DESIGN AND ANALYSIS OF ALGORITHMS

CS 4120/5120

MASTER THEOREM

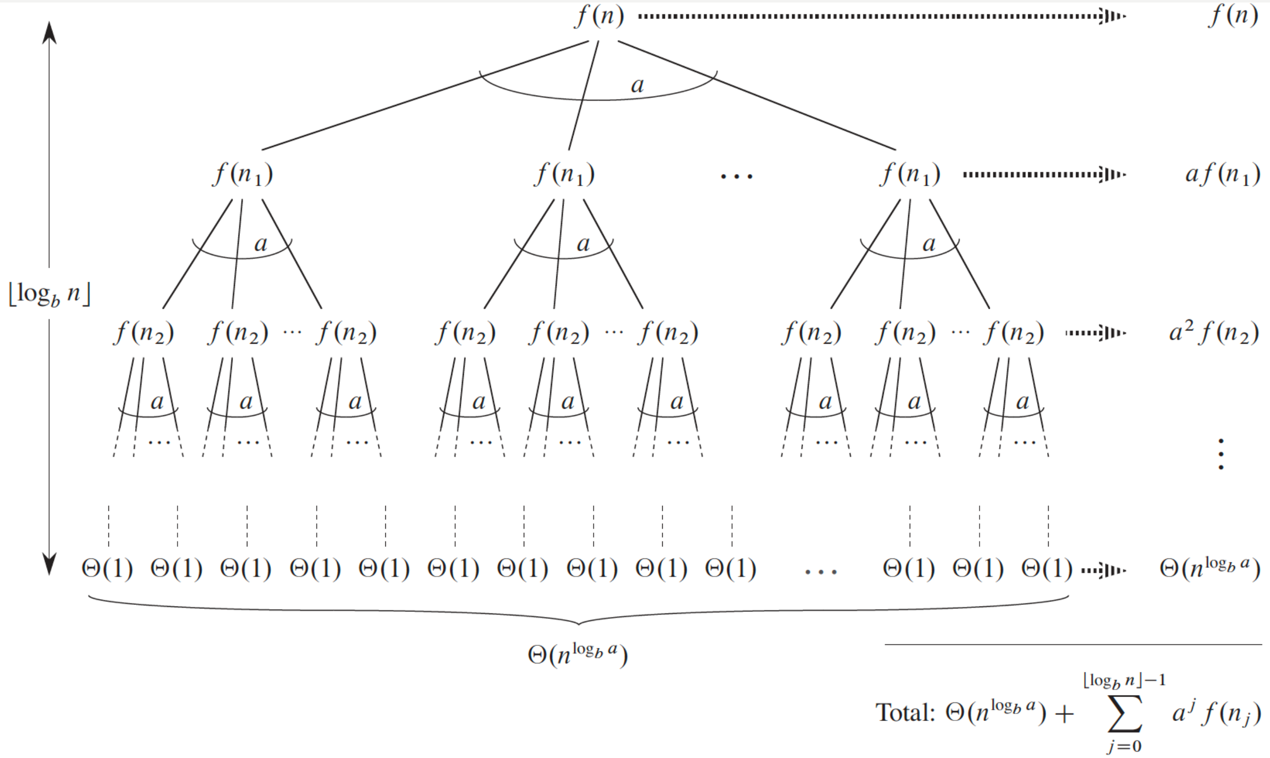
AGENDA

- Review the generalized recursion tree method of $T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$
- Master theorem for solving recurrence

RECALL: RECURSION TREE METHOD GENERALIZATION

- Consider recursive function in the form of $T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$.

$$T(n) = \Theta\left(n^{\log_b a}\right) + \sum_{j=0}^{\lfloor \log_b n \rfloor - 1} a^j f(n_j)$$



