIVATE.
./mtest: Performing lseek in tmpfile.
./mtest: Performing read in tmpfile.
./mtest: Performing munmap tmpfile.
./mtest: Performing close tmpfile.
./mtest: test byte: pre-write: 6c post-write 6c.
(base) [jon@archijon prog5]$ echo $?
1
(base) [jon@archijon prog5]$ ./mtest 4
./mtest: Performing creat tmpfile.
./mtest: Performing fill some data to tmpfile.
./mtest: Performing unlink tmpfile.
./mtest: Performing mmap tmpfile with MAP_SHARED.
./mtest: Performing lseek to TST_LEN+16 in tmpfile.
./mtest: Performing write to TST_LEN+16 in tmpfile.
./mtest: Performing lseek to TST_LEN in tmpfile.
./mtest: Performing read in tmpfile.
./mtest: Performing munmap tmpfile.
./mtest: Performing close tmpfile.
./mtest: test byte: 41; byte at pos TST_LEN+1: 41.
(base) [jon@archijon prog5]$ echo $?
0
```

PSET 6

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
12/9/19

errfat.h

```
#ifndef __ERRFAT_H
#define __ERRFAT_H
#include <errno.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
extern char *PROG;
#define ERR_FAT(op, ctx, msg) {\
    dprintf(2, "%s: ERROR: %s (%s): %s\n", PROG, op, ctx, msg);\
    exit(EXIT_FAILURE);\
  }
#define ERRNO_FAT(op, ctx)\
  ERR_FAT(op, ctx, strerror(errno))
#endif
```

tas.h

```
#ifndef __TAS_H
int tas(volatile char *lock);
#define __TAS_H
#endif
```

tas.S (x64)

```
  .text       #See tas.S (32-bit version) for
.globl tas        #other comments
  .type tas,@function
tas:
  pushq %rbp
  movq  %rsp, %rbp
  movq  $1, %rax
  lock;xchgb  %al,(%rdi)  #arg1 is in the rdi register
  movsbq  %al,%rax    #sign-extend result into rax
  pop %rbp
  ret       #rax contains the return value
.Lfe1:
  .size tas,.Lfe1-tas
```

spinlock.h

```
#ifndef __SPINLOCK_H
#define __SPINLOCK_H
// shadows linux's spinlock_t in /include/linux/spinlock_types.h
// but that's what we're imitating here anyways
typedef volatile char spinlock_t;
void spin_lock(spinlock_t *sl);
void spin_unlock(spinlock_t *sl);
#endif
```

spinlock.c

```
#include "tas.h"
#include "spinlock.h"

void spin_lock(spinlock_t *sl) {
  while(tas(sl));
}
void spin_unlock(spinlock_t *sl) {
  *sl = 0;
}
```

seqlock.h

```
#ifndef __SEQLOCK_H
#define __SEQLOCK_H
#include "spinlock.h"

struct seqlock {
  spinlock_t lock;
  int count;
};
void write_seqlock(struct seqlock *s);
void write_sequnlock(struct seqlock *s);
int read_seqbegin(struct seqlock *s);
int read_seqretry(struct seqlock *s, int orig);
#endif
```

seqlock.c

```
#include <sched.h>
#include "seqlock.h"

// This implementation closely follows the lecture notes. I didn't experience
// issues with non-atomic count incrementing, so didn't protect the field
// with a spinlock.

void write_seqlock(struct seqlock *s) {
  spin_lock(&s->lock);
  s->count++;
}
void write_sequnlock(struct seqlock *s) {
  s->count++;
  spin_unlock(&s->lock);
};
int read_seqbegin(struct seqlock *s) {
  int a;
  while((a=s->count)%2)
    sched_yield();
  return a;
}
int read_seqretry(struct seqlock *s, int orig) {
  return s->count!=orig;
}
```

slab.h

```
#ifndef __SLAB_H
#define __SLAB_H
#include "spinlock.h"
#include "dll.h"

/**
  * Simple slab implementation. For simplicity, the slab is fixed-size,
  * may only slab-allocate struct dlls, and the freemap is a slot. Slab ops
  * are protected with a simple spinlock.
  */

// NSLOTS should be small enough to fill up
#define NSLOTS 10000

struct slab {
  char freemap[NSLOTS];
  struct dll slots[NSLOTS];
  spinlock_t sl;

  // for optimization purposes on insert -- see slab_alloc for details
  int pos;
};

/**
  * Allocate memory for slab. Not thread-safe.
  * @return         pointer to newly-allocated slab
  */
struct slab *slab_create();

/**
  * Unallocate slab. Not thread-safe.
  * @param          pointer to slab to de-allocate
  */
void slab_destroy(struct slab *slab);

/**
  * Allocate object in slab. Thread-safe.
  * @param slab     pointer to slab
  * @return         pointer to newly-allocated object, or NULL if slab is full
  */
void *slab_alloc(struct slab *slab);

/**
  * Deallocate object in slab. Thread-safe.
  * @param slab     pointer to slab
  * @param object   pointer to object to deallocate
  * @return         1 on success, -1 on failure
  */
int slab_dealloc(struct slab *slab, void *object);

// Allocate stats struct. Lumped in here to keep all mmap-ing in this file.
```

```
void stats_alloc();
#endif
```

slab.c

```
#include <sys/mman.h>
#include "errfat.h"
#include "slab.h"
#include "stats.h"

// See slab.h for more details.

struct slab *slab_create() {
  struct slab *slab;
  if((slab = (struct slab *)
        mmap(NULL, sizeof(struct slab),PROT_READ|PROT_WRITE,
             MAP_SHARED|MAP_ANONYMOUS, -1, 0)) == MAP_FAILED)
    ERRNO_FAT("mmap", "slab_create");
  slab->pos = 0;
  return slab;
}
void slab_destroy(struct slab *slab) {
  if(munmap(slab, sizeof(struct slab)) < 0)
    ERRNO_FAT("munmap", "slab_destroy");
}

// to increase performance, allocation searches for a free slot beginning
// from where it last left off (as opposed to searching from start every time)
// i.e., slab->pos
void *slab_alloc(struct slab *slab) {
  unsigned int pos_start;
  void *res_pos;

  spin_lock(&slab->sl);
  for(pos_start=slab->pos?slab->pos-1:NSLOTS-1;
      slab->freemap[slab->pos] && slab->pos!=pos_start;
      slab->pos=(slab->pos+1)%NSLOTS);
  if(slab->pos==pos_start) {
    spin_unlock(&slab->sl);
    return NULL;
  }
  slab->freemap[slab->pos] = 1;
  res_pos = slab->slots+slab->pos;
  spin_unlock(&slab->sl);
  return res_pos;
}
int slab_dealloc(struct slab *slab, void *object) {
  int pos;

  spin_lock(&slab->sl);
  pos = ((struct dll *)object) - slab->slots;
  if(pos>=NSLOTS || !slab->freemap[pos]) {
    spin_unlock(&slab->sl);
```

```
    return -1;
  }
  slab->freemap[pos] = 0;
  spin_unlock(&slab->sl);
  return 1;
}

// see slab.h for details
void stats_alloc() {
  if((stats = (struct stats *)
        mmap(NULL, sizeof(struct stats),PROT_READ|PROT_WRITE,
             MAP_SHARED|MAP_ANONYMOUS, -1, 0)) == MAP_FAILED)
    ERRNO_FAT("mmap", "struct stat");
}
```

dll.h

```
#ifndef __DLL_H
#define __DLL_H
#include "spinlock.h"
#include "seqlock.h"

/**
  * This implementation of a sorted integer-valued circular doubly-linked list
  * doesn't do much error checking, assumes correct inputs. It also assumes
  * that all nodes in a struct dll are allocated from the same struct slab
  * as its anchor. Both spinlock- and seqlock-protected versions are described
  * in this header file. A reasonable space optimization would be to store
  * the lock in the value field of the anchor, but this simple implementation
  * doesn't worry about saving space for clarity/simplicity of the lock.
  */

struct slab; // forward declaration b/c of circ. dep. -- see below
struct dll {
  int value;
  struct dll *fwd, *rev;

  // locks only used on anchor(s); see note above about saving space
  spinlock_t sl;          // for use with dll.c
  struct seqlock seqlock; // for use with dll2.c
};

#include "slab.h" // needs to be placed here b/c of circ. dep. b/t dll, slab

/**
  * Allocates and returns a DLL anchor. Not thread-safe (called from parent).
  * @param slab     slab to allocate from
  * @return         anchor node for DLL
  */
struct dll *dll_create(struct slab *slab);

/**
  * De-allocates all nodes in the DLL. Not thread-safe (called from parent).
  * @param anchor   anchor node for DLL
  * @param slab     slab where nodes are allocated
  */
void dll_destroy(struct dll *anchor, struct slab *slab);

/**
  * Insert an integer into the DLL. Thread-safe.
  * @param anchor   DLL anchor
  * @param value    integer value to add to DLL
  * @param slab     slab to allocate nodes from
  * @return         created node, or NULL on failure
  */
struct dll *spin_dll_insert(struct dll *anchor,int value,struct slab *slab);

/**
```

```
  * Delete a node from the DLL. Thread-safe.
  * @param anchor   DLL anchor
  * @param node     pointer to node to delete
  * @param slab     slab that node is allocated in
  * @return         1 on success, -1 on failure
  */
int spin_dll_delete(struct dll *anchor, struct dll *node,struct slab *slab);

/**
  * Find the first node with a given value in the DLL. Thread-safe.
  * @param anchor   DLL anchor
  * @param value    integer value to search for
  * @return         pointer to first node in DLL containing value, or NULL
  */
struct dll *spin_dll_find(struct dll *anchor,int value);

/**
  * Analogous functions protected using seqlock (extra credit). The insert
  * and delete functions are protected with write seqlocks, and the find is
  * protected with an optimistic read seqlock. Thread-safe. Keeps track of
  * statistics in stats (see stats.h);
  */
struct dll *seq_dll_insert(struct dll *anchor, int value, struct slab *slab);
int seq_dll_delete(struct dll *anchor, struct dll *node, struct slab *slab);
struct dll *seq_dll_find(struct dll *anchor, int value);

void print_dll(struct dll *anchor); // for debugging
#endif
```

dll.c

```
#include "errfat.h"
#include "dll.h"
#include "slab.h"

// This dll implementation uses spinlocking. See dll.h for more info.

struct dll *dll_create(struct slab *slab) {
  struct dll *anchor;
  if(!(anchor = (struct dll *) slab_alloc(slab)))
    return NULL;
  anchor->fwd = anchor->rev = anchor;
  return anchor;
}
void dll_destroy(struct dll *anchor, struct slab *slab) {
  // it_fwd to avoid accessing it->nxt after it has been deallocated (and
  // possibly already reallocated)
  struct dll *it, *it_fwd;
  it = anchor, it_fwd = it->fwd;
  do {
    it_fwd = it->fwd;
    slab_dealloc(slab, it);
  } while((it=it_fwd) != anchor);
```

```
}

struct dll *spin_dll_insert(struct dll *anchor, int value, struct slab *slab) {
  struct dll *new_node, *it;

  if(!(new_node = (struct dll *) slab_alloc(slab)))
    return NULL;
  new_node->value = value;

  spin_lock(&anchor->sl);
  for(it=anchor->fwd; it->value<value && it!=anchor; it=it->fwd);
  new_node->fwd = it;
  new_node->rev = it->rev;
  it->rev = it->rev->fwd = new_node;
  spin_unlock(&anchor->sl);
  return new_node;
}

int spin_dll_delete(struct dll *anchor, struct dll *node, struct slab *slab) {
  spin_lock(&anchor->sl);
  // this condition if multiple deletes on same dll in quick succession
  if(!node || node->fwd==node) {
    spin_unlock(&anchor->sl);
    return -1;
  }
  node->rev->fwd = node->fwd;
  node->fwd->rev = node->rev;
  node->fwd = node->rev = node;
  spin_unlock(&anchor->sl);

  if(slab_dealloc(slab, node)<0)
    ERR_FAT("slab_dealloc", "", "Deallocating node failed");
  return 1;
}

struct dll *spin_dll_find(struct dll *anchor, int value) {
  struct dll *it;

  spin_lock(&anchor->sl);
  // some preliminary checks/optimizations
  if(anchor->fwd==anchor || value<anchor->fwd->value
     || value>anchor->rev->value) {
    spin_unlock(&anchor->sl);
    return NULL;
  }

  for(it=anchor->fwd; it->value<value && it!=anchor; it=it->fwd);
  spin_unlock(&anchor->sl);
  return it->value==value && it!=anchor ? it : NULL;
}

// for debugging
```

```
void print_dll(struct dll *anchor) {
  struct dll *it;

  spin_lock(&anchor->sl);
  dprintf(1, "printing dll: ");
  for(it = anchor->fwd; it != anchor; it=it->fwd)
    dprintf(1, "%d ", it->value);
  dprintf(1, "\n");
  spin_unlock(&anchor->sl);
}
```

seqdll.c

```
#include "errfat.h"
#include "dll.h"
#include "slab.h"
#include "stats.h"

// This dll implementation uses seqlocking. See dll.h for more info.

struct dll *seq_dll_insert(struct dll *anchor, int value, struct slab *slab) {
  struct dll *new_node, *it;
  int seqlock_cnt;

  if(!(new_node = (struct dll *) slab_alloc(slab)))
    return NULL;
  new_node->value = value;

  write_seqlock(&anchor->seqlock);
  for(it=anchor->fwd; it->value<value&&it!=anchor&&it!=it->fwd; it=it->fwd);
  new_node->fwd = it;
  new_node->rev = it->rev;
  it->rev = it->rev->fwd = new_node;
  write_sequnlock(&anchor->seqlock);
  return new_node;
}

int seq_dll_delete(struct dll *anchor, struct dll *node, struct slab *slab) {
  write_seqlock(&anchor->seqlock);
  // this condition if multiple deletes on same dll in quick succession
  if(!node || node->fwd == node) {
    write_sequnlock(&anchor->seqlock);
    return -1;
  }
  node->rev->fwd = node->fwd;
  node->fwd->rev = node->rev;
  node->fwd = node->rev = node;
  write_sequnlock(&anchor->seqlock);

  if(slab_dealloc(slab, node)<0)
    ERR_FAT("slab_dealloc", "", "Deallocating node failed");
  return 1;
}
```

```
struct dll *seq_dll_find(struct dll *anchor, int value) {
  struct dll *it;
  int seqlock_cnt, cnt=0;

  do {
    seqlock_cnt = read_seqbegin(&anchor->seqlock);
    if(anchor->fwd==anchor || value<anchor->fwd->value
       || value>anchor->rev->value)
      return NULL;

    for(it=anchor->fwd; it->value<value&&it!=anchor&&it!=it->fwd; it=it->fwd);
  } while(++cnt, read_seqretry(&anchor->seqlock, seqlock_cnt));

  spin_lock(&stats->lock);
  stats->att_seqlock_read += cnt;
  stats->suc_seqlock_read++;
  spin_unlock(&stats->lock);
  return it->value==value && it!=anchor ? it : NULL;
}
```

spinlocktest.c

```
// usage: spinlocktest [thread_count] [sample_count]

#include <stdio.h>
#include <sys/mman.h>
#include <sys/wait.h>
#include <unistd.h>
#include "errfat.h"
#include "spinlock.h"

#define THREAD_CNT_DFL 8
#define SAMPLE_CNT_DFL 100000
#define PROG "spinlocktest"

// generate spinlock test
void transact_nospinlock(int *p, int sample_cnt) {
  for(int i = 0; i < sample_cnt; i++)
    (*p)++;
}
void transact_spinlock(int *p, spinlock_t *sl, int sample_cnt) {
  for(int i = 0; i < sample_cnt; i++) {
    spin_lock(sl);
    (*p)++;
    spin_unlock(sl);
  }
}
void generate_spinlock_test(int use_spinlock, int thread_cnt, int sample_cnt) {
  int *p, i, wstatus, pid;
  spinlock_t *sl;

  // create shared mmap region; first (sizeof(int)) bytes for data to rapidly
  // change, and last (sizeof(char)) bytes for spinlock
  if((p = (int *)mmap(NULL,sizeof(int)+sizeof(spinlock_t),PROT_READ|PROT_WRITE,
                      MAP_SHARED|MAP_ANONYMOUS, -1, 0)) == MAP_FAILED)
    ERRNO_FAT("mmap", "shared region");
  sl = (spinlock_t *)(p+1);

  // create thread_cnt processes; do sample_cnt transactions in each child,
  // don't do anything in parent
  for(i=0; i<thread_cnt; i++)
    switch(fork()) {
    case -1:
      ERRNO_FAT("fork", i);
    case 0:
      if(use_spinlock)
        transact_spinlock(p, sl, sample_cnt);
      else
        transact_nospinlock(p, sample_cnt);
      exit(EXIT_SUCCESS);
    }

