Data Structures and Algorithms CSE220

Prof. Ramesh Ragala

July 25, 2022

Recurrence Relation

Definition:

A Recurrence Relation for a sequence $\{ a_n \}$ is an equation that express a_n in terms of one or more of the previous terms in the sequence, a_0, a_1,a_2,a_3, ... a_{n-1} for all integers $n \geq n_0$ where n_0 is a non-negative integer.

A sequence is called a solution of a recurrence relation if its terms satisfy the recurrence relation.

- Motivating Examples:
  - Finding the Factorial of a given number, Fibonacci series, Towers of Hanoi, etc.
  - Some of the problems solved by DAC approach
- Simply, A rule to determining subsequent terms from those that precede them is called a recurrence relation.

Introduction to Recurrence Relations

- Simple Example
  - Assume we have a set of integers as like $\{1,2,4,8,16,32,..\}$
  - What will be the next integer in above set

Introduction to Recurrence Relations

- Simple Example
  - Assume we have a set of integers as like {1,2,4,8,16,32,..}
  - What will be the next integer in above set
  - 64

- Simple Example
  - Assume we have a set of integers as like $\{1,2,4,8,16,32,..\}$
  - What will be the next integer in above set
  - 64
  - How did you find the answer

Introduction to Recurrence Relations

- Simple Example
  - Assume we have a set of integers as like {1,2,4,8,16,32,..}
  - What will be the next integer in above set
  - 64
  - How did you find the answer
  - Is there any procedure

Introduction to Recurrence Relations

- Simple Example
  - Assume we have a set of integers as like {1,2,4,8,16,32,..}
  - What will be the next integer in above set
  - 64
  - How did you find the answer
  - Is there any procedure yes

- Simple Example
  - Assume we have a set of integers as like $\{1,2,4,8,16,32,..\}$
  - What will be the next integer in above set
  - 64
  - How did you find the answer
  - Is there any procedure yes
  - After giving the first term, each term of the sequence can be defined from the previous term.
  - $a_1 = 1$

- Simple Example
  - Assume we have a set of integers as like $\{1,2,4,8,16,32,..\}$
  - What will be the next integer in above set
  - 64
  - How did you find the answer
  - Is there any procedure yes
  - After giving the first term, each term of the sequence can be defined from the previous term.
  - $a_1 = 1 \Rightarrow$

- Simple Example
  - Assume we have a set of integers as like {1,2,4,8,16,32,..}
  - What will be the next integer in above set
  - 64
  - How did you find the answer
  - Is there any procedure yes
  - After giving the first term, each term of the sequence can be defined from the previous term.
  - $a_1 = 1 \Rightarrow a_{n-1} = 2a_n$
- When a sequence is defined recursively, mathematical induction can be used to prove results about the sequence.

- A recurrence relation is like a recursively defined sequence, but without specifying any initial values (initial conditions).
- Therefore, the same recurrence relation can have (and usually has) multiple solutions.
- If both the initial conditions and the recurrence relation are specified, then the sequence is uniquely determined
- more generally, recurrences are have the form
- T(n) $= \alpha$ T(n-β) + f(n),   T(δ) = c or
- T(n) $= \alpha$ T(n/β) + f(n),   T(δ) = c is initial condition

- Example on Recurrence Relations:
- Determine whether the sequence a_n , where $a_n = 3n$ for every nonnegative integer n, is a solution of the recurrence realtion $a_n = 2a_{n-1}$-a_{n-2} for n = 2,3,4,...
- Assume $a_0 = 0$ and a_1=3
- Solution:
- Check the Initial Conditions: a_0=3(0)=0 and a_1=3(1) = 3
- Check if 3n satisfies $a_n = 2a_{n-1} - a_{n-2} \rightarrow 2a_{n-1} - a_{n-2} =$ 2[3(n-1)] - 3(n-2) = 6n-6 -3n+6 = 3n = a_n
- So, 3n is a solution for $a_n = 2a_{n-1}$-a_{n-2}

Introduction to Recurrence Relations

- Fibonacci Series:
- $\{ f_n \} = \{ 0,1,1,2,3,5,8,13,21, \}$.
- Recursive Definition for Fibonacci Series is:
  - INITIALIZE: $f_0 = 0$, $f_1 = 1$
  - RECURSIVE: $f_n = f_{n-1} + f_{n-2}$ for $n > 1$
- The recurrence relation is the recursive part of the above Definition.
- Simply, The recurrence relation for a sequence consists of an equation that expresses each term in terms of lower terms.
- Is there another solution to the Fibonacci recurrence relation?