THE MASTER METHOD

- A "cookbook" method for solving recurrence of the form $\boldsymbol{T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)}$.
- Conditions
  - $\boldsymbol{a} \geq 1$ and $\boldsymbol{b} > 1$ are constants.
  - $\boldsymbol{f(n)}$ is an asymptotically **positive** function.
    - $\boldsymbol{f(n)}$ is said to be asymptotically positive if there exists $n_0 \geq 0$ such that $f(n) > 0$ for all $n \geq n_0$.
  - $\boldsymbol{T(n)}$ is defined on the nonnegative integers

MASTER THEOREM
THE COOKBOOK

- For recurrence $\boldsymbol{T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)}$, where we interpret $\boldsymbol{\frac{n}{b}}$ to mean either $\boldsymbol{\left\lfloor\frac{n}{b}\right\rfloor}$ or $\boldsymbol{\left\lceil\frac{n}{b}\right\rceil}$, $\boldsymbol{T(n)}$ has the following asymptotic bounds:

  Case 1: If $f(n) = O\left(n^{\log_b a - \epsilon}\right)$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.

  Case 2: If $f(n) = \Theta\left(n^{\log_b a}\right)$, then $T(n) = \Theta\left(n^{\log_b a} \cdot \log n\right)$.

  Case 3: If $f(n) = \Omega\left(n^{\log_b a + \epsilon}\right)$ for some constant $\epsilon > 0$, and if $a \cdot f\left(\frac{n}{b}\right) \le c \cdot f(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

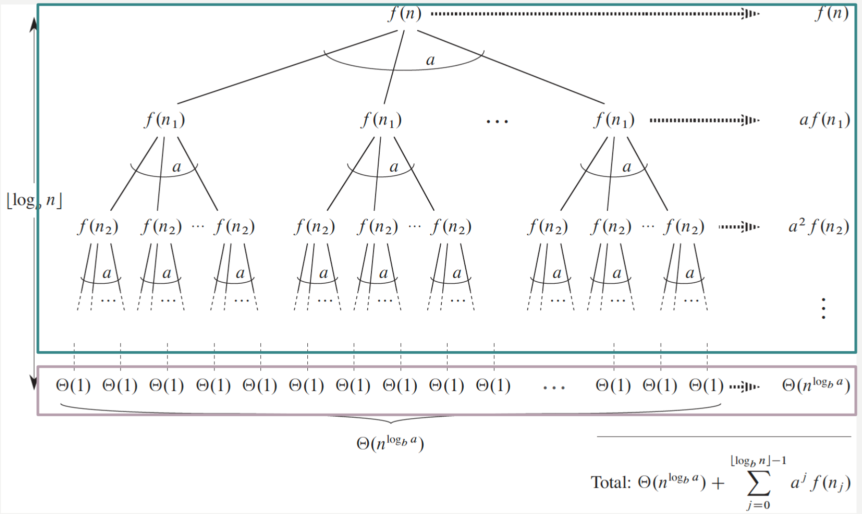
All three cases of master theorem compare $\boldsymbol{f(n)}$ with $\boldsymbol{n^{log_b a}}$.

WHY $n^{\log_b a}$?

- Recall the general format of the running time function derived by the recursion-tree method.

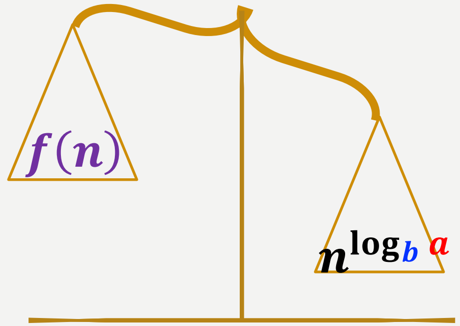
$$T(n) = \Theta\left(n^{\log_b a}\right) + \sum_{j=0}^{\lfloor \log_b n \rfloor - 1} a^j f(n_j)$$

- Essentially, the dominating term between $\Theta\left(n^{\log_b a}\right)$ and $\sum a^j f(n_j)$ will determine the asymptotic bound of $T(n)$.