  // aggregate results and cleanup; ignore wstatus
```

```
  for(i=0; i<thread_cnt; i++)
    if(pid=wait(&wstatus)<0)
      ERRNO_FAT("wait", pid);
  dprintf(2, "%d\n", *p);
  if(munmap(p, sizeof(int)+sizeof(spinlock_t))<0)
    ERRNO_FAT("munmap", "shared region");
}
int main(int argc, char **argv) {
  int thread_cnt, sample_cnt;
  if(argc<3) {
    thread_cnt = THREAD_CNT_DFL;
    sample_cnt = SAMPLE_CNT_DFL;
  } else {
    thread_cnt = atoi(argv[1]);
    sample_cnt = atoi(argv[2]);
  }

  dprintf(1, "Processes:\t%d\nSamples/proc:\t%d\nExpected total:\t%d\n---\n",
          thread_cnt, sample_cnt, thread_cnt*sample_cnt);
  dprintf(1, "w/o spinlock:\t");
  generate_spinlock_test(0, thread_cnt, sample_cnt);
  dprintf(1, "w/ spinlock:\t");
  generate_spinlock_test(1, thread_cnt, sample_cnt);
}
```

Sample output (spinlocktest.c)

```
Processes:  8
Samples/proc:     100000
Expected total:   800000
---
w/o spinlock:     292355
w/ spinlock:      800000
```

stats.h

```
#ifndef __STATS_H
#define __STATS_H
#include "spinlock.h"

extern struct stats *stats;
struct stats {
  spinlock_t lock;
  unsigned int att_seqlock_read, suc_seqlock_read, net_dll_len_chg;
};
#endif
```

slabtest.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <sys/wait.h>
#include <time.h>
#include <unistd.h>
#include "errfat.h"
#include "dll.h"
#include "slab.h"
#include "spinlock.h"
#include "stats.h"

#define THREAD_CNT 16
#define SAMPLE_CNT 10000
#define SAMPLE_MAX 1000
char *PROG = "slabtest";

// check if dll is sorted and of correct length
void check_dll(struct dll *anchor, int exp_len, int is_seqlock) {
  struct dll *it, *it_fwd;
  int tot_cnt, err_cnt;

  dprintf(2, "=====\nSTRESS CHECK COMPLETE. CHECKING DLL.\n");
  for(it=anchor->fwd, it_fwd=it->fwd, tot_cnt=err_cnt=0; it!=anchor;
      it_fwd=(it=it_fwd)->fwd, tot_cnt++)
    if(it_fwd->value<it->value && it_fwd!=anchor) {
      printf("%d\n", it->value);
      err_cnt++;
    }
  dprintf(2, "=====\nOVERALL CHECK:\nSORTING ERRORS:\t%d\nDLL LENGTH:\t%d\n"
             "EXP DLL LENGTH:\t%d\nLENGTH ERROR:\t%d\n",
          err_cnt, tot_cnt, exp_len, (exp_len-tot_cnt)*(exp_len>tot_cnt?1:-1));
  if(is_seqlock)
    dprintf(2, "OPTIMISTIC SEQLOCK SUCCESS RATE: %d/%d (%f\%)\n",
            stats->suc_seqlock_read,
            stats->att_seqlock_read,
            ((float)stats->suc_seqlock_read)/stats->att_seqlock_read);
}
```

```
// driver for slab testing
struct stats *stats;
void generate_slab_test(int is_seqlock) {
  // att/suc_op_cnt: attempted and successful operation counts
  struct slab *slab;
  struct dll *dll, *p;
  struct timeval proc_start, proc_end;
  int i, wstatus, att_op_cnt[3], suc_op_cnt[3], net_len, pid;
  long elap_usec;

  struct dll *(*dll_insert)(struct dll *,int,struct slab *);
  int (*dll_delete)(struct dll *,struct dll *,struct slab *);
  struct dll *(*dll_find)(struct dll *,int);

  if(!(slab = slab_create()))
    ERR_FAT("slab_create", "main slab", "Error creating slab");
  if(!(dll = dll_create(slab)))
    ERR_FAT("dll_create", "main slab", "Error creating dll");

  // set up shared statistics memory region -- see stats.h
  stats_alloc();

  // get correct functions
  dprintf(2, "=====\n%s TEST\n", is_seqlock?"SEQLOCK":"SPINLOCK");
  dll_insert = is_seqlock?seq_dll_insert:spin_dll_insert;
  dll_delete = is_seqlock?seq_dll_delete:spin_dll_delete;
  dll_find = is_seqlock?seq_dll_find:spin_dll_find;

  memset(att_op_cnt, 0, 3*sizeof(int));
  memset(suc_op_cnt, 0, 3*sizeof(int));
  dprintf(2, "PROC\tATT INS\tATT DEL\tATT FND\tSUC INS\tSUC DEL\tSUC FND\t"
             "NET CHG\tELP TME\n");
  for(i = 0; i < THREAD_CNT; i++) {
    switch(fork()) {
    case -1:
      ERRNO_FAT("fork", i);
    case 0:
      srand(time(NULL)+i); // should generate unique seed for each proc
      gettimeofday(&proc_start, NULL);
      for(int j=0; j<SAMPLE_CNT; j++) {
        switch(rand()%3) {
        case 0:
          att_op_cnt[0]++;
          if((*dll_insert)(dll, rand()%SAMPLE_MAX, slab))
            suc_op_cnt[0]++;
          break;
        case 1:
          att_op_cnt[1]++;
          if((*dll_find)(dll, rand()%SAMPLE_MAX))
            suc_op_cnt[1]++;
          break;
        case 2:
```

```
          att_op_cnt[2]++;
          if((*dll_delete)(dll, dll_find(dll, rand()%SAMPLE_MAX), slab)>0)
            suc_op_cnt[2]++;
          break;
        }
      }
      gettimeofday(&proc_end, NULL);
      elap_usec = 1000000*(proc_end.tv_sec-proc_start.tv_sec)
                  + (proc_end.tv_usec-proc_start.tv_usec);
      dprintf(2, "%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%6.03lfs\n",
              i, att_op_cnt[0], att_op_cnt[1], att_op_cnt[2],
              suc_op_cnt[0], suc_op_cnt[1], suc_op_cnt[2],
              suc_op_cnt[0]-suc_op_cnt[2],
              elap_usec/1e6);
      spin_lock(&stats->lock);
      stats->net_dll_len_chg += suc_op_cnt[0]-suc_op_cnt[2];
      spin_unlock(&stats->lock);
      exit(EXIT_SUCCESS);
    }
  }

  for(i=0; i<THREAD_CNT; i++) {
    if(pid=wait(&wstatus)<0) {
      ERRNO_FAT("wait", pid);
    } else if(wstatus)
      // non-fatal notice: child process died with non-zero exit code
      dprintf(2, "%s: wait: \"%d\": Process terminated with exit status %d\n",
              pid, wstatus);
  }

  check_dll(dll, stats->net_dll_len_chg, is_seqlock);

  dll_destroy(dll, slab);
  slab_destroy(slab);
}
int main(void) {
  dprintf(2, "=====\nslabtest.c\n"
             "=====\nTHREAD_CNT:\t%d\nSAMPLE_CNT:\t%d\n"
             "SAMPLE_MAX:\t%d\nNSLOTS:\t\t%d\n",
          THREAD_CNT, SAMPLE_CNT, SAMPLE_MAX, NSLOTS);

  // with spinlock
  generate_slab_test(0);

  // with seqlock
  generate_slab_test(1);
}
```

Test case: Medium parameter values

```
=====
slabtest.c
=====
THREAD_CNT:       16
SAMPLE_CNT:       10000
SAMPLE_MAX:       10000
NSLOTS:           10000
=====
SPINLOCK TEST
```

PROC	ATT INS	ATT DEL	ATT FND	SUC INS	SUC DEL	SUC FND	NET CHG	ELP TME
3	3336	3317	3347	2631	1394	1404	1227	18.876s
6	3366	3290	3344	2586	1331	1376	1210	19.970s
5	3352	3246	3402	2244	1608	1694	550	20.515s
1	3350	3275	3375	2467	1473	1477	990	20.713s
15	3339	3348	3313	2307	1641	1669	638	22.016s
13	3324	3375	3301	2199	1698	1687	512	22.486s
8	3350	3386	3264	2201	1712	1664	537	22.669s
10	3384	3276	3340	2260	1652	1698	562	22.882s
12	3287	3384	3329	2215	1705	1727	488	22.959s
7	3353	3364	3283	2095	1762	1691	404	23.126s
0	3343	3395	3262	2292	1518	1459	833	23.236s
4	3327	3365	3308	2125	1747	1672	453	23.272s
2	3349	3344	3307	2114	1707	1687	427	23.301s
11	3425	3299	3276	2171	1675	1694	477	23.300s
14	3300	3323	3377	2010	1683	1746	264	23.304s
9	3385	3342	3273	2121	1689	1695	426	23.339s

```
=====
STRESS CHECK COMPLETE. CHECKING DLL.
=====
OVERALL CHECK:
SORTING ERRORS:   0
DLL LENGTH:       9998
EXP DLL LENGTH:   9998
LENGTH ERROR:     0
=====
SEQLOCK TEST
```

PROC	ATT INS	ATT DEL	ATT FND	SUC INS	SUC DEL	SUC FND	NET CHG	ELP TME
2	3309	3308	3383	2531	1405	1462	1069	5.230s
0	3435	3242	3323	2570	1392	1423	1147	5.271s
4	3427	3269	3304	2474	1503	1506	968	5.561s
10	3341	3425	3234	2349	1572	1446	903	5.584s
15	3306	3300	3394	2337	1673	1690	647	5.984s
1	3270	3386	3344	2202	1594	1575	627	6.026s
8	3396	3305	3299	2155	1683	1668	487	6.050s
3	3281	3393	3326	2228	1702	1680	548	6.125s
11	3339	3307	3354	2151	1706	1732	419	6.169s
14	3296	3324	3380	2129	1625	1717	412	6.195s
6	3304	3311	3385	2100	1687	1698	402	6.241s
9	3395	3285	3320	2218	1652	1704	514	6.247s
7	3419	3297	3284	2275	1671	1680	595	6.271s
13	3276	3379	3345	2101	1757	1712	389	6.267s

```
5     3307        3390        3303        2155        1708        1628        527         6.279s
12    3302        3288        3410        2061        1661        1717        344         6.276s
=====
STRESS CHECK COMPLETE. CHECKING DLL.
=====
OVERALL CHECK:
SORTING ERRORS:   0
DLL LENGTH:       9998
EXP DLL LENGTH:   9998
LENGTH ERROR:     0
OPTIMISTIC SEQLOCK SUCCESS RATE: 106558/256690 (0.415123%)
```

Test case: More trials, higher chance of find/delete (lower SAMPLE_MAX)

```
=====
slabtest.c
=====
THREAD_CNT:       16
SAMPLE_CNT:       100000
SAMPLE_MAX:       1000
NSLOTS:           10000
=====
SPINLOCK TEST
PROC  ATT INS     ATT DEL     ATT FND     SUC INS     SUC DEL     SUC FND     NET CHG     ELP TME
2     33165       33485       33350       31070       30432       30302       768         265.390s
9     33330       33346       33324       30993       30467       30418       575         275.320s
1     33102       33369       33529       30927       30254       30485       442         277.109s
8     33297       33415       33288       31063       30596       30393       670         277.906s
13    33506       33296       33198       31142       30428       30300       842         279.324s
14    33376       33348       33276       31081       30459       30361       720         279.644s
3     33281       33360       33359       30961       30319       30170       791         280.501s
5     33520       33289       33191       31206       30134       29868       1338        282.112s
4     33329       33231       33440       31094       30357       30518       576         282.790s
12    33286       33210       33504       31037       30376       30583       454         282.915s
7     33351       33124       33525       30951       30339       30683       268         283.135s
15    33380       33438       33182       30848       30690       30376       472         283.849s
6     33178       33516       33306       30814       30557       30454       360         284.059s
10    33146       33455       33399       30874       30539       30519       355         284.186s
0     33252       33453       33295       30907       30545       30271       636         284.226s
11    33484       33164       33352       31228       30326       30503       725         284.216s
=====
STRESS CHECK COMPLETE. CHECKING DLL.
=====
OVERALL CHECK:
SORTING ERRORS:   0
DLL LENGTH: 9992
EXP DLL LENGTH:   9992
LENGTH ERROR:     0
=====
SEQLOCK TEST
PROC  ATT INS     ATT DEL     ATT FND     SUC INS     SUC DEL     SUC FND     NET CHG     ELP TME
4     33393       33287       33320       31007       29765       29761       1246        65.347s
0     33483       33355       33162       31162       29767       29589       1573        67.084s
2     33666       33139       33195       31143       30205       30188       955         69.178s
1     33308       33374       33318       30920       30233       30173       747         69.597s
15    33461       33104       33435       30881       29976       30383       498         69.763s
3     33304       33288       33408       30825       30184       30279       546         70.631s
6     33364       33336       33300       30869       30306       30332       537         70.666s
7     33421       33266       33313       30884       30369       30335       549         70.781s
10    33145       33598       33257       30665       30542       30162       503         70.850s
8     33341       33325       33334       30677       30257       30347       330         70.885s
13    33187       33504       33309       30680       30440       30306       374         71.035s
11    33387       33283       33330       30797       30309       30179       618         71.223s
14    33360       33350       33290       30823       30400       30259       564         71.212s
9     33338       33375       33287       30740       30405       30277       463         71.294s
```

```
12    33070       33428       33502       30572       30437       30500       72          71.297s
5     33145       33756       33099       30441       30772       30017       424         71.323s
=====
STRESS CHECK COMPLETE. CHECKING DLL.
=====
OVERALL CHECK:
SORTING ERRORS:   0
DLL LENGTH:       9999
EXP DLL LENGTH:   9999
LENGTH ERROR:     0
OPTIMISTIC SEQLOCK SUCCESS RATE: 1066512/2874594 (0.371013%)
```

Sample test case: Many operations on small dlls

```
=====
slabtest.c
=====
THREAD_CNT: 8
SAMPLE_CNT: 10000000
SAMPLE_MAX: 10
NSLOTS:          10
=====
SPINLOCK TEST
PROC  ATT INS     ATT DEL     ATT FND     SUC INS     SUC DEL     SUC FND     NET CHG     ELP TME
5     3330938     3333905     3335157     1343106     1412584     1365299     -22193      36.062s
0     3330540     3337773     3331687     1382338     1414653     1361455     20883       37.336s
7     3332757     3334019     3333224     1348168     1412597     1362722     -14554      37.329s
6     3333522     3333159     3333319     1353921     1417094     1369539     -15618      37.399s
2     3332200     3333983     3333817     1380033     1417873     1364507     15526       37.494s
1     3336070     3332361     3331569     1384245     1420747     1369157     15088       37.568s
3     3334897     3332747     3332356     1353696     1420963     1370791     -17095      37.585s
4     3333693     3332416     3333891     1396243     1427207     1378272     17971       37.595s
=====
STRESS CHECK COMPLETE. CHECKING DLL.
=====
OVERALL CHECK:
SORTING ERRORS:   0
DLL LENGTH: 8
EXP DLL LENGTH:   8
LENGTH ERROR:     0
=====
SEQLOCK TEST
PROC  ATT INS     ATT DEL     ATT FND     SUC INS     SUC DEL     SUC FND     NET CHG     ELP TME
0     3333601     3335776     3330623     1517381     1554593     1514760     2621        8.165s
5     3333186     3333231     3333583     1517783     1554074     1512693     5090        8.349s
6     3334786     3331539     3333675     1517505     1550790     1513189     4316        8.791s
7     3332195     3333908     3333897     1514807     1552548     1513864     943         8.866s
2     3332227     3333074     3334699     1506656     1549849     1513289     -6633       8.939s
4     3330786     3335409     3333805     1507838     1552305     1512145     -4307       8.988s
1     3333787     3330356     3335857     1512716     1549427     1510325     2391        8.992s
3     3335166     3330967     3333867     1507613     1550340     1512025     -4412       8.996s
=====
STRESS CHECK COMPLETE. CHECKING DLL.
=====
OVERALL CHECK:
SORTING ERRORS:   0
DLL LENGTH: 9
EXP DLL LENGTH:   9
LENGTH ERROR:     0
OPTIMISTIC SEQLOCK SUCCESS RATE: 42246863/54273870 (0.778401%)
```

