Final Exam

12 Dec 2013
14:00 – 17:00
CMPT231
Dr. Sean Ho
Trinity Western University

YES:
- **Textbook**
- **Paper notes**
- **Calculator**
- **Pen/pencil and spare paper**

NO:
- **Cellphone (off/mute)**
- **Use of desktop Macs**
- **Laptop / tablet / other electronic devs**
- **Real-time communication with anyone except instructor**

Final Exam: 2-5pm, 90pts

- [6] Prove from definition: $n^2 + 4 n \lg n \in \Theta(n^2)$
- [8] Solve (prove, show work): $T(n) = 2T(n/2) + n^3$
- [8] Solve (prove, show work): $T(n) = T(n/2) + T(n/4) + T(n/8) + n$
- Demonstrate on input: A N P L . H F U D . E G B O . M Y K I
  - [8] MergeSort *(# copies?)*, [8] QuickSort (non-rand) *(# swaps?)*
  - [8] Radix sort *(convert using A→0 .. Z→25 and use base 3)*
  - [6] Bucket sort *(divide by 25 to get in the range [0,1))*
- [8] Code an efficient function to count the leaves in a binary tree
- [8] Compare and contrast dynamic programming vs. greedy. How can you tell when to use one vs. the other?
- Given the weighted, directed edge list (sorted by alpha):
  t:(w:0, x:2, y:7), w:(z:3), x:(y:4), y:(w:2, x:2), z:(y:1)
  - [6] Convert to weighted adjacency matrix and draw the graph
  - [6] Demonstrate Dijkstra shortest-paths starting at t
  - [10] Demonstrate Floyd-Warshall. What is the diameter?

Solutions: #1 (6pts) & 2 (8pts)

- Prove from definition: $n^2 + 4 n \lg n \in \Theta(n^2)$
  - Let $n_0 = 1$, $c_0 = 1$, and $c_1 = 5$, for example:
    - $\forall\ n \geq 1$: $\lg n \geq 0$, so $4n \lg n \geq 0$, so $n^2 \leq n^2 + 4n \lg n$
    - $\forall\ n \geq 1$: $\lg n < n$, so $4n \lg n < 4n^2$, so $n^2 + 4n \lg n \leq 5n^2$
  - Thus, $\forall\ n \geq 1$: $1n^2 \leq n^2 + 4n \lg n \leq 5n^2$.
- Solve (prove, show work): $T(n) = 2T(n/2) + n^3$
  - Master method: $a = 2$, $b = 2$, $f(n) = n^3$
    - $n^{log\_b(a)} = n^{lg2}$, and $f(n) = n^3 \in \Omega(n^{lg2+\varepsilon})$ for all $\varepsilon \leq 2$
    - Regularity cond: $a f(n/b) = 2(n/2)^3 = n^3/4 = c f(n)$, $c=1/4$
    - So case 3 holds, so the solution is $T(n) = \Theta(n^3)$

Solutions: #3 (8pts)

- Solve (prove, show work): $T(n) = T(n/2) + T(n/4) + T(n/8) + n$
  - Sketching out a couple levels of the recursion tree suggests that the i^{th} level of the tree has a total of $(7/8)^i n$ work, and there are $\lg n$ levels in the tree.
  - So our guess is $\Sigma_1^{\lg n} (7/8)^i n = \Theta(n)$. We still need to prove it!
  - First prove $T(n) \in O(n)$ by induction:
    - Inductive hypothesis: $T(m) \le cm \ \forall\ m < n$, for some $c > 0$
    - Then $T(n) = T(n/2) + T(n/4) + T(n/8) + n$
      $\le cn/2 + cn/4 + cn/8 + n$
      $= (7/8)cn + n$
      $= (7/8 + 1/c)\ cn$
      $\le cn$ if $c \ge 8$
    - This proves $T(n) \in O(n)$.
  - Also, $T(n) = T(n/2) + T(n/4) + T(n/8) + n \ge n$, so $T(n) \in \Omega(n)$.
  - Hence, $T(n) \in \Theta(n)$.

Solutions: #4a (8pts)

- MergeSort on A N P L . H F U D . E G B O . M Y K I:
  - The entire list is copied at each level of recursion, so the total number of copies of elements is n lg n = 64.