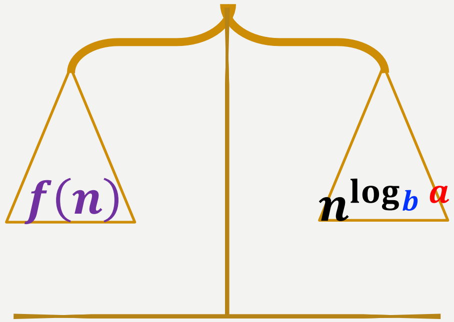
MASTER THEOREM CASE 1 EXPLAINED

- Case 1: If $f(n) = O(n^{\log_b a-\epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
  - Keep in mind that $T(n) = \Theta(n^{\log_b a}) + \sum_{j=0}^{\lfloor \log_b n \rfloor - 1} a^j f(n_j)$
  - The condition $f(n) = O(n^{\log_b a-\epsilon})$ means $f(n)$ **is asymptotically smaller than $n^{\log_b a}$** by a factor of n^{ϵ} for some $\epsilon > 0$.



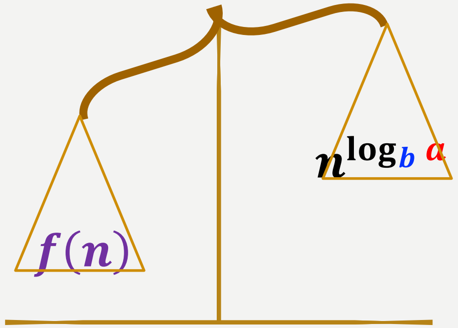
MASTER THEOREM CASE 2 EXPLAINED

- Case 2: If $\boldsymbol{f(n) = \Theta\left(n^{\log_b a}\right)}$, then $T(n) = \Theta\left(n^{\log_b a} \cdot \lg n\right)$.
  - Keep in mind that $\boldsymbol{T(n) = \Theta\left(n^{\log_b a}\right) + \sum_{j=0}^{\lfloor \log_b n \rfloor - 1} a^j f(n_j)}$
  - The condition $\boldsymbol{f(n) = O\left(n^{\log_b a}\right)}$ means $\boldsymbol{f(n)}$ **and** $\boldsymbol{n^{\log_b a}}$ **are about the same size**.



MASTER THEOREM CASE 3 EXPLAINED

- Case 3: If $\boldsymbol{f(n) = \Omega\left(n^{log_b a+\epsilon}\right)}$ for some constant $\epsilon > 0$, and if $\boldsymbol{a \cdot f\left(\frac{n}{b}\right) = c \cdot f(n)}$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$.
  - The condition $f(n) = \Omega\left(n^{\log_b a+\epsilon}\right)$ **means $\boldsymbol{f(n)}$ must be asymptotically larger than $\boldsymbol{n^{\log_b a}}$ by a factor of $\boldsymbol{n^{\epsilon}}$** for some $\epsilon > 0$.
    - **$\boldsymbol{f(n)}$ must be polynomially larger**
    - The regularity condition $a \cdot f\left(\frac{n}{b}\right) = c \cdot f(n)$ restricts the algebraic structure of $f(n)$.



MASTER THEOREM
EXAMPLE - 1

- Recall the running time of binary search algorithm $T(n) = T\left(\frac{n}{2}\right) + \Theta(1)$.
- Solve the recurrence using master theorem.
- **Step 1**: $a =$ $\underline{\mathbf{1}}$ $\geq$ $\underline{\mathbf{1}}$, $b =$ $\underline{\mathbf{2}}$ $>$ $\underline{\mathbf{1}}$, $f(n) =$ $\underline{\boldsymbol{\Theta(1)}}$ that is asymptotically **positive**.
- **Step 2**: $n^{\log_b a} =$ $\underline{\boldsymbol{n^{\lg 1} > n^0}}$, $f(n) =$ $\underline{\boldsymbol{\Theta(n^{\lg 1})}}$~~, where $\epsilon =$ ______~~.
- **Step 3**: Case $\underline{\mathbf{2}}$ of the master theorem can apply.
  - Checking regularity condition if case 3 condition is met.
  - Show that ____________ for constant $c < 1$ and all sufficiently large n.
  - Solve for c, $c =$ ______.
- **Step 4**: $T(n) =$ $\underline{\boldsymbol{\Theta\left(n^{\log_b a} \cdot \lg n\right) = \Theta(\lg n)}}$.