Sample test case: Single-threaded performance

```
=====
slabtest.c
=====
THREAD_CNT:       1
SAMPLE_CNT:       10000000
SAMPLE_MAX:       1000
NSLOTS:           1000
=====
SPINLOCK TEST
PROC  ATT INS     ATT DEL     ATT FND     SUC INS     SUC DEL     SUC FND     NET CHG     ELP TME
0     3334101     3330914     3334985     1667039     1663457     1666040     999         25.803s
=====
STRESS CHECK COMPLETE. CHECKING DLL.
=====
OVERALL CHECK:
SORTING ERRORS:   0
DLL LENGTH: 999
EXP DLL LENGTH:   999
LENGTH ERROR:     0
=====
SEQLOCK TEST
PROC  ATT INS     ATT DEL     ATT FND     SUC INS     SUC DEL     SUC FND     NET CHG     ELP TME
0     3332923     3333276     3333801     1669508     1666818     1668511     997         25.442s
=====
STRESS CHECK COMPLETE. CHECKING DLL.
=====
OVERALL CHECK:
SORTING ERRORS:   0
DLL LENGTH:       997
EXP DLL LENGTH:   997
LENGTH ERROR:     0
OPTIMISTIC SEQLOCK SUCCESS RATE: 6653891/6653891 (1.000000%)
```

Sample test case: Many threads

```
=====
slabtest.c
=====
THREAD_CNT:       1000
SAMPLE_CNT:       1000
SAMPLE_MAX:       1000
NSLOTS:           1000
=====
SPINLOCK TEST
PROC  ATT INS     ATT DEL     ATT FND     SUC INS     SUC DEL     SUC FND     NET CHG     ELP TME
0     340         339         321         340         82          83          257         0.002s
1     313         346         341         313         87          78          235         0.002s
5     328         329         343         324         174         167         157         0.091s
210   354         325         321         139         159         149         -10         4.670s
796   332         326         342         181         159         168         13          4.632s
16    354         316         330         132         150         136         -4          8.046s
539   366         310         324         119         114         126         -7          16.551s
742   328         370         302         139         149         118         21          19.252s
551   341         299         360         101         115         142         -41         23.219s
3     319         345         336         272         131         118         154         25.562s
723   346         314         340         133         121         133         0           25.315s
636   360         318         322         111         114         137         -26         32.399s
[...truncated…]
431   341         324         335         143         129         146         -3          341.603s
34    329         322         349         149         150         169         -20         342.821s
680   347         304         349         160         138         154         6           340.448s
671   324         332         344         165         134         164         1           340.425s
357   319         351         330         157         158         155         2           342.040s
493   331         351         318         154         175         151         3           341.646s
102   355         314         331         185         152         154         31          342.796s
830   340         343         317         163         155         152         11          339.345s
623   321         370         309         144         168         148         -4          340.491s
=====
STRESS CHECK COMPLETE. CHECKING DLL.
=====
OVERALL CHECK:
SORTING ERRORS:   0
DLL LENGTH:       999
EXP DLL LENGTH:   999
LENGTH ERROR:     0
=====
SEQLOCK TEST
PROC  ATT INS     ATT DEL     ATT FND     SUC INS     SUC DEL     SUC FND     NET CHG     ELP TME
0     308         340         352         308         90          91          217         0.002s
3     316         334         350         316         106         112         204         0.002s
1     325         321         354         264         92          113         151         4.840s
113   306         338         356         130         136         149         -19         5.297s
258   320         365         315         134         167         124         10          6.604s
153   317         340         343         133         142         138         -5          6.661s
55    324         328         348         182         153         181         1           7.058s
364   311         357         332         135         166         127         8           6.829s
```

```
[...truncated…]
994   349         332         319         149         142         131         18          12.768s
117   353         320         327         152         142         139         13          17.253s
561   333         342         325         147         149         163         -16         16.277s
921   326         315         359         144         137         151         -7          14.014s
660   307         310         383         120         126         177         -57         15.507s
953   328         354         318         144         166         142         2           12.804s
977   326         316         358         140         131         143         -3          12.781s
727   325         321         354         161         137         165         -4          15.381s
944   338         314         348         155         143         162         -7          13.582s
754   321         350         329         148         159         146         2           14.692s
=====
STRESS CHECK COMPLETE. CHECKING DLL.
=====
OVERALL CHECK:
SORTING ERRORS:   0
DLL LENGTH:       997
EXP DLL LENGTH:   997
LENGTH ERROR:     0
OPTIMISTIC SEQLOCK SUCCESS RATE: 664770/1022575 (0.650094%)
```

Summary of results from test cases

- Both these spinlock and seqlock implementations seem to be successful in maintaining a DLL correctly (at least, the length of the DLL after the stress test is equal to the net changes in length to the DLL from each process, and it remains sorted.)
- It seems that with random results, we can expect at least a 30% success rate.
- For the first three test cases, with a medium number of samples and threads, seqlock seems to be 4-4.5x faster than seqlock. In the single-threaded test case, both appear to have almost the same performance (as expected). For highly-threaded and small number of trials per thread in the last test case, there was an enormous 20x speed improvement.
- The DLL/slab operations are much faster on small DLLs/slabs (expected because both only use linear operations), but it’s unclear whether this affects spinlocks or seqlocks more.

PSET 7

Jonathan Lam
Prof. Hakner
Computer OS
12/15/19

PROBLEM 1 – Some assembly required

helloworld.S

```
.data
  str:  .string "Hello, world\!\n"

.text
  # write(1, "Hello, world!\n", 14)
  movl  $0x4,%eax   # write syscall# = 4
  movl  $0x1,%ebx   # fd = 1
  movl  $str,%ecx   # addr of string "Hello, world!\n"
  movl  $14,%edx    # len of string "Hello, world!\n"
  int   $0x80       # invoke syscall

  # exit(0)
  movl  %eax,%ebx   # exit code = retval of write
  movl  $0x1,%eax   # exit syscall# = 1
  int   $0x80       # invoke syscall
```

Screenshot of terminal output for compilation/linking/running/strace-ing:

```
(base) [jon@archijon prog7]$ as -o helloworld.o helloworld.S --32
(base) [jon@archijon prog7]$ ld -o helloworld helloworld.o -m elf_i386
ld: warning: cannot find entry symbol _start; defaulting to 0000000008049000
(base) [jon@archijon prog7]$ ./helloworld
Hello, world!
(base) [jon@archijon prog7]$ echo $?
14
(base) [jon@archijon prog7]$ strace ./helloworld
execve("./helloworld", ["./helloworld"], 0x7ffc76f92120 /* 44 vars */) = 0
strace: [ Process PID=23065 runs in 32 bit mode. ]
write(1, "Hello, world!\n", 14Hello, world!
)         = 14
exit(14)                                = ?
+++ exited with 14 +++
```

PROBLEM 2 – Scheduling

The following program can be called in the form given in the assignment (./nicetest [num_proc] [nice_val] [test_time]) or without arguments to generate a CSV file with results from testing a

nicetest.c

```
#include <errno.h>
#include <fcntl.h>
#include <signal.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/resource.h>
#include <sys/time.h>
#include <sys/wait.h>
#include <unistd.h>

#define PROG "nicetest"

// these are for generating the CSV; time is constant and num_proc is varied
// from 2 to PROC_MAX
#define TEST_TIME 5
#define PROC_MAX 16

// somewhat relaxed error checking, as this program is exploratory and not
// meant to be rigorously user-tested (i.e., user-abused)
#define ERR_FAT(cmd, ctx, msg) {\
  /*check errno b/c some syscalls (e.g., nice) may have a non-err neg retval*/\
  if(cmd<0 && errno) {\
    dprintf(2, "%s: %s: %s\n", PROG, ctx, msg);\
    kill(0, SIGKILL); /*kill other children if called from child*/\
    exit(EXIT_FAILURE);\
  }\
}
#define ERRNO_FAT(cmd, ctx) ERR_FAT(cmd, ctx, strerror(errno));
#define ERRMSG_FAT(cond, msg)\
  if(cond) {\
    dprintf(2, "%s: %s\n", PROG, msg);\
    exit(EXIT_FAILURE);\
  }

double run_nice_test(int num_proc, int nice_val, int test_time) {
  int i, wstatus, pid, ndn_pid; // non-default nice pid
  long tot_usec, ndn_usec, tmp_cum_usec;
```

```
static long cum_usec;         // cumulative time (since rusage is per-proc)
double tim_pcnt;
struct rusage ndn_ru, tot_ru;

for(i=0; i<num_proc; i++) {
  switch(pid=fork()) {
  case -1:
    dprintf(2, "err fork\n");
    exit(EXIT_FAILURE);
    break;
  case 0:
    signal(SIGTERM, SIG_DFL);
    if(!i) {
      errno = 0;
      ERRNO_FAT(nice(nice_val), "setting nice value");
    }
    while(1);
  default:
    if(!i)
      ndn_pid = pid;
  }
}

// since this is called after forks began, total time may be slightly larger
// than expected total time (i.e., (# processors)*(expected time))
ERRNO_FAT(-sleep(test_time), "sleep-ing for test_time seconds");

// ignore sigterm in parent
signal(SIGTERM, SIG_IGN);

// get ndn pid; unfortunately, may cause this to exit slightly earlier than
// others; forgot about existence of wait4 and should have used that in
// hindsight
ERRNO_FAT(kill(ndn_pid, SIGTERM), "kill ndn child with SIGTERM");
ERRNO_FAT(waitpid(ndn_pid, &wstatus, 0), "waitpid for ndn child");
ERRNO_FAT(getrusage(RUSAGE_CHILDREN, &ndn_ru), "getrusage for ndn child");

// kill rest of children and get rusage
// kill 0 sends signal to everything in process group
ERRNO_FAT(kill(0, SIGTERM), "kill non-ndn child");
for(i=1; i<num_proc; i++)
  ERRNO_FAT(wait(&wstatus), "wait for non-ndn child");
ERRNO_FAT(getrusage(RUSAGE_CHILDREN, &tot_ru), "getrusage for non-ndn child");

tmp_cum_usec = cum_usec;
```

```
  ndn_usec = (ndn_ru.ru_utime.tv_sec+ndn_ru.ru_stime.tv_sec)*1000000
              +ndn_ru.ru_utime.tv_usec+ndn_ru.ru_stime.tv_usec-cum_usec;
  tot_usec = (tot_ru.ru_utime.tv_sec+ndn_ru.ru_stime.tv_sec)*1000000
              +tot_ru.ru_utime.tv_usec+ndn_ru.ru_stime.tv_usec-cum_usec;
  cum_usec = tmp_cum_usec+tot_usec;

  tim_pcnt = ((double)ndn_usec)/tot_usec*100;
  dprintf(2, "=====\n"
             "Num processes:\t%d\n"
             "Task0 nice val:\t%d\n"
             "Test time:\t\t%d\n"
             "Total CPU time:\t%ldus\n"
             "Task0 CPU time:\t%ldus\n"
             "Task0 CPU %:\t%lf%\n",
          num_proc, nice_val, test_time,
          tot_usec, ndn_usec, tim_pcnt);
  return tim_pcnt;
}

int nice_test_all() {
  int ofd, num_proc, nice_val, test_time;
  char buf[20];

  dprintf(2, "=====\nWriting output to niceout.csv\n");

  ERRNO_FAT((ofd=open("niceout.csv", O_WRONLY|O_TRUNC|O_CREAT, 0644)),
            "Opening outfile");

  strncpy(buf, "num_proc,", 10);
  // no need to check partial write cond
  ERRNO_FAT(write(ofd, buf, strlen(buf)), "Writing to outfile");
  for(nice_val=-20; nice_val<20; nice_val++) {
    sprintf(buf, "%d,", nice_val);
    ERRNO_FAT(write(ofd, buf, strlen(buf)), "Writing to outfile");
  }
  strncpy(buf, "\n", 2);
  ERRNO_FAT(write(ofd, buf, strlen(buf)), "Writing to outfile");

  for(num_proc=2; num_proc<=PROC_MAX; num_proc++) {
    sprintf(buf, "%d,", num_proc);
    ERRNO_FAT(write(ofd, buf, strlen(buf)), "Writing to outfile");

    for(nice_val=-20; nice_val<20; nice_val++) {
      sprintf(buf, "%.5f,", run_nice_test(num_proc, nice_val, TEST_TIME));
      ERRNO_FAT(write(ofd, buf, strlen(buf)), "Writing to outfile");
```

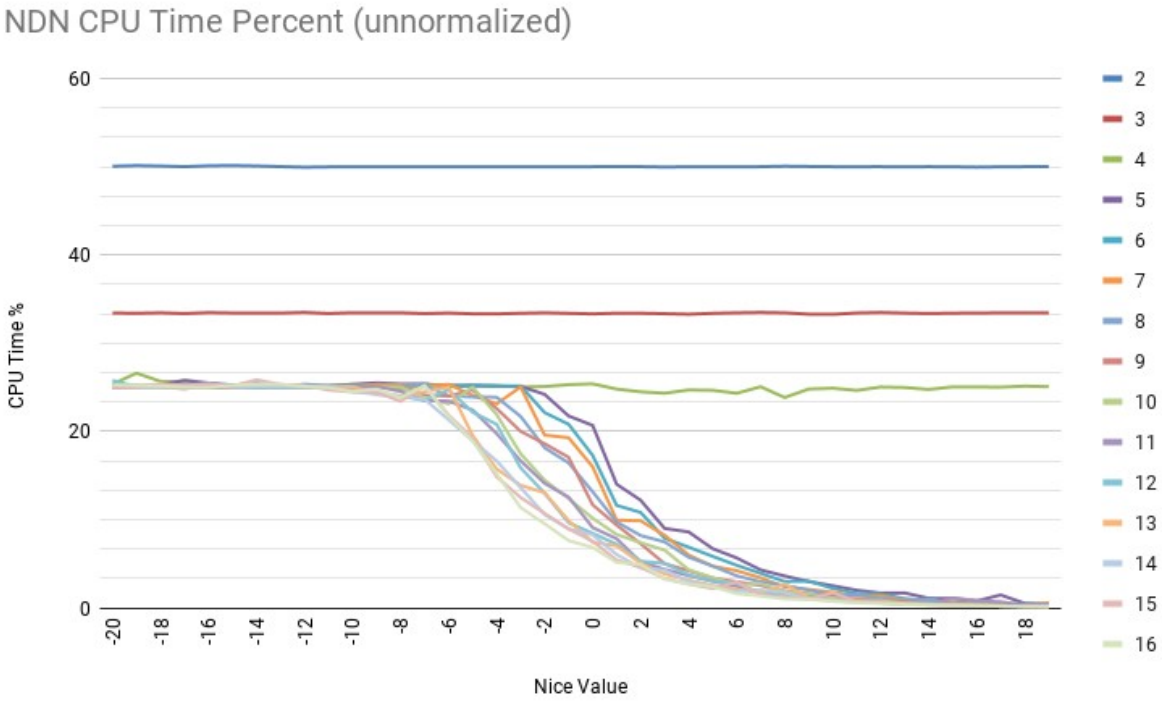
```
    }

    strncpy(buf, "\n", 2);
    ERRNO_FAT(write(ofd, buf, strlen(buf)), "Writing to outfile");
  }

  ERRNO_FAT(close(ofd), "Closing outfile");
  exit(EXIT_SUCCESS);
}

int main(int argc, char **argv) {
  // time in seconds, nice values from -20 to 19
  // nice values will be validated; time (in seconds) will only checked to be
  // positive but should be a value >5 to see discernible results
  int num_proc, nice_val, test_time;

  dprintf(2, "=====\nnicetest.c\n");

  // if called with no args, run series of tests and generate csv
  if(argc==1)
    return nice_test_all();

  // else just run single test with given params
  ERRMSG_FAT(argc<4, "Usage: nicetest [num_proc] [nice_val] [test_time]");
  ERRMSG_FAT((num_proc=atoi(argv[1]))<1,
             "num_proc must be a positive integer");
  ERRMSG_FAT((nice_val=atoi(argv[2]))<-20||nice_val>19,
             "nice number must be in range [-20,19]");
  ERRMSG_FAT((test_time=atoi(argv[3]))<1,
             "test_time must be a positive integer");

  run_nice_test(num_proc, nice_val, test_time);
}
```