Solutions: #4b (8pts)

- QuickSort on A N P L . H F U D . E G B O . M Y K I:
  - Total of 18 non-trivial swaps:

```
 0: A N P L H F U D E G B O M Y K I
 1: . H P L N
 2: . . F L N P
 3: . . . D N P U L
 4: . . . . E P U L N
 5: . . . . . G U L N P
 6: . . . . . . B L N P U
 7: A H F D E G B I N P U O M Y K L
 8: A B F D E G H
 9: . . D F
10: . . D E F
11: . . . . . . . . K P U O M Y N
12: . . . . . . . . K L U O M Y N P
13: . . . . . . . . . . O U
14: . . . . . . . . . . . M U
15: . . . . . . . . . . . . N Y U
16: . . . . . . . . . . O M N P U Y
17: . . . . . . . . . . M O
18: . . . . . . . . . . M N O
```

Solutions: #4c (8pts)

- Radix sort on A N P L . H F U D . E G B O . M Y K I:

A: 0 0 0	0 0 0	0 0 0	0 0 0	A
N: 1 1 1	1 2 0	0 0 1	0 0 1	B
P: 1 2 0	0 1 0	1 0 1	0 1 0	D
L: 1 0 2	0 2 0	1 0 2	0 1 1	E
H: 0 2 1	1 1 0	2 0 2	0 1 2	F
F: 0 1 2	2 2 0	0 1 0	0 2 0	G
U: 2 0 2	1 1 1	1 1 0	0 2 1	H
D: 0 1 0	0 2 1	1 1 1	0 2 2	I
E: 0 1 1	0 1 1	0 1 1	1 0 1	K
G: 0 2 0	0 0 1	0 1 2	1 0 2	L
B: 0 0 1	1 0 1	1 1 2	1 1 0	M
O: 1 1 2	1 0 2	1 2 0	1 1 1	N
M: 1 1 0	0 1 2	0 2 0	1 1 2	O
Y: 2 2 0	2 0 2	2 2 0	1 2 0	P
K: 1 0 1	1 1 2	0 2 1	2 0 2	U
I: 0 2 2	0 2 2	0 2 2	2 2 0	Y

Solutions: #4d (6pts)

- Bucket sort on A N P L . H F U D . E G B O . M Y K I:

bkt	start	start25	ltrs	actual
0	0.	0.	A B	A B
1	0.0625	1.5625	C D	D
2	0.125	3.125	E	E
3	0.1875	4.6875	F G	F G
4	0.25	6.25	H	H
5	0.3125	7.8125	I J	I
6	0.375	9.375	K	K
7	0.4375	10.9375	L M	L M
8	0.5	12.5	N O	N O
9	0.5625	14.0625	P	P
10	0.6250	15.625	Q R	
11	0.6875	17.1875	S	
12	0.75	18.75	T U	U
13	0.8125	20.3125	V	
14	0.875	21.875	W X	
15	0.9375	23.4375	Y	Y

Solutions: #5 (8pts)

- Code an efficient function to count the leaves in a binary tree:
  - def CountLeaves(T):
    - if isnull(T):   // null ref
      - return 0
    - if isnull(T.left) and isnull(T.right): // I am a leaf on the wind
      - return 1
    - return CountLeaves(T.left) + CountLeaves(T.right)

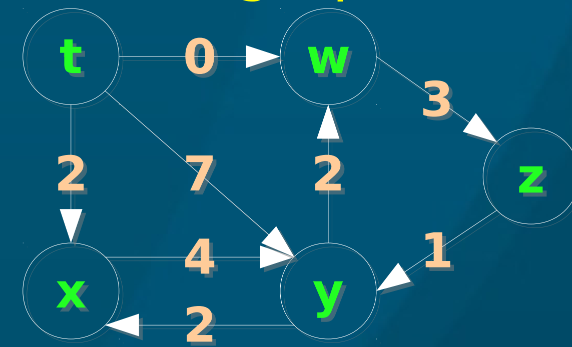
Solutions: #6 (8pts)

- Dynamic programming:
  - Task substructure: split subtasks recursively
  - Optimal substructure: optimal solutions are built from optimal solutions to subproblems
  - Choice / decisions / optimisation: examine all subtasks and choose the best one
  - Reuse of subproblems: hence storing subproblem results in table for reuse, speedup
- Greedy algorithms:
  - Subset of dynprog: also needs optim substr
  - Adds greedy choice property: optimal solutions are built from picking the greedy choice
    - no need to check all subtasks, just greedy choice