Introduction to Recurrence Relations

- Fibonacci Series:
- $\{ f_n \} = \{ 0,1,1,2,3,5,8,13,21, \}$.
- Recursive Definition for Fibonacci Series is:
  - INITIALIZE: $f_0 = 0$, $f_1 = 1$
  - RECURSIVE: $f_n = f_{n-1} + f_{n-2} \quad$ for $n > 1$
- The recurrence relation is the recursive part of the above Definition.
- Simply, The recurrence relation for a sequence consists of an equation that expresses each term in terms of lower terms.
- Is there another solution to the Fibonacci recurrence relation?
- Yes. we can have different set of initial conditions, $f_0 = f_1 = 1$
- In this case, what will be the sequence????

Recurrence Relation for Counting Problem

- It is very difficult to get a closed formula for counting particular set.
- It is easy to get a recurrence relation for counting particular set.
- Examples:
  - Geometric example: counting the number of points of intersection of n lines.
  - A recurrence relation for the number of bit strings of length n which contain the string 00.
  - Partition Function
  - Financial Recurrence Relation

PROBLEM

RAMU deposited Rs.10,000/- in a savings account at a bank yielding 5% per year with interest compounded annually. How much money will be in the account after 30 years?

- Let P_n denote the amount in the account after n years.
- How can we determine P_n on the basis of P_{n-1}?
- We can derive the following recurrence relation:
- $P_n = P_{n-1} + 0.05P_{n-1} = 1.05P_{n-1}$.
- The initial condition is $P_0 = 10{,}000$.
- $P_1 = 1.05P_0$
- $P_2 = 1.05P_1 = (1.05)^2P_0$
- ...
- $P_n = 1.05P_{n-1} = (1.05)^nP_0 \Rightarrow$ no iteration. Just Formula $\Rightarrow$ Recurrence Relation

Recurrence Relation Examples on of String

PROBLEM

Let a_n denote the total number of bit strings of length n that do not have two consecutive 0s (valid strings). Find a recurrence relation and give initial conditions for the sequence $\{ a_n \}$.

- The number of valid strings equals the number of valid strings ending with a 0 plus the number of valid strings ending with a 1. (for n = 1 and 2)
- Let us assume that $n \geq 3$, so that the string contains at least 3 bits.
- Let us further assume that we know the number a_{n-1} of valid strings of length (n -1).
- Then how many valid strings of length n are there, if the string ends with a 1?

- There are a_{n-1} such strings, namely the set of valid strings of length (n-1) with a 1 appended to them.
- Now we need to know: How many valid strings of length n are there, if the string ends with a 0?
- Valid strings of length n ending with a 0 must have a 1 as their $(n\text{-}1)^{st}$ bit (otherwise they would end with 00 and would not be valid).
- And what is the number of valid strings of length (n - 1) that end with a 1?
- We already know that there are a n-1 strings of length n that end with a 1.
- Therefore, there are a_{n-2} strings of length (n - 1) that end with a 1.

- So there are a_{n-2} valid strings of length n that end with a 0 (all valid strings of length (n -2) with 10 appended to them).
- As we said before, the number of valid strings is the number of valid strings ending with a 0 plus the number of valid strings ending with a 1.
- That gives us the following recurrence relation:
  $\mathbf{a_n = a_{n-1} + a_{n-2}}$
- The Initial Conditions are:
- $a_1=2$ (0 and 1), $a_2 = 3$ (01, 10 and 11), $a_3 = a_2 + a_1 = 3 + 2 = 5$, $a_4 = a_3 + a_2 = 5 + 3 = 8$
- This sequence satisfies the same recurrence relation as the Fibonacci sequence. Since $a_1 = f_3$ and $a_2 = f_4$, we have $a_n = f_{n+2}$.