MASTER THEOREM EXAMPLE - 2

- Recall the running time of binary search algorithm $T(n) = 3\left(\frac{n}{4}\right) + n \lg n$.
- Solve the recurrence using master theorem.
- **Step 1**: $a =$ $\underline{\mathbf{3}}$ $\geq$ $\underline{\mathbf{1}}$, $b =$ $\underline{\mathbf{4}}$ $>$ $\underline{\mathbf{1}}$, $f(n) =$ $\underline{(\boldsymbol{n}\,\mathbf{lg}\,\boldsymbol{n})}$ that is asymptotically **positive**.
- **Step 2**: $n^{\log_b a} =$ $\underline{\boldsymbol{n}^{\log_4 3} < \boldsymbol{n}^{1}}$, $f(n) =$ $\underline{\boldsymbol{\Omega}(\boldsymbol{n}^{\log_4 3+\epsilon})}$, where $\epsilon =$ $\underline{\mathbf{1} - \mathbf{log}_4 \mathbf{3} \approx \mathbf{0.2}}$
- **Step 3**: Case $\underline{\mathbf{3}}$ of the master theorem can apply.
  - Checking regularity condition if case 3 condition is met.
  - Show that $\underline{\mathbf{3} \cdot \boldsymbol{f}(\boldsymbol{n}/\mathbf{4}) \leq \boldsymbol{c} \cdot \boldsymbol{f}(\boldsymbol{n})}$ for constant $c < 1$ and all sufficiently large n.
  - Solve for c, $c =$ $\underline{\mathbf{3/4}}$.
- **Step 4**: $T(n) =$ $\underline{\boldsymbol{\Theta}(\boldsymbol{f}(\boldsymbol{n})) = \boldsymbol{\Theta}(\mathbf{nlg}\,\boldsymbol{n})}$.

MASTER THEOREM
PRACTICE 1

- Recall the running time of Strassen's square matrix algorithm $T(n) = \mathbf{7}T\left(\frac{n}{2}\right) + \Theta(n^2)$
- Solve the recurrence using master theorem.
- **Step 1**: $a =$ _____ $\geq$_____, $b =$ _____ $>$_____, $f(n) =$ ______ that is asymptotically __________.
- **Step 2**: $n^{\log_b a} =$ ____________, $f(n) =$ _______________, where $\epsilon =$ _______________.
- **Step 3**: Case _____ of the master theorem can apply.
  - Checking regularity condition if case 3 condition is met.
  - Show that _____________________ for constant $c < 1$ and all sufficiently large n.
  - Solve for c, $c =$ __________.
- **Step 4**: $T(n) =$ ______________________________.

MASTER THEOREM PRACTICE 1

- Recall the running time of Strassen's square matrix algorithm $T(n) = \mathbf{7}T\left(\frac{n}{2}\right) + \Theta(n^2)$
- Solve the recurrence using master theorem.
- **Step 1**: $a =$ __**7**__ $\geq$ __**1**__, $b =$ __**2**__ $>$ __**1**__, $f(n) = \boldsymbol{\Theta(n^2)}$ that is asymptotically __**positive**__.
- **Step 2**: $n^{\log_b a} = \boldsymbol{n^{\lg 7} > n^{\lg 4}}$, $f(n) = \boldsymbol{O(n^{\lg 7 - \epsilon})}$_____, where $\epsilon = \mathbf{lg 7 - lg\ 4 > 0}$.
- **Step 3**: Case __**I**__ of the master theorem can apply.
  - Checking regularity condition if case 3 condition is met.
  - Show that ____________________ for constant $c < 1$ and all sufficiently large n.
  - Solve for c, $c =$ __________.
- **Step 4**: $T(n) = \boldsymbol{\Theta\left(n^{\log_b a}\right) = \Theta(n^{\lg 7})}$________.