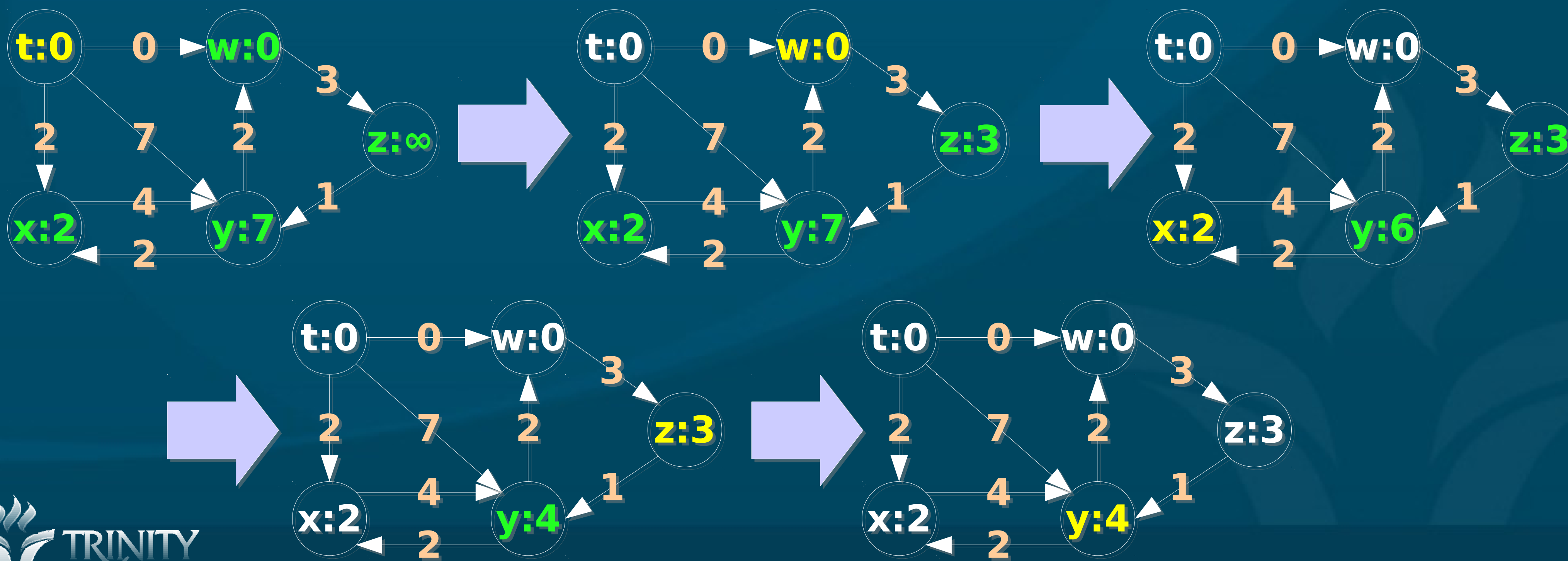
Solutions: #7a (6pts), 7b (6pts)

- Convert to weighted adjacency matrix and draw the graph

.	t	w	x	y	z
t:	0	0	2	7	∞
w:	∞	0	∞	∞	3
x:	∞	∞	0	4	∞
y:	∞	2	2	0	∞
z:	∞	∞	∞	1	0



- Demonstrate Dijkstra shortest-paths starting at t



Solutions: #7c (10pts)

- Demonstrate Floyd-Warshall. What is the diameter?
  - Diameter is max value in the final matrix: δ(x,z) = 9

k=0 (orig)

.	t	w	x	y	z
t:	0	0	2	7	∞
w:	∞	0	∞	∞	3
x:	∞	∞	0	4	∞
y:	∞	2	2	0	∞
z:	∞	∞	∞	1	0

k=1 (via t)

(same as k=0 since no edges go into t)

k=2 (via w)

.	t	w	x	y	z
t:	0	0	2	7	3
w:	∞	0	∞	∞	3
x:	∞	∞	0	4	∞
y:	∞	2	2	0	5
z:	∞	∞	∞	1	0

k=3 (via x)

.	t	w	x	y	z
t:	0	0	2	6	3
w:	∞	0	∞	∞	3
x:	∞	∞	0	4	∞
y:	∞	2	2	0	5
z:	∞	∞	∞	1	0

k=4 (via y)

.	t	w	x	y	z
t:	0	0	2	6	3
w:	∞	0	∞	∞	3
x:	∞	6	0	4	9
y:	∞	2	2	0	5
z:	∞	3	3	1	0

k=5 (via z)

.	t	w	x	y	z
t:	0	0	2	4	3
w:	∞	0	6	4	3
x:	∞	6	0	4	9
y:	∞	2	2	0	5
z:	∞	3	3	1	0