- There are many methods to solve the recurrence relations
  - Characteristic Equations
  - Forward and Backward Substitution
  - Master Theorem
  - Recurrence Trees

Substitution Method

- "Making a good guess" method.
  - Guess the form of the answer
  - Use Induction to find the constants and show that solution works.
- The substitution method can be used to establish either upper or lower bounds on a recurrence.
- This method can be applied only $\Rightarrow$ when it is easy to guess the form of the answer.
- Example: $\mathrm{T(n)} = 2\mathrm{T}(\lfloor \mathrm{n}/2 \rfloor) + \mathrm{n}$
- The guess solution is $\mathrm{T(n)} = \mathrm{O(n\ log\ n)}$.

$$T(n) \leq 2(c\lfloor n/2 \rfloor) log(\lfloor n/2 \rfloor)) + n$$

$$\leq cnlog(n/2) + n$$

$$= cnl0gn - cnlog2 + n$$

$$= cnlogn - cn + n$$

$$\leq cnlogn, \quad c \geq 1.$$

Substitution Method

- Mathematical induction now requires us to show that our solution holds for the boundary conditions.
- we do so by showing that the boundary conditions are suitable as base cases for the inductive proof.
- Suppose , T(1)=1 is the sole boundary condition of the recurrence.
- Then the bound $T(n) \leq c\ n\ \lg n$ yields $T(1) \leq c \log 1 = 0$ Which is odds with $T(1) = 1$.
- The base case of our inductive proof fails to hold.
- To overcome this difficulty, we can take advantage of the asymptotic notation.
- we need to prove $T(n) \leq c\ n\ \lg n$ for $n \geq n_0$.
- The idea is to remove the difficult boundary condition $T(1) = 1$ from consideration.

Substitution Method

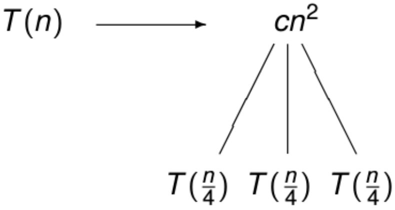
- Thus , we can replace T(1) by T(2) as the base cases in the inductive proof , letting n=2.
- From the recurrence , with $T(1) = 1$, we get T(2)=4.
- We require $T(2) \leq c\ 2\ \lg 2$.
- It is clear that , any choice of $c \geq 2$ suffices for the base cases.
- But it will work for n=3.
- choice of $c \geq 2$ is sufficient for this to hold.
- Finally $T(n) \leq c\ n\ \lg n$ for any $c \geq 2$ and $n \geq 2$.

Disadvantages of Substitution Met

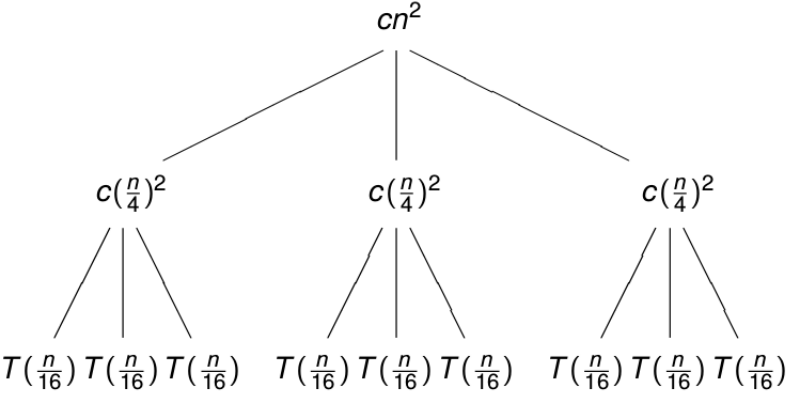
- There is no general way to guess the correct solution to the recursion.
- Guessing a solution takes experience and, occasionally, creativity.
- Some heuristics that can help you become a good guesser
- We can use recursion trees to generate good guess to recursion.
- Another way to make a good guess is to prove loose upper and lower bounds.
- Examples in class
- For some application, even we can (nearly)correctly guess at asymptotic bound $\Rightarrow$ Induction does not work properly.
- Proving exactness is missing $\Rightarrow$ Asymptotic Notations

- Expanding the recurrence into a tree
- Summing the cost at each level
- Applying the substitution method

- Consider the Recurrence Relation: $T(n) = 3T(n/4) + cn^2$ for some constant c.
- Assumption "n" is an exact power of 4.
- In the recursion-tree method we expand T (n) into a tree:

