06 Induction and Recursion

CS201 Discrete Mathematics

Instructor: Shan Chen

Mathematical Induction

We can reach step $k + 1$ if we can reach step k

Step $k + 1$

Step k

Step 4

Step 3

We can reach step 1

Step 2

Step 1

SUSTech
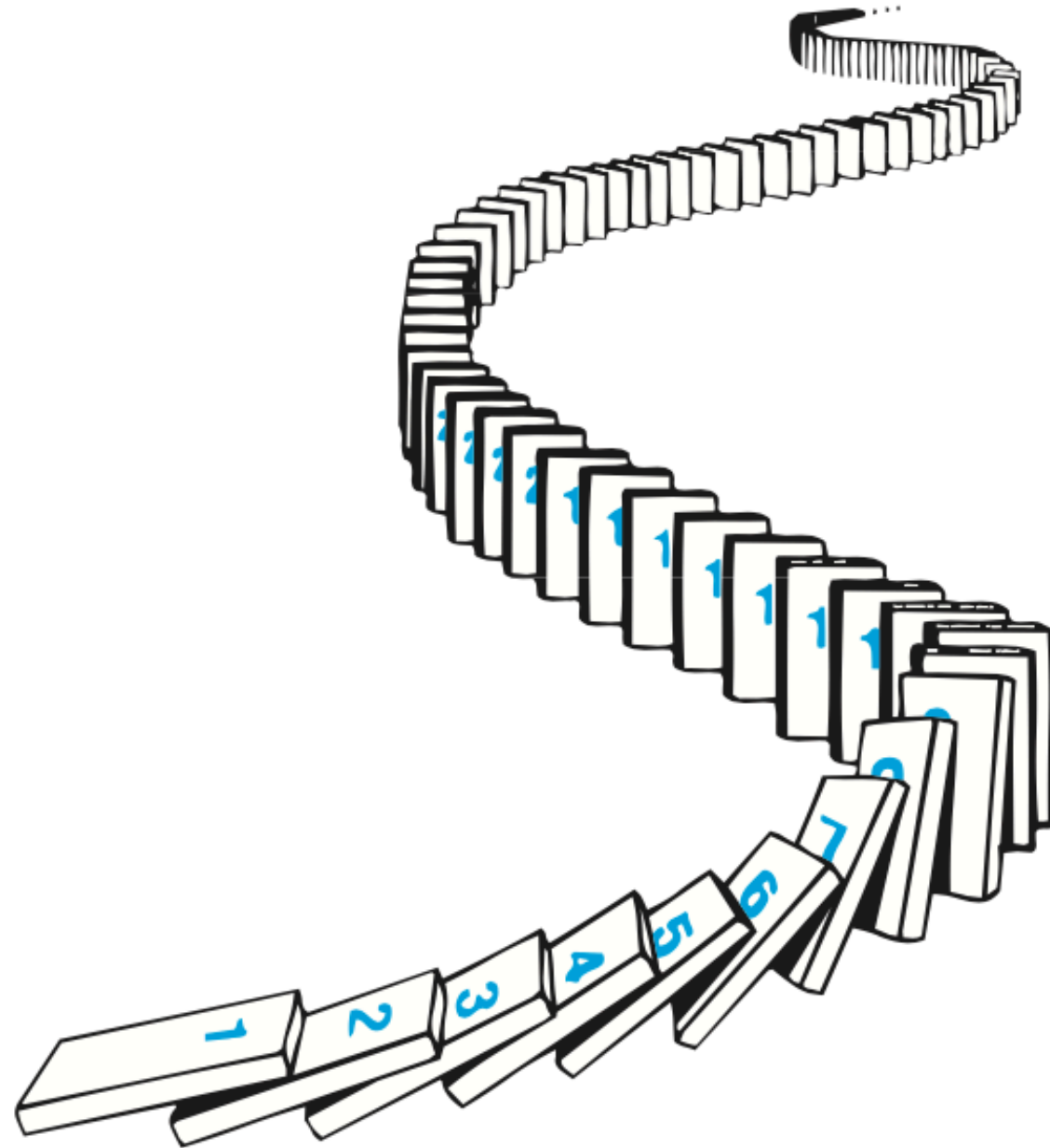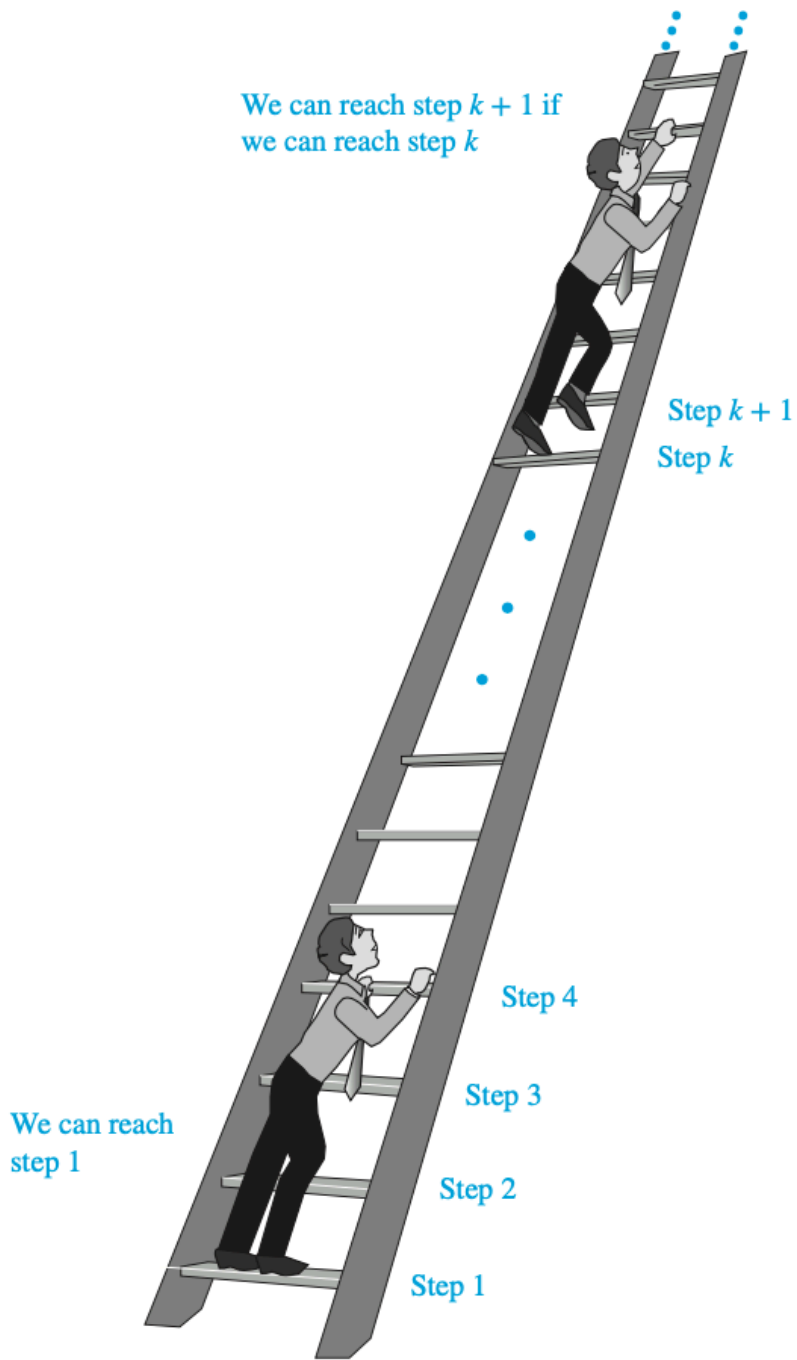
Mathematical Induction