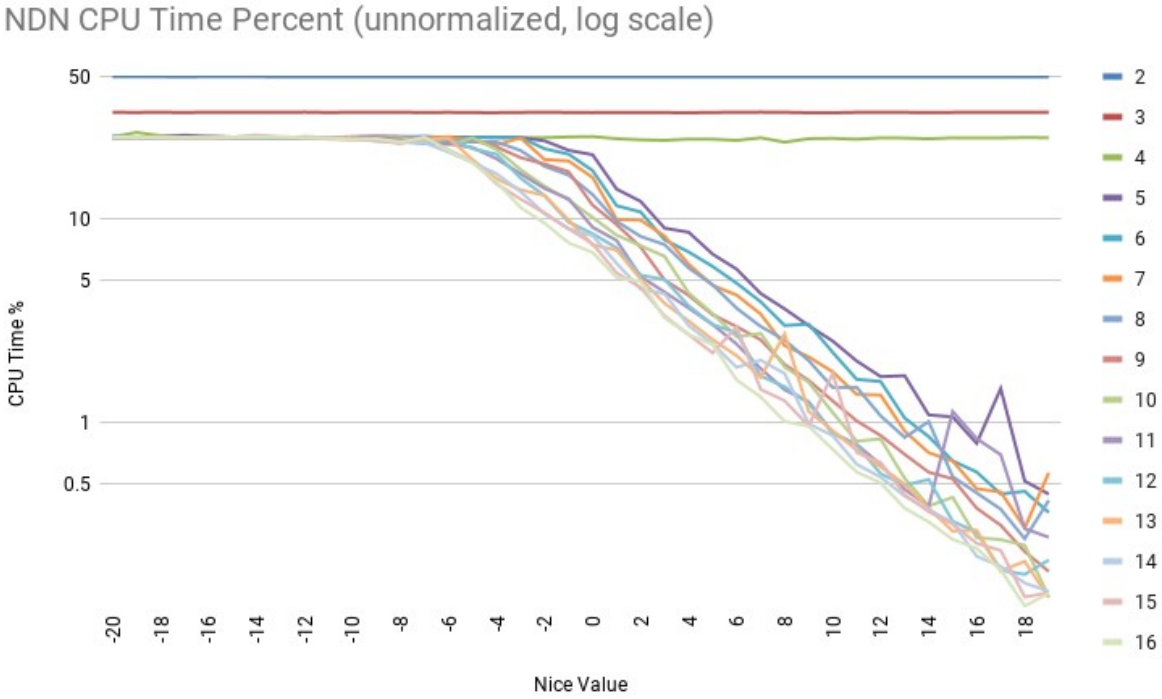
Test run outputs
Note: Test CPU: Intel i7 7500U (4 logical cores)
(usage: ./nicetest [num_proc] [nice_val] [test_time])

```
(base) [jon@archijon prog7]$ ./nicetest 16 0 5
=====
nicetest.c
=====
Num processes:    16
Task0 nice val:   0
Test time:        5
Total CPU time:   19250036us
Task0 CPU time:   1193019us
Task0 CPU %:      6.197490%
(base) [jon@archijon prog7]$ ./nicetest 16 10 5
=====
nicetest.c
=====
Num processes:    16
Task0 nice val:   10
Test time:        5
Total CPU time:   18173635us
Task0 CPU time:   137645us
Task0 CPU %:      0.757388%
```

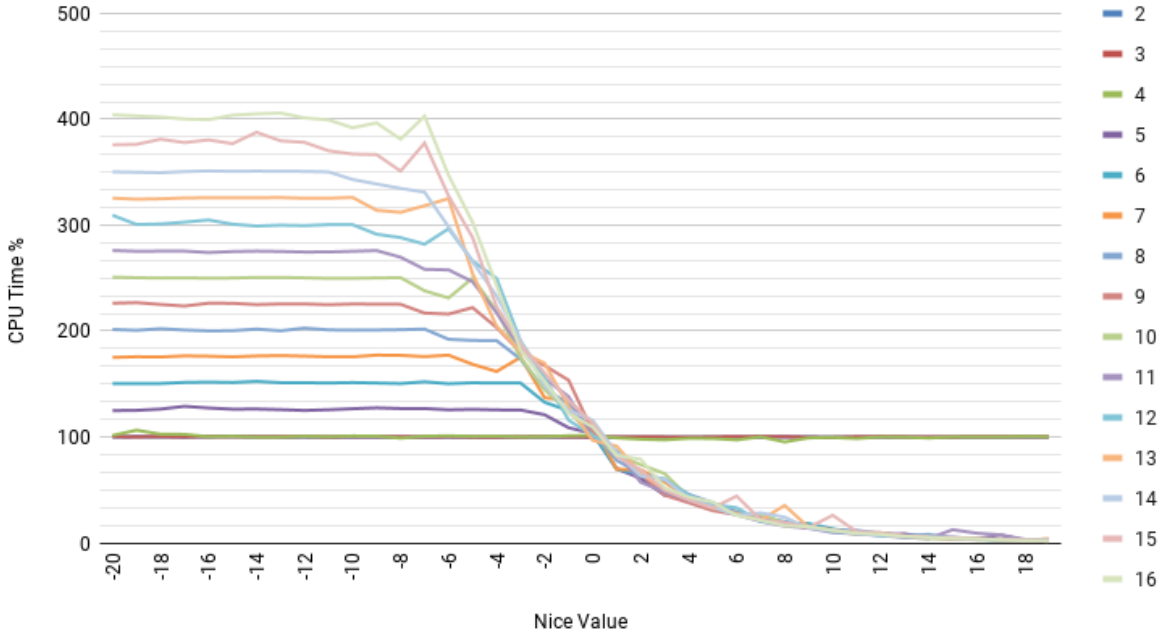
These are close to the demo’s values shown of 6.25549% and 0.69258%, respectively.

Test run outputs
(usage: ./nicetest)

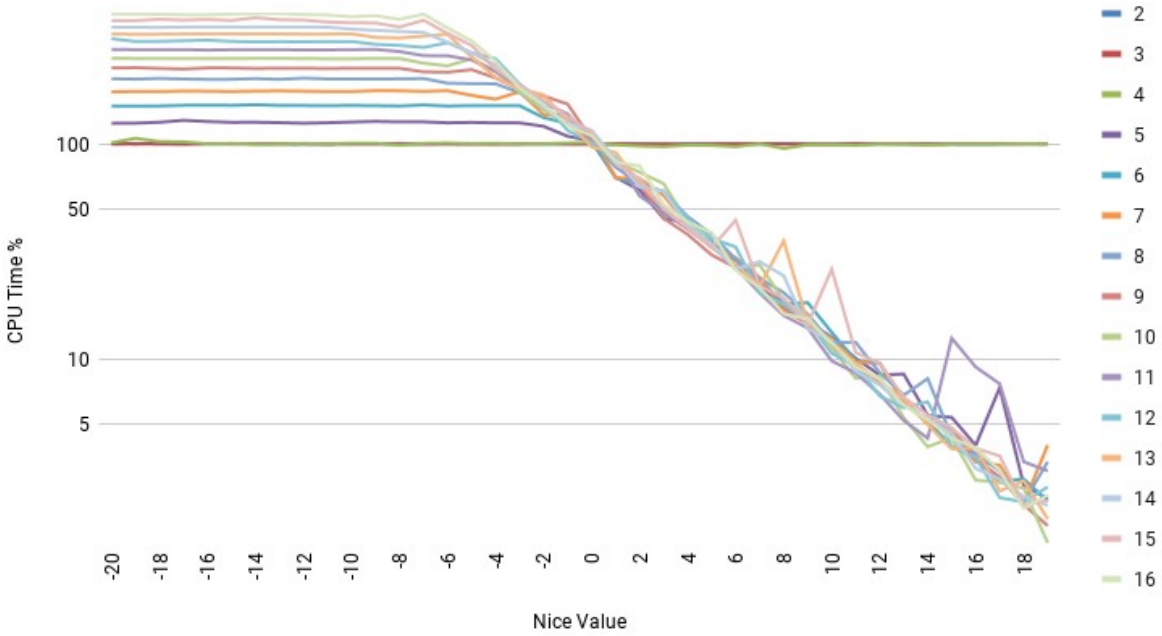




NDN CPU Time Percent (normalized)



NDN CPU Time Percent (normalized, log scale)



Explanation of the charts:

The graphs are plots of CPU time share (%) (CTS) vs. nice value of the non-default nice (NDN) process. Each line is a given value for number of processors (NP) used in the test. These are generated directly from Google Sheets by loading the generated niceout.csv file generated when calling niceout.c without any arguments.

The first two plots are non-normalized (i.e., the plotted CTS are exactly the CTS calculated as (NDN proc time)/(total CPU time for all processes)). In the latter two, they are normalized to the number of processors using spreadsheet magic (i.e., the plotted values are (NDN proc time)/(total CPU time for all processes)∗(NP)). Thus, the “100%” value in the normalized graphs indicates the CTS for one of N processes if all had the same nice value; anything higher means the process had more than average CPU share, and anything lower means the process had less. We can call this the “expected” CTS for one of N processes.

One noticeable aspect of the graphs is that for N∈[2,4], there is virtually no variation from the expected CTS value. This can be expected on a system with four logical cores (which the test environment has) that doesn’t have much else running (the test environment was relatively “quiet” other than the test) since little scheduling happens. Not much is really competing for CPU time other than these processes, so all get roughly the same amount of CPU time.

However, for any NP>5, there is a clear correlation between CTS and nice value. There appears to be a roughly-constant region until some negative cutoff nice value (we can denote this NCO, “nice cutoff” value). This cutoff value may be explained by the CPU allowing a minimum scheduling time for the other processes – after a certain point, it doesn’t allow the high-priority NDN process to monopolize the CPU. This can better be seen with the unnormalized plots: for NP>4 (when the nice values start affecting the CTS), we see that the maximum unnormalized CTS is capped at roughly 25%. (Given that this is a quad-logical core CPU, this makes sense – no process can be using more than one CPU simultaneously).

It appears that the drop is inverse exponential when nice value>NCO, as expected, asymptoting towards 0. The reason for this asymptote is obvious.

Study Guide for HSS3 Final Exam
Rose Auditorium, Monday, December 16, 12-12:50.

As noted on the syllabus, the final counts for 20% of your course grade. Keep in mind also that this is a common exam held during lecture time in the Rose Auditorium 12-12:50 on Monday December 16. However, your individual instructors will grade the final exam and determine your course grade. You may address questions about the exam to your instructors.

Students with accommodation arrangements for extra time on exams should report to Room LL 101. The room will be available from 12PM to 2PM.

The exam will include:
PART I. IDs (10 pts each/70% of total):
7 terms or quotations (from a choice of out of 12)

PART II. **One essay** question (30 points/30%), The essay question is cumulative and designed to let you address themes that have been discussed throughout the semester. You should refer to a minimum of three readings (your choice) that you have discussed in class in order to provide specific evidence for your argument(s).

Remember: The more you review your notes the easier the exam will be. Moreover, this is an intentionally long list and you certainly do not need to know everything to do well on the exam. You have a generous choice of IDs and the terms and quotes have all been referenced in lecture and/or discussed in class. For the IDs *briefly* identify who, what, where, and when and explain why each term or document quote is significant in the context of the course.

Here is a list of terms that you should be able to identify. Studying these terms will help you review for both the IDs and the short essay question.

Liberty, Equality, Fraternity
– useful b/c it represents enlightenment ideals, equality among all of the classes
– motto for the French revolution

Declaration of the Rights of Man
– aimed towards aiding bourgeoisie
– wanted general will to create politics, and that governs everyone (a.o.t. Church)
– resistance to opposition (clearly drawing a parallel to the American revolution)
– protects property (i.e., helps bourgeoisie)

Third Estate
- everyone who wasn't in the first or second estates (including bourgeoisie and the peasants)
– led the French Revolution, left the Estates General meeting and formed the National Assembly, which led uprisings such as the storming of the Bastille, the september massacre, women's march, execution of the king, and the turning into the Reign of Terror

Sans Culottes
– literally "without pants," the lower-class members of the French Revolution
– seen as more radical as they advocated for a direct democracy, as opposed to a representative republic with a parliament

– wanted complete social equality, price controls, showed the spirit of the French revolution – during the Reign of terror some aristocrats even clothed themselves like the sans culottes to avoid prosecution

Republic of Virtue
– Robespierre's philosophy for the society, which is imposed by the guillotine

Robespierre
– leader of the Jacobins, who were very radical
– put to death many of the counter-revolutionaries
– enforced «Republic of Virtue» for multiple years, put the King to death, was eventually put to death himself because he went overboard

Touissant L'Ouverture
– Helped fight for Haitian freedom against the French by using the Spanish help
– Slave revolt, the only one to cause the creation of a state
– Declared Liberty and Equality for the people in Saint Domingue

"Time, Work Discipline, and Industrial Capitalism"
– E.P. Thompson said that time measurements in society and politics is caused by the industrial revolution
– People's regular lives and everyday activities – even going to the bathroom, sleeping, or eating – are in accordance with certain times of day rather than with the typical views of time; e.g., New York city ball drop
– i.e., this was a capitalist construct to make sure that the capitalists earn a maximal amount of money

moral economy
– Also E.P. Thompson
– an economy based on fairness, justice, and goodness, rather than capitalist goals
– cultural economy, e.g., fair market trading a.o.t. free market (socialist goals)
– a type of Utopian dream – was not carried out and based on the idea that workers were always so barely subsisting

Manchesterism
– i.e., Manchester Liberalism, which evolved in Manchester
– Many liberal ideas, e.g., secularism, freedom of the press, anti-slavery, pacifism, free trade
– Repealed the Corn laws that held the cost of corn high – opposite of the moral economy – but argues in a different way – it argues that free trade will make more products available to all, which makes it beneficial to other people
– Workers in Manchester (during the industrial revolution) were hurt by the tariffs imposed by protectionism, wanted to decrease prices (note setting)

Laissez Faire
– Usually referred to capitalism and closely tied to economics.
– Questioned by socialists, who try to decrease the inequalities caused by capitalism via government control

Hungry Forties
– i.e., the Irish Potato Blight/Famine

– Huge emigration and deaths
– Exacerbated by government laissez-faire capitalism, so that its farmers had no help
– Became a part of the national identity
– Also Britain had a bad harvest, which exacerbated these effects

Revolutions of 1848
– February revolution in France – second French Republic – "right to work" and national assemblies for the unemployed – universal suffrage
– Sicily
– Austria, Germany, Italy, etc.
– series of bourgeois revolutions, democratic, liberal, asking for the removal of old political, monarchial structures
– food was prohibitively expensive, urban workers were angry
– Freedom of the press in Germany
– New constitution in Denmark, abolished absolute monarchy
– For the most part, mostly unsuccessful, but showed a major revolutionary moment and globalization

Treaty of Guadalupe Hildago
– Major political treaty between the U.S. and Mexico
– Mexico ceded 55% of their land to the U.S.
– Disrupted the slave-free state balance, opposed by Senate
– Manifest destiny and colonialism

The Communist Manifesto
– Communism is the natural and inevitable end result when it comes to bourgeoisieism
– Capitalism affects every part of society, and not in a good way
– Bourgeoisie is so revolutionary that it is destroying itself.
– The idea of capital is something that is meant to earn profit by exploiting workers – founding principles of Marxism

Potato Famine
– see notes above for Hungry Forties

Bismarck
– German chancellor who managed the unification of Germany through the Franco-Prussian war
– Also sided with the working class with a welfare state, universal male suffrage

Cavour
– A key leader in Italian unifcation, but did not live long to see much of it
– Somewhat dictatorial

Franco-Prussian War
– a.k.a., War of 1870
– Prussian ambitions to unify Germany with a common goal, French fears of the Germans doing exactly that
– Part of a long series of conflicts between the French and German people

Second Industrial Revolution
– A great increase of steel and electricity

– Greatly improved transportation and communication (in particular, this is what allowed WWI to be so devastating)
– 1870s until the WWI, innovations allowed it to happen
– Great increase in urbanization
– Like the first industrial revolution, rise of working-class struggles, but now rise of Marxism (after 1848)

Social Imperialism
– Used by Lenin derogatively: "socialist in words, imperialist in deeds"
– Also: when governments engage in imperialism meant to preserve the domestic social peace
– Used derogatively towards Jews

Politics in a new key
– Relates to social democracy (connection to the U.S. and the "New Deal" – i.e., FDR and public works projects) – social interventions and justice in a democratic, capitalist framework
– SPD in Germany is one of the first Marxist-influenced parties in the world
– What is this article about?????

Dreyfus Affair
– a political scandal that lasted for a decade
– a symbol for modern injustice
– one anti-semitic case for zionism

Theodore Herzl
– father of zionism

Jack the Ripper scandal
– series of brutal murders in late 1800s in London
– media (writers) portrayed a serial killer, something that people began to believe, even though likely a hoax
– shows the growing influence of the media to influence people's thoughts and emotions

Scramble for Africa
– period known as "new imperialism" – 1880s-1914
– invasion, occupation, divison of Africa
– starting point is referred to as the Berlin conference
– mainly caused by European hostilities and competition, also for economic gain (imperialism)

Berlin Conference
– was in Berlin because of Germany's great rise in power during the second industrial revolution
– many European (and U.S.) world powers met to discuss how to regular activities in Africa (no African representatives)

The White Man's Burden
– a poem by Kipling
– critiques the colonial powers by showing the contradictions in the doctrine of imperialism, e.g., savage wars of peace, fill full the mouth of Famine, etc.

3 C's "Commerce and Christianity Make Civilization"

– ?????