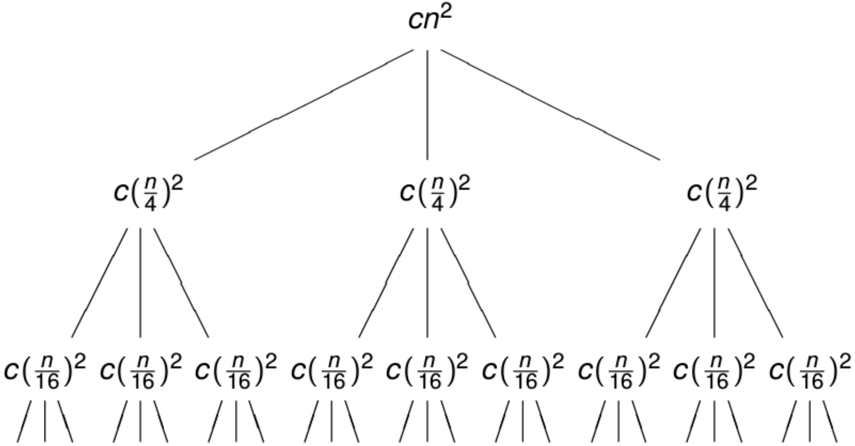


- Applying $T(n) = 3T(n/4) + cn^2$ to $T(n/4)$ leads to $T(n/4) = 3T(n/16) + c(n/4)^2$, expanding the leaves:
- The subproblem size for a node at depth i is $n/4^i$.



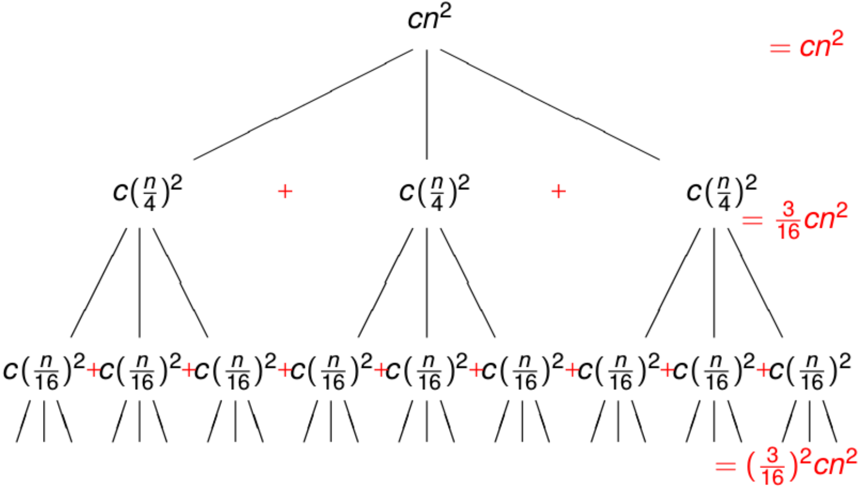
Recursion Tree Method

- Applying $T(n)=3T(n/4)+cn^2$ to $T(n/16)$ leads to $T(n/16)=3T(n/64)+c(n/16)^2$, expanding the leaves:



Recursion Tree Method

- Summing the cost at each level.



- Adding up the cost

$$T(n) = cn^2 + (3/16)cn^2 + (3/16)^2 cn^2 + \ldots$$
$$= cn^2(1 + 3/16 + (3/16)^2 + \ldots)$$

- The subproblem size hits n=1 when $n/4^i \rightarrow i = \log_4 n$
- The above equation will be disappear when $n = 16$
- **The tree has depth at least 2 if $n \geq 16 = 4^2$.**
- For $n = 4^k$, $k = \log_4(n)$, we have:

$$T(n) = cn^2 \sum_{i=0}^{\log_4(n)} \left(\frac{3}{16}\right)^i .$$

- Apply the Geometric Sum

Applying

$$S_n = \sum_{i=0}^{n} r^i = \frac{r^{n+1} - 1}{r - 1}$$

to

$$T(n) = cn^2 \sum_{i=0}^{\log_4(n)} \left(\frac{3}{16}\right)^i$$

with $r = \frac{3}{16}$ leads to

$$T(n) = cn^2 \frac{\left(\frac{3}{16}\right)^{\log_4(n)+1} - 1}{\frac{3}{16} - 1}.$$

Instead of $T(n) \leq dn^2$ for some constant d, we have

$$T(n) = cn^2 \frac{\left(\frac{3}{16}\right)^{\log_4(n)+1} - 1}{\frac{3}{16} - 1}.$$