Principle of Mathematical Induction

○ Let *P(n)* be a propositional function, i.e., *P(n)* is either true or false when *n* is a specific number.

○ **Principle of mathematical induction:** To prove that *P(n)* is true for all positive integers *n*, we complete two steps:

  • **Basis step:** prove *P(1)* is true

  • **Inductive step:** prove $\forall k \in \mathbf{Z^+}, P(k) \rightarrow P(k+1)$ is true
    * the assumption "P(k) is true" is called the **inductive hypothesis**

○ **Q:** Why is this principle valid?

○ Proof by contradiction: Assume *P(n)* is false for some integer *n* ≥ *1*, then the set S of all positive integer *n* such that *P(n)* is false is non-empty. Let *m* be the smallest integer in S. * why m exists? We have *m* ≥ *2* as *P(1)* is true. However, since *P(m - 1)* is true and *P(m - 1)* → *P(m)* is true, *P(m)* must be true, contradiction!

Principle of Mathematical Induction

○ **Principle of mathematical induction:** To prove that *P*(*n*) is true for all positive integers *n*, we complete two steps:

- **Basis step:** prove *P*(*1*) is true

- **Inductive step:** prove $\forall k \in \boldsymbol{Z^+}$, *P*(*k*) → *P*(*k* + *1*) is true

○ Proof by contradiction: Assume *P(n)* is false for some integer *n* ≥ *1*, then the set S of all positive integer *n* such that *P(n)* is false is non-empty. Let *m* be the smallest integer in S. ** why m exists?* We have *m* ≥ 2 as *P(1)* is true. However, since *P(m - 1)* is true and *P(m - 1)* → *P(m)* is true, *P(m)* must be true, contradiction!

○ **Well-ordering principle** (axiom): every nonempty subset of the set of positive integers has a least/minimum element.

- this principle is equivalent to principle of mathematical induction ** the proof is left as an exercise*

SUSTech

Example

○ Show that *1 + 2 + ⋯ + n = n(n + 1)/2* for any positive integer *n*.

○ Proof by induction:

- Let *P(n)* be the proposition that the sum of the first *n* positive integers is equal to *n(n + 1)/2*.

- **Basis step:** *P(1)* is true, because *1 = 1(1 + 1)/2*.

- **Inductive step:** From the inductive hypothesis, i.e., *P(k)* is true for an arbitrary positive integer *k*, we need to show that *P(k + 1)* is true, i.e., *1 + 2 + ⋯ + k + 1 = (k + 1)((k + 1) + 1)/2*.

  1 + 2 + ⋯ + k + (k + 1) = k(k + 1)/2 + k + 1
  = (k(k + 1) + 2(k + 1))/2 = (k + 1)(k + 2)/2 = (k + 1)((k + 1) + 1)/2

- By mathematical induction, we know that *P(n)* is true for all positive integers *n*. That is, we have proven that *1 + 2 + ⋯ + n = n(n + 1)/2* holds for all positive integers *n*.

SUSTech

Exercise *(5 mins)*

- For any positive integer *n*, *1 + 3 + 5 + ⋯ + (2n - 1) = ?* Prove it.

  - Show that *1 + 2 + ⋯ + n = n(n + 1)/2* for any positive integer *n*.

  - Proof by induction:

    - Let *P(n)* be the proposition that the sum of the first *n* positive integers is equal to *n(n + 1)/2*.

    - **Basis step:** *P(1)* is true, because *1 = 1(1 + 1)/2*.

    - **Inductive step:** From the inductive hypothesis, i.e., *P(k)* is true for an arbitrary positive integer *k*, we need to show that *P(k + 1)* is true, i.e., *1 + 2 + ⋯ + k + 1 = (k + 1)((k + 1) + 1)/2*.

      1 + 2 + ⋯ + k + (k + 1) = k(k + 1)/2 + k + 1
      = (k(k + 1) + 2(k + 1))/2 = (k + 1)(k + 2)/2 = (k + 1)((k + 1) + 1)/2

    - By mathematical induction, we know that *P(n)* is true for all positive integers *n*. That is, we have proven that *1 + 2 + ⋯ + n = n(n + 1)/2* holds for all positive integers *n*.

SUSTech

Exercise *(5 mins)*

○ For any positive integer *n*, *1 + 3 + 5 + ⋯ + (2n - 1) = ?* Prove it.

○ *1 + 3 + 5 + ⋯ + (2n - 1) = n^2*

- *1 + 3 = 4, 1 + 3 + 5 = 9, …*

○ Proof by induction: (Let *P(n)* be the above equation.)

- **Basis step:** *P(1)* is true, because *1 = 1^2*.