Social imperialism
– lol this word was already here

Protectionism
– Tariffs to protect local industry
– Cause of some short-term economic crises (e.g., the great depression)
– Opposite of free-market ideals, more for fair-market and local producers

Organized Capitalism
– A product of long-term industrialization
– Organization of capitalism at the corporate and national levels; reinvestment/savings for future capacity
– away from the classical scale of small-scale capitalism

Boer War
– around 1900
– between the British Empire and South Africa
– union of South Africa after the Boers surrendered

Herero and Nama Rebellion/Genocide
– Precursor/prelude to the Holocaust
– Germany to the Namibia, totally torturing and inhumane methods of killing

Eugenics
– The idea of beliefs and practices to improve the genetic quality of a human population by excluding certain genetic groups judged to be inferior and promoting others
– relevance obvious

Social Darwinism
– application of natural selection and survival of the fittest to social and political aspects
– idea that the better people should be wealthier and more successful, and these people should be the ones to survive
– greatly associated with Nazi Germany, lost most popularity after WWII

Eiffel Tower
– 1889 World Fair for Paris

What is a Nation
– Renan
– Shared history and desire to live together, not geography or people or economy or war

Social Truce
– truce between parties in Germany during the war

Trench warfare
– trenches dug, made casualties extremely high, very low gain but high casualties
– devastating during the war

Spanish Flu
– major flu during WWI, made life more miserable

Shellshock
– PTSD, symptom first widely recorded from this war
– Wide impact on the war, and recollection of the war (many people didn't want to talk about the war (and same with WWI))

Triage
– in medicine? More effective way to help patients

Armenian Genocide
– systematic mass extermination of ethnic Armenians in Ottoman empire
– 1.5 million deaths over a decade
– coining of the word genocide
– nationalism – new rulership of "Young Turks" who wanted to "Turkify" the nation

Bolsheviks
– Leninist faction in Russia, Social Democratic Labour party (Marxist) – leaders of revolutionary working class in Russia

Bread, Land, and Peace
– Russians divided up land amongst themselves, withdrew from the war, and had some worker's rights laws instantiated
– Used by Lenin to win the support of the people

New Economic Policy
– (short NEP)
– proposed free market capitalism under state control
– partially revoked complete nationalization of industry

Socialism in one Country
– national communism within Russia, after seeing socialism fall in all western European nations
– a.o.t. Marxist view of global communism, but a practical move

Forced collectivization
– forcing farmers to collect into collective farms, and submit some grain so that prices wouldn't raise too high
– famine because prices were unobtainable and some bad harvests
– had terrible effect on Russians, especially in Ukraine, even causing people to resort to cannibalism, and 7 million Russians died because of it
– covered up to international viewers by Russian propaganda

Treaty of Versailles
– six months after the armistice signed (after 6 months of negotiation in Paris)
– War Guilt clause on Germany
– Major reparations for Germany to make
– Differing views – some people thought too harsh on Germany, France thought too lenient

Weimar Republic
– interwar German government founded by November Revolution shortly after the war – led by Socialist Republic party
– major reforms to workers' rights

Rosa Luxemburg
– radical leader of the Sparticus movement (more radical than the SPD)
– talked about the necessity of revolution and the downfall of capitalism

hyperinflation
– Germany had major ties with other world powers due to its physical and economic occupation, and succumbed to the Great Depression
– huge inflation – huge poverty in Germany and no way to pay off reparations

League of Nations
– part of Wilson's 14 points
– failed to recognize the genocides going on in the interwar periods, no power to enforce anything
– failed b/c the U.S. didn't join

Bauhaus
– "total work of art" – modernism
– there should be no distinction between form and function
– after the war, new revived liberal spirit

Rationalization
– rational doing terrible things

Fordism
– economic growth by mass production, social work
– raise wages

Taylorism
– improving economic efficiency, elimination of waste, etc.

My Workday, My Weekend
– everyday is just work, gendering of these ideas
– how men have a weekend off, not the same for women

March on Rome
– resulted in Mussolini's fascist party to take power in Italy
– fascist intimidation in order to legally rule
– laissez-faire promises, but didn't work out that way in the end, much nationalized

Mussolini
Italian fascism
– consolidating Italian power and expanding its land

Blackshirts

– members of Mussolini's fascist party

January 30, 1933
Enabling Act
– gave Hitler to enact laws without the Parliament

Gleichschaltung
– process of Nazification establishing a system of total control

Triumph of the Will
Nuremberg Laws
– racist laws, eugenics, etc.
– stripped Jews of their citizenship

Kristallnacht
– violence and destruction against jews

Operation Barbarossa
– German goal to conquer western Soviet Union in order to populate it with German people

"*Ordinary Men*"
– Opposition and diversity of many ordinary men in Germany who were fighting
– Older men, one Commander Trap set the precedent that a man could step away from the shooting if he wants to

Wannsee Conference
– The idea of the Final Solution is discussed

T4 Program
– involuntary euthanasia for non-perfect people (not only Jewish people)

Posen Speech
– evidence of German official responsibility for Jewish killings

"Alors…qui est responsable?" [Who then is responsible?]
– from Night and Fog
– who's responsible for the violence
– capos said we were just following orders – both persecuted and persecuted others

Atlantic Charter
– British and U.S. view after the conclusion of WWII, asking for disarmament, etc.
– created the UN

Four Freedoms
– freedom of speech, worship, from want, from fear
– before U.S. involvement in the war

Yalta Conference
– Second of three meetings of the Big Three after the war (others were Tehran and Potsdam)

– discuss the postwar reorganization of Germany and Europe

Zero Hour
– Germany to dissociate itself with pre-war memory
– much repair needed to the country, terrible strife
– some people say it's to avoid their punishment

Nuremberg Trials
– **military tribunal after WWII, major generals tried**

Displaced Persons
– camps set up throughout europe

Iron Curtain
– divided Europe into two following WWII due to Cold War

Marshall Plan
– U.S. attempting to help Europe rebuild economically after WWII, to prevent the same mistakes that happened after WWI

Declaration of Independence, Republic of Vietnam
– copies the rhetoric of the U.S. DOI, asking for independence
– doesn't say anything about communism, just asking for independence – yet the U.S. clearly opposes it, even though Ho Chi Minh believes that "the nations at Tehran will not object to this"

Nehru
– First prime minister of India, tutored by Gandhi, helped Indian independence revolutionary moment

Partition
– Two nation theory
– Divided India
– massive refugee crisis (a dozen million people displaced)
– huge riots, some violence

Genocide Convention
– UN will now officially recognize and protect against acts of Genocide
– Example breach in Rwanda in 1990s
– make the term legally actionable
– coined the term after WWII (after Nazi Germany)
– people have rights outside of states

Universal Declaration of Human Rights
– Document by the UN trying to instate natural rights as rule of law
– important b/c now the basic human rights can be said to be law from somewhere, for people to use as a model

Social Market Economy
– between laissez-faire capitalism and socialist economics
– combines fair market policies with free-market capitalism

Bandung Declaration
– meeting of Asian and African states to discuss colonialization (or at least the resistance against colonialism)
– non-aligned movement

Dien Bien Phu
Pieds Noirs
– French people born in Algeria during French rule, who returned to France after Algerian independence
– Double alienation – once from Algeria, once from France

FLN (*Front Liberation National)*
– National Liberation Front in Algeria (1950s)
– One-party state
– successfully won against France

1968
– Civil Rights movement in the U.S., opposition to Vietnam War all over the world, wildcat strikes in France, protests in Eastern Europe (Eastern Bloc) against lack of freedoms; also Brazil and Mexico
– a social revolution – not a political one – can happen even in a time of wealth
– against authority, illigitimate
– generational gap
– people want to have freedom/control over their own bodies

Aimé Césaire
Prague Spring
– January 1968
– against soviet puppet state

Thatcherism
Solidarity
– trade union in Poland
– rise in price of meat (price controls)
– not enough consumer economy, more national/industrial economy
– first Polish pope

Chernobyl
Gorbachev
Velvet Revolution
– Chekoslovakia independence in 1989, success after Prague Spring failed

November 9, 1989
– fall of Berlin War

Srebrenica
– genocide in 1990s
– things about genocide didn't end after WWII

Jonathan Lam
Prof. Fink
HSS 3-C
September 18th, 2019

(In response to the prompt: According to Marx in The Communist Manifesto, what made capitalism revolutionary?)

Second-class Citizens to Capital

Yes, the Communists intend to "do away with private property" (Marx 23); they want to "destroy the most hallowed of relations" (24), the family; they desire to remove such *freedoms* as free trade. But the notions the Communists intend to eradicate, claim Marx, are only the *addictive* bourgeois image imposed onto the world, notions whose real representations are already stolen from the proletariat.

Due to their great influence, the bourgeois industry leaders stripped the "motley feudal ties that bound man to his 'natural superiors', and … left remaining no other nexus between man and man than … than callous 'cash payment'" (16). This fundamental change in societal values turns money into the first-class citizen of the bourgeois era (hierarchal feudal relations were arguably very restrictive, but are described as "motley" relative to money relations, for which there is "no other nexus" between people), which in turn sets the stage for the disparity between *capital* and *wage money*; the introduction of mass capital and its conflation with wage earnings is an addicting, revolutionary, and dangerous aspect of capitalism.

It is important to discuss what this *capital* is and how it differs from *money*. Marx defines capital as:

> "… that kind of property which exploits wage-labour, and which cannot increase except upon condition of begetting a new supply of wage-labour for fresh exploitation" (22).

In a broader sense, capital comprises property that grows in value. Marx declares that the source of revenue is the difference between the wage-labourer's labour and wage – i.e., "exploitation" of the proletariat. He claims next that capital is "a collective product"; i.e., it belongs only to the general populace which generates it. The *capitalist*, then, accumulates the capital and hands out only the minimal wages for subsistence of the working class. The latter is plain *wage money,* not suitable for growth.

The transformation of power from "natural superiors" to purely money relations has a series of implications. Firstly, bourgeois society is tightly coupled with industrialization and globalization, both generating an enormous workforce suitable for generating capital, resulting in more production in a century than in the rest of history (17). There is

also the primitive sense of (economic) freedom, as everyone earns money; however, like the success of production, this is an optimistic view caused by conflation of capital and minimum wage, and wage-labourers don't have the freedom for innovative success afforded to the bourgeoisie with sufficient capital.

As opposed to a feudal society, in which there are well-defined relationships and responsibilities determined between the social classes, the distinction is more subtle because both classes earn money despite the disparity in economic capability of minimum wage and accumulation of capital (thus hiding the inequality). This is why Marx warns that the reader's "ideas are the outgrowth of the conditions of [the] bourgeois production and bourgeois property" (24); thus, the concepts of "progress," "property," "freedom," and even "family" are capitalist notions whose realities are not admitted to the proletariat, and eliminating the bourgeois versions (i.e., the "class character" (23)) removes only the embedded inequality.

Bibliography

Marx, Karl, and Friedrich Engels. "The Communist Manifesto." *Marxists Internet Archive*, 2010, pp. 14–27.

Jonathan Lam
Prof. Fink
HSS 3-C
October 7th, 2019

The Perpetrators and Innocents among Revolutionaries

The background of the three women interviewed in “Women Testify Concerning Their Participation in the October Days (1789)” do not belong to revolutionary leaders. They are ordinary, working-class citizens – a cleaning woman, a seamstress, and a lace-worker, early-adults to middle-aged women – forced to join this march by a group of other women, whose purpose was, supposedly, “to ask for bread at Versailles.” These depositions come from an investigation into the October Days, and these examples demonstrate clearly that the majority of the revolt are unassuming commoners led by a few radical leaders.

The detail of being forced into the march is identical for the three accounts, and likely for many other women, for they were forced, in the words of Mrs. Glain and Mrs. Delaissement, “as many other women were,” and, according to Ms. Barre, by reason of “not being able to resist this great number of women” – a clear display of the persuasive power of the masses, even unarmed.

There’s a superficial reiteration of the phrase “to ask for bread” – it appears once in Mrs. Glain’s deposition, and is used by the protesters several times in Ms. Barre’s deposition, first to her, then to the duc de Guiche, then to M. de Saint Priest, and then to King Louis XVI. It was a simple and virtuous claim that attracted the hundreds of women and gained them audience to the National Assembly and to the king; and the king, knowing the importance of feeding his citizens, delivers a response that is, in Ms. Barre’s deposition, interpreted to be a genuine willingness to help, to which the women exclaim in joy, “vive Le Roi!” This cry also signifies that those exclaiming women do not direct their anger irrationally towards the king.

But what appears to be a good-natured plea for bread is blackened in Mrs. Delaissement’s deposition, with protestors initially armed for a violent encounter with makeshift weaponry. She notices Maillard, one of the prominent National Guardsmen to storm the Bastille in the infamously bloody conflict only three months earlier, and who surely knows about the potential for violence from such a protest. There are also fiery individuals, such as the prostitute who states she was “going to Versailles to bring back the queen’s head,” and later physically confronted a Royal Guardsman. While Mrs. Glain’s deposition stated that this protestor was “sharply reproached” for her death threat to the queen, she wasn’t alone – the queen narrowly escaped when the protestors aimed to capture and kill her in the palace upon their arrival.

There is also the animosity against the Royal Guards. The Flanders soldier warns Mrs. Delaissement of the Royal Guards, “who, during a meal, had trampled the

national cockade," a symbol of the National Guard and of the revolution. Mrs. Glain reports that the crowd kills and beheads two Royal Guardsmen. This happens after the king had agreed to the women's requests for bread security, so it was not a necessary action for the women to get into the palace to deliver their message to the king, nor was decapitation after their deaths a protective measure; thus suggests an ulterior motive for the march to Versailles.

Under the guise of asking for bread, the Women's March was a multifaceted success. Not only did it achieve royal protection for Parisian bread supplies, but the women were able to intimidate the Royal Guards, almost murder the queen, and relocate the monarch to Paris. The latter two motives aren't even recognized by the three women, who are only exposed to the former.

This major event in the Revolution foreshadows a great deal of the radicalization to come in the French Revolution. Unlike the storming of the Bastille earlier in the year, which was carried out by a militia force, these are ordinary citizens. (And not only ordinary citizens – working class *women*.) The fact that ordinary women could persuade the king and kill soldiers was surely an inspiring feat and a demonstration of the strength of the mob against authority.

This isn't to say that the whole population mobilized towards revolution. Many of the mob storming Versailles, such as the three women of focus, were relatively innocent in that they only wanted fairer prices for bread and were coerced into joining the march – mobilization of innocent, working-class citizens towards some common goal allows them to achieve greater political goals. Like the rise of the bourgeois class and industrialization, a few French revolutionaries were trying to enact change at such a rate that they themselves are surprised by the extent of the effects, foreshadowing a higher level of radicalism such as that that would appear in the Reign of Terror.

Works Cited

"Women Testify Concerning Their Participation in the October Days (1789)," *Liberty, Equality, Fraternity*, accessed October 7, 2019, http://chnm.gmu.edu/revolution/d/474.

Jonathan Lam
Prof. Fink
HSS 3-C
November 15th, 2019

Woman and Machine: The Changing Face of Industrialization

Replace a worker with a machine -- what does it look like? In a new surge of industry fostered by a wartime need and a postwar consumer demand, Hoch presents a colorful image of the contemporary worker in her painting, *Study for Man and Machine*. A worker, but mechanical, but with a face. Her artwork is an acceptance of the degrading changes to the German workforce; a similar view (albeit with an altogether different presentation) is presented in a very human photograph around the same era.

Take away the worker, and there's still a machine, a brass- and silver-colored tool with screws and edges and knobs. This is the conventional tool. The background is grey, the mechanical tool is faded. Put back the worker, and there is a spring for an arm and shutters for eyes, the yellow and rust-colors of the machine's facade resemble human skin and blood, and even the background behind the worker is a lighter shade than that surrounding it. It's a little difficult to tell what is morphing. Is the worker becoming a machine, or is the machine becoming more human?

Both are certainly plausible, given the massive increase in industrialization during and after the Great War, which wreaked an immense toll on the German industrial workforce. Some of the major outcomes included the following general outcomes:

- the highest throughput of German industrial machines to feed the war industry;
- a great reduction of the workforce both during and after the war, as many young men left to fight during the war and many never returned;
- terrible conditions for German workers, who were miserable and dying from overwork, starvation, Spanish flu, and some of course to the wartime violence.

The latter two consequences culminated in larger effects directly after the war. Women played a more important role in society and the workforce during the war due to the lack of male workers, and after the war they were indignant to return to their previous state. Hoch was one such outspoken feminist (Gallagher). Similarly, workers fought for better working conditions after the war, which led to the foundation of the Weimar Republic by a social revolution only a few months after the war. While not directly a champion of worker's rights and not an industrial worker herself, Hoch's work is associated with the Dadaist art movement, which strongly opposed the capitalistic exploitation of the working force. Both led to substantial leftist movements following the war, namely women's suffrage in 1919 and dramatic labor reform shortly following the foundation of the Weimar Republic.

Perhaps what is most important and striking about the mechanical robot is that it has a face. With Hoch's involvement in the feminist movement, and with the sharp, colorful contrasts like makeup, the mechanical worker has some semblance of a female face. It doesn't fit into the surroundings, into the traditionally-male role of the breadwinner, but in a particularly vibrant way.