Recall

$$T(n) = cn^2 \sum_{i=0}^{\log_4(n)} \left(\frac{3}{16}\right)^i.$$

To remove the $\log_4(n)$ factor, we consider

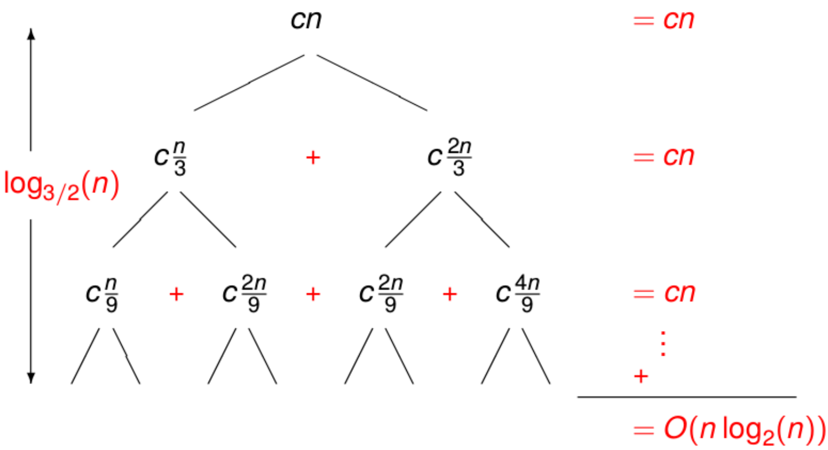
$$\begin{aligned} T(n) &\leq cn^2 \sum_{i=0}^{\infty} \left(\frac{3}{16}\right)^i \\ &= cn^2 \frac{-1}{\frac{3}{16} - 1} \leq dn^2, \text{ for some constant } d. \end{aligned}$$

Let us see if $T(n) \leq dn^2$ is good for $T(n) = 3T(n/4) + cn^2$.

Applying the substitution method:

$$
\begin{aligned}
T(n) &= 3T(n/4) + cn^2 \\
&\leq 3d\left(\frac{n}{4}\right)^2 + cn^2 \\
&= \left(\frac{3}{16}d + c\right)n^2 \\
&= \frac{3}{16}\left(d + \frac{16}{3}c\right)n^2 \\
&\leq \frac{3}{16}(2d)\,n^2, \quad \text{if } d \geq \frac{16}{3}c \\
&\leq dn^2
\end{aligned}
$$

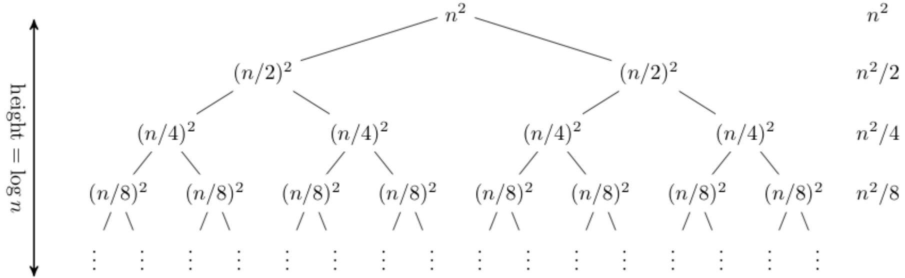
Consider $T(n) = T(n/3) + T(2n/3) + cn$.



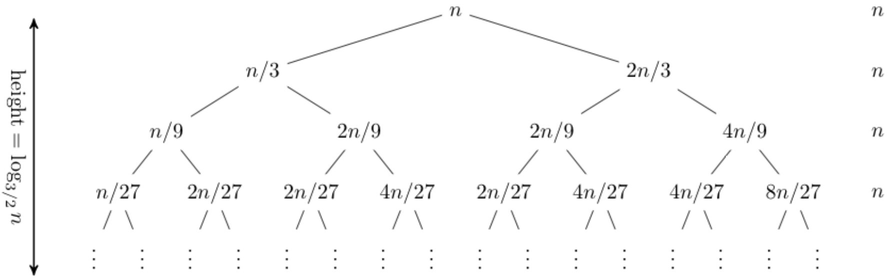
Master Method

- It is mainly used to solve the recurrence relations of the form $T(n) = a\,T(n/b) + f(n)$ where $a \geq 1$, $b > 1$ and $f(n)$ is an asymptotically positive function.
- The above recurrence relation describes the running time of an algorithm that divides a problem of size n into a subproblems, each of size n/b, where a and b are positive constants.
- The a subproblems are solved recursively, each in time $T(n/b)$.
- The cost of dividing the problem and combining the results of the subproblems is described by the function $f(n)$.