MASTER THEOREM
PRACTICE 2

- Solve recurrence $T(n) = 3 \cdot T\left(\frac{n}{4}\right) + n$ using master theorem.
- **Step 1**: $a =$ ____ $\geq$____, $b =$ ____ $>$____, $f(n) =$ _____ that is asymptotically ________.
- **Step 2**: $n^{\log_b a} =$ __________, $f(n) =$ ____________, where $\epsilon =$ ____________.
- **Step 3**: Case ____ of the master theorem can apply.
  - Checking regularity condition if case 3 condition is met.
  - Show that __________________ for constant $c < 1$ and all sufficiently large n.
  - Solve for c, $c =$ ________.
- **Step 4**: $T(n) =$ _________________________.

MASTER THEOREM PRACTICE 2

- Solve recurrence $T(n) = 3 \cdot T\left(\frac{n}{4}\right) + n$ using master theorem.
- **Step 1**: $a =$ <u>**3**</u> $\geq$ <u>**1**</u>, $b =$ <u>**4**</u> $>$ <u>**1**</u>, $f(n) =$ <u>$\boldsymbol{n}$</u> that is asymptotically <u>**positive**</u>.
- **Step 2**: $n^{\log_b a} =$ <u>$\boldsymbol{n^{\log_4 3} < n^1}$</u>, $f(n) =$ <u>$\boldsymbol{\Omega(n^{\log_4 3+\epsilon})}$</u>, where $\epsilon =$ <u>$\boldsymbol{1 - \log_4 3 > 0}$</u>.
- **Step 3**: Case <u>**3**</u> of the master theorem can apply.
  - Checking regularity condition if case 3 condition is met.
  - Show that <u>$\boldsymbol{3 \cdot f(n/4) = c \cdot f(n)}$</u> for constant $c < 1$ and all sufficiently large n.
  - Solve for c, $c =$ <u>$\boldsymbol{3/4 < 1}$</u>.
- **Step 4**: $T(n) =$ <u>$\boldsymbol{\Theta(f(n)) = \Theta(n)}$</u>.

MASTER THEOREM PRACTICE 3

- Recall the running time of merge-sort algorithm $T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$
- Solve the recurrence using master theorem.
- **Step 1**: $a =$ ____ $\geq$____, $b =$ ____ $>$____, $f(n) =$ _____ that is asymptotically ________.
- **Step 2**: $n^{\log_b a} =$ __________, $f(n) =$ ____________, where $\epsilon =$ ____________.
- **Step 3**: Case ____ of the master theorem can apply.
  - Checking regularity condition if case 3 condition is met.
  - Show that _________________ for constant $c < 1$ and all sufficiently large n.
  - Solve for c, $c =$ ________.
- **Step 4**: $T(n) =$ ________________________.

MASTER THEOREM PRACTICE 3

- Recall the running time of merge-sort algorithm $T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$
- Solve the recurrence using master theorem.
- **Step 1**: $a =$ **2** $\geq$ **1**, $b =$ **2** $>$ **1**, $f(n) =$ $\boldsymbol{\Theta(n)}$ that is asymptotically **positive**.
- **Step 2**: $n^{\log_b a} = \boldsymbol{n^{\lg 2} = n^1}$, $f(n) = \boldsymbol{\Theta(n^{\lg 2})}$ ~~, where $\epsilon =$ ________~~.
- **Step 3**: Case **2** of the master theorem can apply.
  - Checking regularity condition if case 3 condition is met.
  - Show that ________________ for constant $c < 1$ and all sufficiently large n.
  - Solve for c, $c =$ ________.
- **Step 4**: $T(n) = \boldsymbol{\Theta\left(n^{\log_b a} \cdot \lg n\right) = \Theta(n \lg n)}$.

NEXT UP
PRUNE-AND-SEARCH

REFERENCE