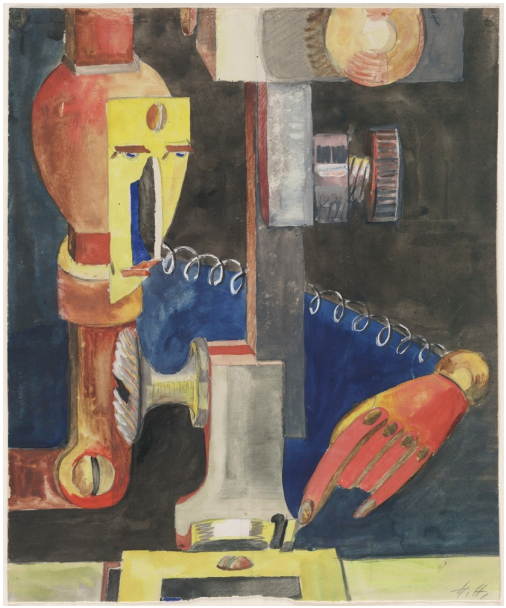
However, the placard brings up the unnaturalness of this machine: “How functional is this machine after all?” Throw in the unconventional painting media – gouache, pencil, and watercolor – and you have quite the absurd being, of which placing in an industrial setting is both machine-ly un-human and human-ly inefficient. Thus the feminist interpretation is an optimistic one, hoping instead to combine the advantages of the human and machine parts; its colors are breaking out of the mundaneness of the background, a representation of the victories of liberal movements for workers and women that accompanied the end of the terrible war and its working conditions. While the worker is certainly not entirely human, she remains vivacious.

Contrast this to the photograph of the carpenter from the Red Cross Institute for Crippled and Disabled Men. Clearly there is no color photography by this time, but the lack of color and the very ordinary setting of the work set a tone opposite to the whimsical nature of the awkward mechanical worker. There is no more human mask, no silly colors. There is still the man and the machine, however, and now the machine may be removed from the man: as the handwritten note suggests, the man uses his arm as a tool chuck. The view of this image is, expectedly, somewhat different than Hoch's; rather than provoke a sense of absurdity in the changing workforce, this is more illustrative of the poor state of the working class that survived the war: injured and incomplete, but hopeful. The hand may be the most important tool to the carpenter or the worker of another trade, and yet even in its absence the worker survives. As the caption also notes, “on Sundays he has an arm made in careful imitation of the one he has lost” – even his most mechanical aspect still has a human character, a character that persists outside of his work (on Sundays).

Hoch was a Red Cross volunteer briefly during the war (Gallagher). It's likely that she met many women and men like the carpenter both during and after the war, finding hope in their successful struggles in the workplace that continued after the war.

References: Works from MoMA, Room 510, “Man and Machine”

Hannah Hoch: *Study for Man and Machine (Skizze zu Mensch und Maschine)*. 1921. https://www.moma.org/collection/works/33945.



Placard text: “An ideal worker might be entirely mechanical. This idea inspired advocates and worried critics of mechanization since the advent of the Industrial Age. This watercolor renders the merging of human and machine in a cacophonous, colorful modernity, depicting a laborer composed of springs, gears, and assembled parts not so different from the apparatus at which they toil. Still, the design is hardly streamlined, both body and tool feature awkward knobs and extraneous corners. How functional is this thing after all?”

Red Cross Institute for Crippled and Disabled Men. *French War Veteran* c. 1919. https://www.moma.org/collection/works/58679

Handwritten text: “For a working man, this carpenter wears a chuck into which he inserts any tools he may need. On Sundays he has an arm made in careful imitation of the one he has lost.”

Works Cited

Hoch, Hannah. *Study for Man and Machine*. Artists Rights Society (ARS), Germany, 1921, MoMA, New York.

Red Cross Institute for Crippled and Disabled Men. *French War Veteran*. New York, 1919.

Gallagher, Paul. "The Mama & the Dadas: The Pioneering Feminist Artwork of Hannah Höch." *Dangerous Minds*, Dangerous Minds, 13 Nov. 2018, https://www.dangerousminds.net/comments/the_mama_the_dadas_the_pioneering_feminist_artwork_of_hannah_hoech.

MA326 – CH4-6 Final Review

Jonathan Lam

December 9, 2019

4.1. Determinants of order 2

DEF. 2×2 **determinant** If

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in M_{2\times 2}(F)$$

then we define the **determinant** of A, denoted $\det(A)$ or $|A|$ to be the scalar $ad - bc$.

THM 4.1. The function $det : M_{2\times 2}(F) \to F$ is a linear function of each row of a 2×2 matrix when the other row is held fixed. That is, if $u, v, w \in F^2$ and k is a scalar, then

$$\det \begin{pmatrix} u + kv \\ w \end{pmatrix} = \det \begin{pmatrix} u \\ w \end{pmatrix} + k\det \begin{pmatrix} v \\ w \end{pmatrix}$$

(and the same for the other row.)

THM 4.2. Let $A \in M_{2\times 2}(F)$. Then the determinant of $A \neq 0 \iff A$ invertible. If A invertible, then

$$A^{-1} = \frac{1}{\det(A)} \begin{pmatrix} A_{22} & -A_{12} \\ -A_{21} & A_{11} \end{pmatrix}$$

Other results

Area of a parallelogram The determinant is equal to the area of a parallelogram (why?) formed by (u_1, u_2) and (v_1, v_2) is

$$\begin{vmatrix} u_1 & u_2 \\ v_1 & v_2 \end{vmatrix}$$

This works for n-dimensional volumes of n-dimensional parallelpipeds.

4.2. Determinants of order n

DEF. higher-order determinant by cofactor expansion along the first row Let $A \in M_{n\times n}(F)$. If $n = 1$, then $|A| = (A_{11})$. See (THM. 4.4.) with $i = 1$. (This is not the most useful/general form.)

THM 4.3. This is an extension of (THM 4.1.) to higher-order matrices.

Corollary If a square matrix A has a row consisting entirely of zeroes, then $\det(A) = 0$.

THM 4.4. The determinant of a square matrix can be evaluated by cofactor expansion along any row i.

$$\det(A) = \sum_{j=1}^{n} (-1)^{i+j} A_{ij} \det(\tilde{A}_{ij})$$

Corollary If $A \in M_{n\times n}(F)$ has two identical rows, then $\det(A) = 0$.

THM 4.5. Swapping two rows means flipping the sign of the determinant.

THM 4.6 Adding a multiple of one row to another doesn't change the determinant.

Corollary If $A \in M_{n\times n}(F)$ has rank less than n, then $\det(A) = 0$.

Other results

Determinant of an upper-triangular matrix The determinant of an upper-triangular matrix is the product of its diagonal entries.

4.3. Properties of determinants

THM 4.7. $\forall A, B \in M_{n\times n}(F)$, $\det(AB) = \det(A) \cdot \det(B)$.

Corollary A matrix $A \in M_{n\times n}(F)$ is invertible IFF $\det(A) \neq 0$. If A invertible, then $\det(A^{-1}) = \frac{1}{\det(A)}$.

THM. 4.8. $\forall A \in M_{n\times n}(F)$, $\det(A) = \det(A^T)$. This means that cofactor expansions can occur along any row or column (generalizing (THM 4.4.) even further).

THM 4.9. (Cramer's rule) Let $Ax = b$ be a linear system in n unknowns. If $\det(A) \neq 0$, then $x_k = \frac{\det(M_k)}{\det(A)}$., where M_k is the matrix obtained by replacing column k of A by b.

4.4. Summary – important facts about determinants

(Not much new is introduced in this section.)

- The determinant of similar matrices is the same.

5.1. Eigenvalues and Eigenvectors

DEF. diagonalizable A linear operator T on a finite-dimensional v.s. V is called **diagonalizable** if there is an O.B. β for V s.t. $[T]_\beta$ is a diagonal matrix. A square matrix A is called **diagonalizable** if L_A is diagonalizable.

DEF. e-val and e-vect Let T be a linear operator on a v.s. V. A nonzero vector $v \in V$ is called an **eigenvector** of T if $\exists \lambda \in F : T(v) = \lambda v$. The scalar λ is called the **eigevalue** corresponding to the eigenvector v.

THM 5.1. A linear operator T on a finite-dimensional v.s. V is diagonalizable IFF there exists an O.B. for V consisting of eigenvectors of T. Furthermore, if T is diagonalizable, $\beta = \{v_1, v_2, \ldots, v_n\}$ is an O.B. of e-vects of T, and $D = [T]_\beta$, then D is a diagonal matrix and D_{jj} is the e-val corresponding to v_j for $1 \leq j \leq n$,

THM 5.2. Let $A \in M_{n\times n}(F)$. Then a scalar λ is an e-val of A IFF $\det(A - \lambda I_n) = 0$.

DEF. characteristic polynomial Let $A \in M_{n\times n}(F)$. The polynomial $f(t) = \det(A - tI_n)$ is called the **characteristic polynomial** of A. (Similar for T – characteristic polynomial is the characteristic polynomial of $[T]_\beta$).

THM 5.3. Let $A \in M_{n\times n}(F)$.

- The characteristic polynomial of A is a polynomial of degree n with leading coefficient $(-1)^n$.
- A has at most n distinct eigenvalues.

THM 5.4. Let T be a linear operator on a v.s. V, and let λ be an eigenvalue of T. A vector $v \in V$ is an e-vect of T corresponding to λ IFF $v \neq 0$ and $v \in N(T - \lambda I))$.

5.2. Diagonalizability

THM 5.5. Let T be a linear operator on a vector spacce V, and let $\lambda_1, \lambda_2, \ldots, \lambda_k$ be distinct e-vals of T. If $v_1, v_2, \ldots, v_k$ are e-vects of T s.t. λ_i corresponds to $v_i (1 \leq i \leq k)$. then $\{v_1, v_2, \ldots, v_k\}$ is linearly independent.

Corollary Let T be a linear operator on an n-dimensional v.s. V. If T has n distinct e-vals, then T is diagonalizable.

THM 5.6. The characteristic polynomial of any diagonalizable linear operator splits.

DEF. Let λ be an e-val of a linear operator or matrix with characteristic polynomial $f(t)$. The **(algebraic) multiplicity** of λ is the largest positive integer k is the largest positive integer k for which $(t-\lambda)^k$ divides $f(t)$.

DEF. e-space Let T be a linear operator on a v.s. V, and let λ be an e-val of T. Define $E_\lambda = \{x \in T : T(x) = \lambda x\} = N(T - \lambda I_V)$. The set E_λ is called the **eigenspace** of T corresponding to the e-val λ.

THM. 5.7. Let T be a linear operator on a finite-dimensional v.s. V, and let λ be an eigenvalue of T having multiplicity m. Then $1 \leq \dim(E_\lambda) \leq m$.

THM 5.8. Let T be a linear operator on a v.s. V, and let $\lambda_1, \lambda_2, \ldots, \lambda_k$ be distinct e-vals of T. For each $i = 1, 2, \ldots, k$, let S_i be a finite linearly independent subset of the eigenspace E_{λ_i}. Then $S = S_1 \cup S_2 \cup \cdots \cup S_k$ is a linearly independent subset of V.

THM. 5.9. Let T be a linear operator on a finite-dimensional v.s. V s.t. the characteristic polynomial T splits. Let $\lambda_1, \lambda_2, \ldots, \lambda_k$ be the distinct eigenvalues of T. Then

1. T is diagonalizable IFF the multiplicity of λ_i is equal to $\dim(E_{\lambda_i})$ for all i.
2. If T is diagonalizable and β_i is an O.B. for E_{λ_i} for each i, then $\beta = \beta_1 \cup \beta_2 \cup \cdots \cup \beta_k$ is an O.B. for V consisting of e-vects of T.

DEF. sum of subspaces Let $W_1, W_2, \ldots, W_k$ be subspaces of a v.s. V. We define the **sum** of these subspaces to be the set

$$\{v_1 + v_2 + \cdots + v_k : v_i \in W_i \ \forall 1 \leq i \leq k\}$$

which we denote by $\sum_{i=1}^k W_i$.

DEF. direct sum of subspaces Let $W_1, W_2, \ldots, W_k$ be subspaces of a v.s. V. We call V the **direct sum** of the subspaces of $W_1, W_2, \ldots, W_k$ and write $V = \bigoplus_{i=1}^k W_i$ if V is the sum of those subspaces and

$$W_j \cap \sum_{i \neq j} W_i = \{0\} \ \forall 1 \leq j \leq k$$

THM 5.10. Let $W_1, W_2, \ldots, W_k$ be subspaces of a finite-dimensional v.s. V. The following conditions be equivalent.

- V is the direct sum of subspaces $\{W_i\}$.

- $V = \sum_{i=1}^{k} W_i$, and for any vectors $v_1, v_2, \ldots, v_k$ s.t. $v_i \in W_i$, if $v_1 + v_2 + \cdots + v + k = 0$, then $v_i = 0 \ \forall i$.
- Each vector $v \in V$ can be uniquely written as $v = v_1 + v_2 + \cdots + v_k$, where $v_i \in W_i$.
- If γ_i is an O.B. for W_i, then $\gamma_1 \cup \gamma_2 \cup \cdots \cup \gamma_k$ is an O.B. for V.
- For each $i = 1, 2, \ldots, k$, there exists an O.B. γ_i for W_i s.t. $\gamma_1 \cup \gamma_2 \cup \cdots \cup \gamma_k$ is an O.B. for V.

THM 5.11. A linear operator T on a finite-dimensional v.s. V is diagonalizable IFF V is the direct sum of the eigenspaces of T.

5.4. Invariant subspaces and the Cayley-Hamilton theorem

DEF. T-invariant subspace Let T be a linear operator on a v.s. V. A subspace W of V is called a $T - invariantsubspace$ of V if $T(W) \subseteq W$, that is, if $T(v) \in W \ \forall v \in W$.

DEF. T-cyclic subspace Let T be a linear operator on a v.s. V, and let x be a nonzero vector in V. The subspace $W = \text{span}(\{x, T(x), T^2(x), \ldots\})$ is called the **T-cyclic subspace** of V **generated by** x.

THM 5.21. Let T be a linear operator on a finite-dimensional v.s. V, and let W be a T-invariant subspace of V. Then the characteristic polynomial of T_W divides the characteristic polynomial of T.

THM 5.22. Let T be a linear operator on a finite-dimensional v.s. V, and let W denote the T-cyclic subspace of V generated by a nonzero vector $v \in V$. Let $k = \dim(V)$. Then

1. $\{v, T(v), \ldots, T^{k-1}(v)\}$ is a basis for W.
2. If $a_0 + a_1 T(v) + \cdots + a_{k-1} T^{k-1}(v) + T^k(v) = 0$, then the characteristic polynomial of T_W is $f(t) = (-1)^k (a_0 + a_1 t + \cdots + a_{k-1} t^{k-1} + t^k)$.

THM 5.23. Cayley-Hamilton. Let T be a linear operator on a finite-dimensional v.s. V, and let $f(t)$ be the characteristic polynomial of T. Then $f(T) = T_0$, the zero transformation. That is, T "satisfies" its characteristic equation.

Corollary Let $A \in M_{n \times n}(F)$, and let $f(t)$ be the characteristic polynomial of A. Then $f(A) = 0$.

THM 5.24. Let T be a linear operator on a finite-dimensional v.s. V, and suppose that $V = W_1 \bigoplus W_2 \bigoplus \cdots \bigoplus W_k$, where W_i is a T-invariant subspace of V for each i $(1 \leq i \leq k)$. The characteristic polynomial of V is the product of the characteristic polynomials of the T-invariant subspaces.

DEF. direct sum of matrices Direct sum of matrices eww

THM 5.25. Let T be a linear operator on a finite-dimensional v.s. V. Let V be a direct sum. Then for $A = [T]_\beta$, $B_i = [T_{W_i}]_{\beta_i}$, then A is the direct sum of $\{B_i\}$.

MA326 Linear Algebra – Definitions, Theorems, and other Results

Jonathan Lam

November 11, 2019

Contents

1 Appendix C: Fields

DEF binary operation A binary operation is a mapping of two elements of the same set to the same set, i.e., $\text{op}_{\text{bin}}(x_1, x_2) : S \times S \to S$.

DEF field A field F is a set on which the binary operators addition ($+$) and multiplication ($\cdot$) are defined, such that addition and multiplication are closed on the set and the five field axioms apply:

1. Commutativity of addition and multiplication
2. Associativity of addition and multiplication
3. Existence of (distinct) identity elements for addition (0) and multiplication (1)
4. Existence of inverse elements for addition and multiplication (except with 0 for multiplication)
5. Distributivity of multiplication over addition

THM C.1 (Cancellation Laws for Fields) For arbitrary elements $a, b \in F$, the following statements are true:

- $a + b = c + b \Rightarrow a = c$
- $a \cdot b = c \cdot b \Rightarrow a = c$

THM C.1 COR 1: Uniqueness of identity and inverse elements (Proof straightforward)

DEN: additive and multiplicative inverse $\forall a \in F$, a's additive and multiplicative inverses exist (F4) and are unique (THM C.1 COR 1), and are denoted $-a$ and a^{-1}, respectively.