- The Recurrence Relation is: $T(n) = 2T(n/2)+n^2$



- The Recurrence Relation(not balanced) is:
  T(n) = 2T(n/3)+T(2n/3)+n



Master Theorem

- The master method uses the following theorem
- Let a$\geq$1 and b$>$1 be constants, let f(n) be a function, and let T(n) be defined on the non-negative integers by the recurrence T(n) $=$ aT(n/b)$+$f(n),
  where we interpret n/b to mean either $\lfloor$n/b $\rfloor$ or $\lceil$ n/b $\rceil$.
- Then T(n) can be bounded asymptotically as follows.
- CASE-1: If f(n) $=$ O(n$^{log_b a-\epsilon}$) for some constant $\epsilon > 0$, then T(n) $=$ Θ(n$^{log_b a}$)
- CASE-2: If f(n) $=$ Θ(n$^{log_b a}$), then T(n) $=$ Θ(n$^{log_b a}$log n)
- CASE-3: If f(n) $=$ Ω(n$^{log_b a+\epsilon}$) for some constant ϵ $>$0, and if a f(n/b) $\leq$ cf(n) for some constant c$<$1 and all sufficiently large n, then T(n) $=$ $\Theta(f(n))$

- **Simple manner,**

Theorem (Master Theorem)

Let $T(n)$ be a monotonically increasing function that satisfies

$$\begin{aligned} T(n) &= aT(\tfrac{n}{b}) + f(n) \\ T(1) &= c \end{aligned}$$

where $a \geq 1, b \geq 2, c > 0$. If $f(n) \in \Theta(n^d)$ where $d \geq 0$, then

$$T(n) = \begin{cases} \Theta(n^d) & \text{if } a < b^d \\ \Theta(n^d \log n) & \text{if } a = b^d \\ \Theta(n^{\log_b a}) & \text{if } a > b^d \end{cases}$$

Master Method

- There is a gap between cases 1 and 2 when f(n) is smaller than n $^{log_b a}$ but not polynomially smaller.
- There is a gap between cases 2 and 3 when f(n) is larger than n$^{log_b a}$ but not polynomially larger.
- If the function f(n) falls into one of these gaps, or if the regularity condition in case 3 fails to hold, the master method cannot be used to solve the recurrence.

- **Examples:**
  - $T(n) = 9T(n/3)+n$.
  - Here a = 9, b=3, f(n) =n and $n^{log_b a} = n^{log_3 9} = \Theta(n^2)$.
  - Since $f(n) = O(n^{log_3 9-\epsilon})$, where ϵ=1.
  - we can apply the CASE-1 of the master theorem.
  - So Solution is $T(n) = \Theta(n^2)$
  - Here, Given $T(n)= T(2n/3)+1$
  - Here a = 1, b=3/2, f(n)=1 and $n^{log_b a} = n^{log_{3/2} 1} = n^0 = 1$.
  - Here f(n) and $n^{log_b a}$ are equal. Then we can use CASE-2 $\rightarrow$ $f(n) = \Theta(n^{log_b a}) = \Theta(1) \rightarrow$ The solution to the recurrence $T(n) = \Theta(lgn)$

- **Examples:**
  - Given T(n) = 3 T(n/4) + n lgn
  - Here a=3, b=4, f(n) = n lgn and $n^{log_b a} = n^{log_4 3} = O(n^{0.793})$.
  - Here f(n) is larger than $n^{log_b a}$, hence we can use CASE-3.
  - For sufficiently large n, the solution to the recurrence is T(n)=$\Theta(nlgn)$
- The master method does not apply to the recurrence T(n)= 2T(n/2)+n lgn $\rightarrow$ f(n) asymptotically larger than $n^{log_b a}$. $\rightarrow$ The problem is that it is not polynomially larger.

Master Theorem Example

- We can use master theorem if
  - T(n) is not monotone, example: $T(n) = \sin n$
  - f(n) is not a polynomial, example: $T(n) = 2T(n/2) + 2^n$
  - b cannot be expressed as a constant, ex: $T(n) = T(\sqrt{n})$
- **Examples:**
  - $T(n) = T(n/2) + 1/2n^2 + n$. What are the parameters?
  - $a = ?$, $b = ?$, $d = ?$ $\rightarrow$ $a = 1$, $b = 2$ and $d = 2$
  - Which condition? $\rightarrow$ since $1 < 2^2$, then case -1 can be used.
  - Then we can conclude that $\rightarrow$ $T(n) \in \Theta(n^d) = \Theta(n^2)$


Thank you!