THM C.2 Given $a, b \in F$, then the following statements are true:

- $a \cdot 0 = 0$
- $(-a) \cdot b = a \cdot (-b) = -(a \cdot b)$
- $(-a) \cdot (-b) = a \cdot b$

THM C.2. COR 1 The additive identity of a field has no multiplicative inverse.

DEF characteristic The characteristic of a field is defined as the smallest $p \in \mathbb{Z}^+$ s.t. $1 + 1 + \cdots + 1 = 0$ (p summands). Notes:

- If no integer exists, then field has characteristic zero.
- Fields of characteristic 2 are problematic.
- If $p \neq 0$, the sum $\forall x \in F$, $x + x + \cdots + x = 0$ (p summands).

2 1.2. Vector Spaces

DEF external binary NOT SURE ABOUT THIS DEF. IS IT $V \times F \to V$, or is more general than for v.s.es?

DEF vector space (v.s.) A vector space (linear space) V over a set F consists of a set on which two operations (addition (binary) and scalar multiplication (external binary)) are defined s.t. addition and scalar multiplication are closed over the v.s., and the following eight vector space axioms hold.

1. Commutativity of vector addition
2. Associativity of vector addition
3. Existence of the identity 0 for vector addition
4. Existence of the inverse for vector addition
5. $\forall x \in V,\ 1x = x$
6. $\forall a, b \in F, \forall x \in V,\ (ab)x = a(bx)$
7. $\forall a \in F, \forall x, y \in V,\ a(x + y) = ax + ay$
8. $\forall a, b \in F, \forall x \in V,\ (a + b)x = ax + bx$

THM 1.1. Cancellation Law for Vector Addition Let $x, y, z \in F$. Then $x + y = y + z \Rightarrow x = z$.

THM 1.1. COR 1 Uniqueness of the 0 vector.

THM 1.1. COR 2 Uniqueness of the additive inverse.

THM 1.2. Let V be a v.s. Then:

- $0x = 0$
- $(-a)x = -(ax) = a(-x)$
- $a0 = 0$

2.1 Examples of common v.s.

F^n set of n-tuples over elementwise addition and scalar multiplication

$M_{n \times m(F)}$ matrices over elementwise addition and scalar multiplication

$\mathbb{F}(S, F)$ set of functions mapping from a set S to a field F. Examples of this include sequences, which are $\mathbb{F}(\mathbb{Z}^+, F)$

$P(F)$, $P_n(F)$ set of polynomials of infinite or finite degree

3 1.3. Subspaces

DEF subspace A subset W of a v.s. V over a field F is a subspace of V if W is a v.s. over F with the $+$, $\cdot$ defined on V.

THM 1.3. Sufficient conditions for a subspace Let V be a v.s., $W \subseteq V$. Then W subsp. V IFF:

- $0 \in W$
- $x + y \in W \ \forall x, y \in W$
- $cx \in W \ \forall c \in F, \forall x \in W$

THM 1.4. Intersection of subspaces Any intersection of subspaces of a v.s. is a subspace.

3.1 Notes on subspaces

- For any v.s. V, $\{0\}$, V subsp. V.

3.2 Examples of subspaces

- The set of symmetric $n \times n$ matrices, diagonal $n \times n$ matrices, and matrices M s.t. trace$(M) = 0$ over F are subspaces of $M_{n \times n}(F)$.
- Even and odd functions in the set of functions over a field.
- P_n is the direct sum of the even and odd polynomials in P_n.

3.3 Results from the homework

20. Union of subspaces $W_1 \cup W_2$ subsp. V IFF $W_1 \subseteq W_2$ or $W_2 \subseteq W_1$.

DEF set sum Let S_1, S_2 be non-empty subsets in V. The sum $S_1 + S_2 = \{x + y : x \in S_1, y \in S_2\}$.

DEF direct sum A v.s. V is the direct sum $V = W_1 \bigoplus W_2$ if W_1, W_2 subsp. V s.t. $W_1 \cap W_2 = \{0\}$, $W_1 + W_2 = V$.

23. Set sum of subspaces Let W_1, W_2 subsp. V. Then $W_1 + W_2$ subsp. V, and W_1, $W_2 \subseteq W_1 + W_2$. Also, any subsp. of V containing W_1 and W_2 contains $W_1 + W_2$.

30. Uniqueness of representation in direect sum If $W_1 \bigotimes W_2 = V$, then $\forall v \in V$, v can be uniquely be expressed as $w_1 + w_2, \ w_1 \in W_1, \ w_2 \in W_2$.

1. Null set vs. zero subspace The null set is never a subspace (all spaces must include the zero vector, so it is the smallest subspace)

9, 10. Good ways to check for subspaces:

- Check if the subspace has the zero vector. This is very easy to see and happens any time a linear combinations equals a nonzero vector.
- If the condition is that a linear combination equals zero, then it is the null space of a linear transformation, which is a v.s.

4 1.4. Linear Combinations and Systems of Linear Equations

DEF linear combination (l.c.) Let V be a v.s., $S \subseteq V$ nonempty subset of V. A vector $v \in V$ is a lin. comb. of vectors in S if there exist a finite number of vectors in S and a finite number of scalars in F s.t. $v = \sum_{i=1}^{n} a_n u_n$.

DEF span Let S be a nonempty subset of v.s. V. The span of S is the set of all lin. combs. of the vectors in S. Define span$(\varnothing) = \{0\}$.

THM 1.5 The span of any subset S of a v.s. V subsp. V. Moreover, any subset of V containing S must also contain span(S).

DEF generate A subset S of a v.s. V generates (or spans) V if span(S)=V.

4.1 Notes about linear combinations

- 0 is a possible linear combination of any set.
- To find whether a vector is a linear combination of others, write it as lin. comb. and solve for coefficients w/ system of equations.

4.2 Results from the homework

17. Number of generating subsets $\text{card}(V) = n \iff V$ has finitely many generating subsets.

5 1.5. Linear dependence and linear independence

DEF linear dependence A subset S of a v.s. V is called linearly dependent if there exists a nontrivial lin. comb. over S that yields 0; i.e., there exists a nontrivial lin. comb. over S that equals 0.

DEF linear independence A subset S of a v.s. that is not linearly dependent is linearly dependent; i.e., lin. ind. IFF the only representations of 0 as linear combinations over S are trivial representations.

THM 1.6. Let V be v.s., $S_1 \subseteq S_2 \subseteq V$. If S_1 lin. dep, then S_2 lin. dep. Corollary (contrapositive): If S_2 lin. ind., then S_1 lin. ind.

THM 1.7. Let S be a lin. ind. subset of a v.s. V, let v be a vector in V not in S. Then $S \cup \{v\}$ lin. ind. IFF $v \in \text{span}(S)$.

5.1 Notes and results from the homework

- Any subset containing the 0 vector is linearly dependent.
- The empty set is linearly independent.
- A set consisting of a single nonzero vector is linearly independent.
- To show lin. ind., set lin. comb. of all vectors to 0, show all coefficients 0.

1a. Not every vector from a linearly dependent set may be expressable as a linear combination of the other vectors in that set.

6 1.6. Bases and dimension

DEF basis A basis β for a v.s. V is a lin. ind. subset of V s.t. $\text{span}(\beta) = V$.

THM 1.8 Let V be a v.s., $\beta = u_i$, $1 \leq i \leq n$. Then β basis for V IFF each $v \in V$ can be uniquely expressed as a lin. comb. of vectors of β.

THM 1.9 If a v.s. V is generated by a finite set S, then some subset of S is a basis for V. Hence V has a finite basis. ("A finite spanning set can be reduced to a basis for V")

THM 1.10 Replacement theorem Let V be a v.s. generated by a set G containing n vectors, let L be a lin. ind. subset of V containing m vectors. Then $m \leq n$ and $\exists H \subseteq G$, $\text{card}(H) = n - m$ s.t. $\text{span}(L \cup H) = V$.

THM 1.10 Corollary 1 Dimension of bases Let V be a v.s. having a finite basis. Then every basis for V contains the same number of vectors.

DEF dimension A v.s. is finite-dimensional if it has a basis consisting of a finite number of vectors. The unique number of vectors in each basis for V is called the dimension of V. A v.s. that is not finite-dimensional is called infinite-dimensional.

THM 1.10 Corollary 2 Let V be a v.s., $\dim(V) = n$. Then:

- Any finite generating set for V contains at least n vectors, and a generating set for V that contains exactly n vectors is a basis for V.
- Any linearly independent subset of V that contains exactly n vectors is a basis for V.
- Every linearly independent subset of V can be extended to a basis for V.

(i.e., think: generating set always greater cardinality than lin. ind. set)

THM 1.11 Dimension of subspaces Let W subsp. finite-dimensional v.s. V. Then W is finite-dimensional, $\dim(W) \leq \dim(V)$. If $\dim(W) = \dim(V)$, then $W = V$.

THM 1.11 Corollary 1 If W subsp. a finite-dimensional v.s. V, then any basis for W can be extended to a basis for V.

6.1 Example bases

- $\varnothing$ is the basis for $\{0\}$.
- $\{e_i\}$, $1 \leq i \leq n$ is the standard basis for F^n. Standard bases are also defined for $M_{m\times n}(F)$, $P_n(F)$.
- The Lagrange polynomials. STUDY THIS

6.2 Results from the HW

20. Extension of THM 1.9 to arbitrary subspaces Let V be a v.s. having dimension n, and let S be a subset of V that generates V. Then there is a subset of S that is a basis for V. (S can be infinite.)

24. Polynomial and derivatives If $f \in P_n(F)$ has degree n, then $\{f, f', f'', \cdots f^{(n)}\}$ is a basis for $P_n(F)$.

29. Dimension of set sum Two results:

- If W_1, W_2 finite-dimensional, subsp. V, then $dim(W_1 + W_2) = dim(W_1) + dim(W_2) - dim(W_1 \cap W_2)$.
- Let W_1, W_2 be finite-dimensional subspaces of v.s. V, let $V = W_1 + W_2$. Then $V = W_1 \bigoplus W_2 \iff \dim(V) = \dim(W_1) + \dim(W_2)$.

Examples 11, 12 from chapter The dimension of a v.s. may depend on its field; i.e., the v.s. of complex numbers may have dimension 1 over a field of $\mathbb{C}$, and dimension 2 over a field of $\mathbb{R}$.

6.3 Lagrange Interpolation Formula

Probably not on the tests, but a very interesting example nonetheless.

Given a finite set of distinct scalars $c_0, c_1, \cdots, c_n$ in an infinite field F. Then the polynomials $f_0, f_1, \cdots, f_n$, where $f_i(x) = \prod_{k=0, k\neq i}^{n} \frac{x-c_k}{c_i-c_k}$, are a basis for $P_n(F)$, and $f_i(c_j) = \begin{cases} 0 & i \neq j, \\ 1 & i = j \end{cases}$. Thus $\sum_{i=1}^{n} a_i f_j(c_j) = a_j$. We can show that $\{f_i\}$ is a basis by assuming $\sum_{i=1}^{n} a_i f_i = 0$; then, at each c_j we have an equation that asserts that c_j equals zero, i.e., $\sum_{i=1}^{n} a_i f_i(c_j) = 0 = c_j$. Since $\{f_i\}$ is a basis,

then $\forall g \in P_n(F)$, $g(c_j) = \sum_{i=1}^{n} b_i f_i(c_j) = b_j \Rightarrow g = \sum_{i=1}^{n} g(c_i) f_i$. This can be applied to any set of ordered pairs of coordinates, $(c_i, g(c_i)),\ 1 \leq i \leq n$ to express (exactly) some polynomial g.

7 2.1. Linear Transformations, Null Spaces, and Ranges

DEF linear transformation Let V, W be v.s. over F. Then, the function $T : V \to W$ is called a linear transformation from V to W, if $\forall x, y \in F$, $\forall c \in F$, $T(x + y) = T(x) + T(y)$ and $cT(x) = T(cx)$).

Properties of a linear transformation Useful properties:

T linear, then $T(0) = 0$

- T linear IFF $T(cx+y) = cT(x)+T(y)$ (easier to check this than two things).
- The linear map over a linear combination preserves its structure, yielding a lin. comb. over the images of the vectors in the lin. comb with the same coefficients.

DEF kernel (null space) and range (image) Let V, W be v.s., $T : V \to W$ is linear. Define the null space (kernel) $N(T) = \{x \in V : T(x) = 0\}$. Also, define the range (image) $R(T) = \{T(x) : x \in V\}$.

THM 2.1 Let V, W be v.s., $T : V \to W$ linear. Then $R(T)$, $N(T)$ subsp. W, V, respectively.

THM 2.2 Let V, W be v.s., $T : V \to W$ linear, and β is a basis for V. Then $\text{span}(T(\beta)) = \text{span}(R(T))$.

DEF nullity, rank Let V, W be v.s., $T : V \to W$ linear. If $N(T)$, $R(T)$ finite dimensional, then define $\text{rank}(T) = \text{rank}(R(T))$, $\text{nullity}(T) = \text{rank}(N(T))$.

THM 2.3 Dimension Theorem Let V, W be v.s., $T : V \to W$ is linear. If V finite-dimensional, then $\text{nullity}(T) + \text{rank}(T) = \dim(V)$.

THM 2.4 Null space of 1-1 transforms Let V, W be v.s. $T : V \to W$ linear. Then T is 1-1 IFF $N(T) = \{0\}$.

THM 2.5 Equivalence of 1-1, onto transforms Let V, W be v.s. of equal (finite) dimension, and let $T : V \to W$ be linear. Then the following are equivalent:

- T 1-1 (injective)
- T onto (surjective)
- $\text{rank}(T) = \dim(V)$

THM 2.6 Let V, W be v.s. over F, suppose $\{v_1, v_2, \cdots, v_n\}$ is a basis for V. $\forall w_1, w_2, \cdots, w_n \subseteq W,\ \exists!\ T \in \mathcal{L}(V,W) : T(v_i) = w_i,\ 1 \leq i \leq n$.

THM 2.6 Corollary 1 Let V, W be v.s., suppose V has the basis $\{v_1, v_2, \cdots, v_n\}$. If $U, T \in \mathcal{L}(V,W)$ and $U(v_i) = T(v_i),\ 1 \leq i \leq n$, then $U = T$.

7.1 Examples of linear transformations

rotation in cartesian coordinates in $\mathbb{R}^2$ $T_\theta(a,b) = (a_1 \cos\theta - a_2 \sin\theta, a_1 \sin\theta + a_2 \cos\theta)$

zero and identity transforms den. 0_V, 1_V or I_V

7.2 Notes from the homework

1. **Finding a transform that maps a given set to a given image** You cannot always find a linear map s.t. for some given $v_i \subset V$, $w_i \subset W$, $T(v_i) = w_i$.

2. **1-1 doesn't imply onto** It is possible for two transforms to be onto but not 1-1; THM 2.5 only holds if the domain and codomain are of the same dimension.

DEF T-invariant Let V be a v.s, and let $T : V \to V$ be linear. A subspace W of V is said to be T-invariant if $T(x) \in W\ \forall x \in W$; i.e., $T(W) \subseteq W$. If W is T-invariant. we define the restriction of T on W to be the function $T_W : W \to W$, $T_W(x) = T(x)\ \forall x \in W$.

7.3 Notes on linear transforms

- (THM 2.5) doesn't apply for infinite-dimensional v.s.; i.e., onto isn't equivalent to 1-1 in this case. Also, (THM 2.4) and (THM 2.5) rely on linearity of T.

- (THM 2.6 Corollary 1) indicates that the mapping of any linear transform on the basis of its domain is unique. (THM 2.6) allows us to find a transform if a mapping of its basis is known, since it must be unique.

8 2.2 The Matrix Representation of a Linear Transformation

DEF ordered basis Let V be a finite-dimensional v.s. An ordered basis for V is a basis endowed with a specific order. For some v.ses, a standard ordered basis is defined. For our purposes, the choice of standard basis is arbitrary but must be consistent.

DEF coordinate vector Let $\beta = u_1, u_2, \cdots, u_n$ be an O.B. for a finite-dimensional v.s. V. $\forall x \in V$ x can be represented as a lin. comb. over β, i.e., $x = \sum_{i=1}^{n} a_i u_i$, $1 \leq i \leq n$. We define the coordinate vector of x relative to β as $[x]_\beta =$ HOW TO USE MATRICES

DEF matrix representation of a linear transform Let $\beta = v_1, v_2, \cdots, v_n$ be an O.B. for V, $\gamma = w_1, w_2, \cdots .w_n$ be an O.B. for W. Let $T \in \mathcal{L}(V, W)$. Then $T(v_j) = \sum_{i=1}^{n} a_{ij} w_i$. The matrix representation of T in the ordered bases β and γ is defined by $A_{ij} = a_{ij}$, and denote $A = [T]_\beta^\gamma$. In other words, it is the partitioned matrix $A = [\ [T(v_1)]_\gamma \mid [T(v_2)]_\gamma \mid \cdots \mid [T(v_n)]_\gamma\ ]$.

DEF addition and scalar multiplication over lin. trans. Define addition and multiplication over linear transforms as the addition and scalar multiplication in the field of the results of the transforms.

THM 2.7 Set of linear transf. is a vector space Let V, W be v.s. over F. Let $T : V \to W$ be linear. Then:

- $\forall a \in F,\ aT + U$ is linear.
- The collection of all linear transformations from V to W is a v.s. over F. We'll denote this $\mathcal{L}(V, W)$.

THM 2.8 Correspondence of lin. trans. with matrix representations Let V, W be finite-dim. v.s. with O.B.s β, γ, respectively, and let $T, U \in \mathcal{L}(V, W)$. Then:

- $[T + U]_\beta^\gamma = [T]_\beta^\gamma + [U]_\beta^\gamma$
- $[aT]_\beta^\gamma = a[T]_\beta^\gamma$

DO EXERCISES

9 2.3. Composition of Linear Transformations and Matrix Multiplication

THM 2.9 Linearity of composition of lin. trans. Let V, W, Z be v.s. over F, and let $T \in \mathcal{L}(V, W)$, $U \in \mathcal{L}(W, Z)$. Then $UT \in \mathcal{L}(V, Z)$.

THM 2.10 Properties of compositions of lin. trans. Let V be a v.s. Let $T, U_1, U_2 \in \mathcal{V}$. Then:

- $T(U_1 + U_2) = T(U_1) + T(U_2)$, $(U_1 + U_2)T = U_1T + U_2T$
- $T(U_1U_2) = (TU_1)U_2$
- $TI = IT = T$
- $a(U_1U_2) = (aU_1)U_2 = U_1(aU_2),\ \forall a \in F$

DEF matrix product Let $A \in M_{m\times p}(F), B \in M_{p\times n}(F)$. Define the product $AB \in M_{m\times n}(F)$ s.t. $(AB)_{ij} = \sum_{k=1}^{p} A_{ik}B_{kj}$.

THM 2.11 Matrix representation of composition Let V, W, Z be finite-dimensional v.s. with O.B.s α, β, γ, respectively. Let $T : V \to W$ and $U : W \to Z$ be linear transformations. Then $[UT]_\alpha^\gamma = [U]_\beta^\gamma [T]_\alpha^\beta$.

Kronecker delta and I_n Define:

$$\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

Define $I_n \in M_{n\times n}(F)$ s.t. $I_{ij} = \delta_{ij}$, and call this the n-by-n identity matrix.

THM 2.12 Properties of matrix multiplication Let $M \in M_{m\times n}(F)$, $B, C \in M_{n\times p}(F)$, $D, E \in M_{q\times m}(F)$. Then:

- $A(B + C) = AB + AC$, and $(D + E)A = DA + EA$
- $a(AB) = (aA)B = A(aB)\ \forall a \in F$
- $I_m A = A = AI_n$
- If V v.s. with dim. n and O.B. β, then $[I_V]_\beta = I_n$

THM 2.13 Let $A \in M_{m\times n}(F)$, $B \in M_{n\times p}(F)$. Let u_j be the jth column of AB, v_j be the jth column of $B\ \forall 1 \leq j \leq p$. Then:

- $u_j = Av_j$
- $v_j = Be_j$

THM 2.14 Linear transformations as matrix multiplication Let V, W be finite-dimensional v.ses with bases O.B.s β, γ, respectively. Let $T \in \mathcal{L}(V, W)$. Then, $\forall u \in V$, $[T(u)]_\beta^\gamma = [T]_\beta^\gamma [u]_\beta$.

DEF Left-multiplication Transformation Let $A \in M_{m\times n}(F)$. Define the transformation L_A, called the left-multiplication transformation, by $L_A : F^n \to F^m$, $L_A(x) = Ax\ \forall x \in F^n$.

THM 2.15 Properties of the left-multiplication transformation Let $A \in M_{m\times n}(F)$. Then L_A is linear. Furthermore, let $B \in M_{m\times n}(F)$, β, γ S.O.B.s for F^n, F^m, respectively, then:

- $[L_A]_\beta^\gamma = A$
- $L_A = L_B \iff A = B$
- $L_{A+B} = L_A + L_B$, and $L_{aA} = aL_A\ \forall a \in F$
- If $T : F^n \to F^m$ is linear, then $\exists!\, C$ s.t. $T = L_C$, and $C = [T]_\beta^\gamma$.
- If $E \in M_{n\times p}(F)$, then $L_{AE} = L_A L_E$.
- $m = n \Rightarrow L_{I_n} = I_{F^n}$.

THM 2.16 Let A, B, C be matrices s.t. $A(BC)$ is defined. Then so is $(AB)C$, and multiplication is associative.

9.1 Notes about matrix multiplication

- The transpose of a matrix product is the product of the transposes of the factors in reverse order, i.e., $(AB)^T = B^T A^T$.
- The cancellation law is not valid for matrices, since two different matrices may multiply the same matrix to get the same product. (E.g., some non-zero matrices square to 0.)

10 2.4. Invertibility and Isomorphisms

DEF invertible linear transformation Let V and W be v.ses, and let $T \in \mathcal{L})(V, W)$. A function $U \in \mathcal{L}(W, V)$ is said to be an inverse of T if $TU = I_W$ and $UT = I_V$. If T has an inverse, it is said to be invertible, and its inverse is unique (by properties of invertible functions). Some properties for inverse linear transformations hold from the study of inverse functions (from Appendix B, can be used without proof):

- $(TU)^{-1} = U^{-1}T^{-1}$
- $(T^{-1})^{-1} = T$ (and thus T^{-1} is invertible)
- T is invertible IFF it is one-to-one (and thus onto, $\text{rank}(T) = \dim(V)$, etc.)

THM 2.17 Let V, W, be v.ses, and let $T \in \mathcal{L}(V, W)$ be invertible. Then T^{-1} is also linear.

DEF. invertible matrix Let $A \in M_{n\times n}(F)$. Then A is invertible if $\exists B \in M_{n\times n}(F)$ s.t. $AB = BA = I_n$.

LEM. Dimensions of domain and codomain of invertible transformations Let $T \in \mathcal{L}(V, W)$ invertible. Then V is finite-dimensional IFF W is finite-dimensional. In this case, $\dim(V) = \dim(W)$.

THM 2.18 Matrix representation of inverse transformation Let V, W be finite-dimensional v.ses with O.Bs β and γ, respectively. Let $T \in (V, W)$. Then T invertible IFF $[T]_\beta^\gamma$ invertible, and $[T]_\beta^\gamma = ([T^{-1}]_\gamma^\beta)^{-1}$.

THM 2.18 COR 2 Let $A \in M_{n\times n}(F)$. Then A invertible IFF L_A invertible, and $L_A^{-1} = L_{A^{-1}}$.

DEF isomorphic, isomorphism Let V, W be v.ses. We say that V is isomorphic to W if there $\exists T \in \mathcal{L}(V, W)$ invertible. T is called an isomorphism from V onto W.

THM 2.19 Let V, W be finite-dimensional v.ses. (over the same field). Then V isomorphic to W IFF $\dim(V) = \dim(W)$.

THM 2.20 Let V, W be finite-dimensional v.ses over F of dimensions n and m, respectively, and let β and γ be the respective O.Bs. Then the function $\Phi : \mathcal{L}(V, W) \to M_{m\times n}(F)$, defined by $\Phi(T) = [T]_\beta^\gamma$ for $T \in \mathcal{L}(V, W)$, is an isomorphism.

THM 2.19 COR 1 Let V, W be finite-dimensional v.ses of dimensions n, m, respectively. Then $\mathcal{L}(V, W)$ has dimension nm.

DEF standard representation function Let β be an O.B. for an n-dimensional v.s. V over the field F. The standard representation of V w.r.t. β is the function $\phi_\beta : V \to F^n$ defined by $\phi_\beta(x) = [x]_\beta \, \forall x \in V$.

THM 2.21 For any finite-dimensional v.s. V with O.B. β, ϕ_β is an isomorphism.

Commutativity diagram: performing matrices indirectly $\phi_\gamma T = L_{[T]_\beta^\gamma}\phi_\beta$. Note that ϕ_β, ϕ_γ invertible (but not necessarily T or L_A), so this has more possibilities open.

11 2.5. The Change of Coordinate Matrix

THM 2.22 Let β, β' be O.Bs for a finite dimensional v.s. V, and let $Q = [I_V]_{\beta'}^\beta$. Then:

- Q is invertible
- For any $v \in V$, $[v]_\beta = Q[v]_{\beta'}$

DEF linear operator A linear operator is a linear map with the same domain and codomain.

THM 2.23 Let T be a linear operator on a finite-dimensional v.s. V, and let β and β' be O.Bs for V. Suppose that Q is the change of coordinate matrix changing β' coordinates to β coordinates. Then $[T]_{\beta'} = Q^{-1}[T]_\beta Q$.

THM 2.23 COR 1 Let $A \in M_{n\times n}(F)$, and γ be an O.B. for F^n. Then $[L_A]_\gamma = Q^{-1}AQ$, where Q is the $n \times n$ matrix whose jth column is the jth vector of γ.

DEF similar Let A, B be matrices in $M_{n\times n}(F)$. We say that B is similar to A if there exists an invertible matrix Q s.t. $B = Q^{-1}AQ$.

12 2.6. Dual Spaces (covered in brief)

DEF linear functional A linear functional is a linear transformation from a v.s. V to its field of scalars F.

DEF dual space For a v.s. V over F, define the dual space of V to be $\mathcal{L}(V, F)$, denoted by V^*.

13 3.1. Elementary Matrix Operations and Elementary Matrices

DEF elementary row [column] operation Let $A \in M_{m\times n}$. Any one of the following operations on the rows [columns] of A is called an elementary row [column] operation:

1. interchanging any two rows [columns] of A
2. multiplying any row [column] of A by a nonzero scalar
3. adding any scalar multiple of a row [column] of A to another row [column]

DEF elementary matrix An $n \times n$ elementary matrix is a matrix obtained by performing an elementary operation on I_n. The elementary matrix is called type 1, type 2, or type 3, depending on the type of elementary operation applied to it.

THM 3.1 Let $A \in M_{m\times n}(F)$, and suppose B is obtained from A by performing an elementary row [column] operation. Then there exists an $m \times m$ $[n \times n]$ elementary matrix E s.t. $B = EA$ $[B = AE]$. In fact, E is obtained from I_m $[I_n]$ by performing the same elementary row [column] operation performed on A to get B. The converse is also true.

THM 3.2 Elementary matrices are invertible, and the inverse of an elementary matrix is also an elementary matrix.

13.1 Notes about elementary matrices

- Each elementary matrix can be obtained by either a row or column operation.

14 3.2. The Rank of a Matrix and Matrix Inverses

DEF. rank If $A \in M_{m\times n}(F)$, we define $\text{rank}(A) = \text{rank}(L_A)$.

THM 3.3 Let $T \in \mathcal{L}(V, W)$ be a lin. trans. between finite-dim. v.ses, and let β, γ be respective O.Bs for V, W. Then $\text{rank}(T) = \text{rank}([T]_\beta^\gamma)$.

THM 3.4 Let $A \in M_{m\times n}$. If $P \in M_{m\times m}$, $Q \in M_{n\times n}$ invertible, then: $\text{rank}(PA) = \text{rank}(A) = \text{rank}(AQ) = \text{rank}(PAQ)$.

THM 3.4 COR 1 Elementary operations are rank-preserving.

THM 3.5 The rank of any matrix equals the maximum number of its linearly independent columns; that is, the rank of a matrix is the dimension of $\text{Col}(A)$.

THM 3.6 Let $A \in M_{m\times n}(F)$ of rank r. Then $r \leq m, n$, and, by means of (a finite number of) elementary operations, A can be transformed into a matrix s.t. $A_{ij} = \begin{cases} 1 & i = j, i \leq r \\ 0 & \text{else} \end{cases}$

THM 3.6 COR 1 Let $A \in M_{m\times n}(F)$. Then $\exists B \in M_{m\times m}(F)$, $C \in M_{n\times n}(F)$ invertible s.t. BAC is the matrix described in the statement of the theorem.

THM 3.6 COR 2 For a matrix A, $\text{rank}(A) = \text{rank}(A^t) = \text{rank}(\text{Row}(A))$ $(= \text{rank}(\text{Col}(A)))$.

THM 3.6. COR 3 Every invertible matrix is a product of elementary matrices.

THM 3.7 Let $T \in \mathcal{L}(V, W)$, $U \in \mathcal{L}(W, Z)$ on finite dim. v.s.es V, W, and Z, and let A, B be matrices s.t. AB is defined. Then:

- $\text{rank}(UT) \leq \text{rank}(U),\ \text{rank}(T)$
- $\text{rank}(AB) \leq \text{rank}(A),\ \text{rank}(B)$

DEF augmented matrix Let $A \in M_{m\times n}(F)$, $B \in M_{m\times p}(F)$. By the augmented matrix $(A|B)$, we mean the $m \times (n + p)$ matrix (AB).

Use of augmented matrix to find inverse If $A \in M_{m\times n}(F)$ invertible, and the augmented matrix $(A|B)$ is converted to $(I_n|B)$ by means of a finite number of elementary row operations, then $A^{-1} = B$. (If A is not invertible, then the rank will be less than n, which means that there will be a row of zeros on the left side).

15 3.3. Systems of Linear Equations – Theoretical Aspects

DEF system of m linear equations over n unknowns in the field F known

DEF coefficient matrix (of a system) known

DEF solution (of a system) known

DEF (in)consistent (describing a system) known

DEF homogeneous (describing a system) known

THM 3.8 Let $Ax = 0$ be a homogeneous lin. sys. of m lin. equations in n unknowns over a field F. Let K denote the set of all solutions to the system. Then $K = N(L_A)$; hence, K is a subspace of F^n of dimension $n - \text{rank}(L_A) = n - \text{rank}(A)$.

THM 3.8 COR 1 If $m < n$, then the system $Ax = 0$ has a nonzero solution.

THM 3.9 Let K be the sol'n set of a system $Ax = b$, and let K_H be a solution of the corresponding homogeneous system $Ax = 0$. Then for any solution s to $Ax = b$, $K = \{s\} + K_H = \{s + k : k \in K_H\}$.

THM 3.10 Let $Ax = b$ be a system of n linear equations in n unknowns. If A is invertible, then the system has exactly one solution, namely, $A^{-1}b$. Conversely, if the solution has exactly one solution, then A is invertible.

THM 3.11 Let $Ax = b$ be a system of linear equations. Then the system is consistent IFF $\text{rank}(A) = \text{rank}(A|b)$. (i.e., $b \in R(L_A)$).

16 3.4. Systems of Linear Equations – Computational Aspects

DEF equivalent systems Two systems of linear equations are called equivalent if they have the same solution set.

THM 3.13 Let $Ax = b$ be a system of m linear equations in n unknowns, and let C be an invertible $n \times n$ matrix. Then the system $(CA)x = Cb$ is equivalent to $Ax = b$.

THM 3.13 COR 1 Let $Ax = b$ be a system of m linear equations and n unknowns. If $(A'|b')$ is obtained from $(A|b)$ by a finite number of elementary row operations, then the system $A'x = b'$ is equivalent to the original system.

DEF reduced row echelon form A matrix is said to be in reduced echelon if the following three conditions are satisfied:

- Any row containing a nonzero entry precedes any row in which all the entries are zero (if any).
- The first nonzero row in each row is the only nonzero entry in its column.
- The first nonzero entry in each row is 1 and it occurs in a column to the right of the first nonzero entry in the preceding row.

DEF Gaussian elimination back-substitution is more efficient than Gauss-Jordan elimination method

THM 3.14 Gaussian elimination transforms any matrix into its RREF form.

16.1 Notes about systems of linear equations

- Consistent solutions can have more than one, but a finite number of solutions (if a finite field).

17 Ways to prove v.s.

- Use 8 v.s. axioms.
- Show subspace of another v.s.
- Show it is R(T) or N(T) for some l.t. T.

18 Ways to prove basis

- Prove linearly independent, generating set.
- Prove linearly independent, correct dimension.
- Prove generating set, correct dimension.
- Image of a 1-1 linear transformation (i.e., isomorphism) on a basis.

19 Ways to show set equality

- Show set containment both ways

20 Methods for proofs

- Find all cases of (some condition): i.e., Be able to show IFF.
- Proof by contrapositive: $p \Rightarrow q \equiv \neg q \Rightarrow \neg p$. Often useful for showing reverse direction in IFF.
- Induction: show base and inductive steps.

21 Questions

- Notation: $\{T(v)\}$ vs. $T(v)$ if v is a set of vectors?
- Notation: If $T \in \mathcal{L}(V, W)$, are $R(T)$ and $T(V)$ equivalent and/or interchangeable?
- When proving invertibility by finding an inverse and multiplying it, do you have to show that it multiplies to I_n with both left and right multiplication, or is one side sufficient?
- Is example in 2.5 backwards?