- **Inductive step:** From the inductive hypothesis, i.e., *P(k)* is true for an arbitrary positive integer *k*, we need to show that *P(k + 1)* is true, i.e., *1 + 3 + ⋯ + 2(k + 1) - 1 = $(k + 1)^2$*.

  1 + 3 + ⋯ + 2k - 1 + 2(k + 1) - 1 = k^2 + 2(k + 1) - 1
  = k^2 + 2k + 2 - 1 = k^2 + 2k + 1 = $(k + 1)^2$

- By mathematical induction, we know that *P(n)* is true for all positive integers *n*. That is, we have proven that *1 + 3 + 5 + ⋯ + (2n - 1) = n^2* for all positive integers *n*.

x

Exercise *(5 mins)*

○ Prove that for any integer $n \geq 2$, $2^{n+1} \geq n^2 + 3$

○ Show that $1 + 2 + \cdots + n = n(n+1)/2$ for any positive integer n.

○ Proof by induction:

- Let $P(n)$ be the proposition that the sum of the first n positive integers is equal to $n(n+1)/2$.

- **Basis step:** $P(1)$ is true, because $1 = 1(1+1)/2$.

- **Inductive step:** From the inductive hypothesis, i.e., $P(k)$ is true for an arbitrary positive integer k, we need to show that $P(k+1)$ is true, i.e., $1 + 2 + \cdots + k + 1 = (k+1)((k+1)+1)/2$.

  $1 + 2 + \cdots + k + (k+1) = k(k+1)/2 + k + 1$
  $= (k(k+1) + 2(k+1))/2 = (k+1)(k+2)/2 = (k+1)((k+1)+1)/2$

- By mathematical induction, we know that $P(n)$ is true for all positive integers n. That is, we have proven that $1 + 2 + \cdots + n = n(n+1)/2$ holds for all positive integers n.

SUSTech

Exercise *(5 mins)*

○ Prove that for any integer $n \geq 2$, $2^{n+1} \geq n^2 + 3$

○ Proof by induction:

- Let *P(n)* be $2^{n+1} \geq n^2 + 3$.

- **Basis step:** *P(2)* is true, because $2^{2+1} = 8 \geq 7 = 2^2 + 3$.

- **Inductive step:** From the inductive hypothesis, i.e., *P(k)* is true for an arbitrary integer $k \geq 2$, we need to show that *P(k + 1)* is true:

$$2^{(k+1)+1} = 2 \cdot 2^{k+1} \geq 2(k^2 + 3) = 2k^2 + 6 = (k + 1)^2 - 2k - 1 + k^2 + 6$$
$$= (k + 1)^2 + (k - 1)^2 + 4 \geq (k + 1)^2 + 3$$

- By mathematical induction, *P(n)* is true for all integers $n \geq 2$.

SUSTech

Another Form of Induction

○ We may have another form of mathematical induction as follows:

- First suppose that we have a proof that *P(1) is true*.

- Next suppose that we have a proof that
  $$\forall k \geq 1,\ P(1) \wedge P(2) \wedge \cdots \wedge P(k) \rightarrow P(k + 1)$$

- Then,
  $$P(1) \rightarrow P(2)$$
  $$P(1) \wedge P(2) \rightarrow P(3)$$
  $$P(1) \wedge P(2) \wedge P(3) \rightarrow P(4)$$

  …

- Iterating gives us a proof of *P(n)* for all *n*

SUSTech

Strong Induction

○ **Second principle of mathematical induction:** To prove that $P(n)$ is true for all positive integers n, we complete two steps:

  • **Basis step:** prove $P(1)$ is true

  • **Inductive step:** prove $\forall k \in \mathbf{Z}^+, P(1) \wedge \cdots \wedge P(k) \rightarrow P(k + 1)$ is true
    * the assumption "$P(1) \wedge P(2) \wedge \cdots \wedge P(k)$ is true" is called the **inductive hypothesis**

○ This is called strong induction or complete induction, while the previous principle is called weak or incomplete induction.

○ In practice, strong induction is often easier to apply than its weak form, because the inductive hypothesis is stronger.

○ However, these two forms of induction are actually equivalent.

  • *the proof is left as an exercise*

SUSTech

Example

○ Prove that every positive integer is a power of a prime or the product of powers of primes.

○ Proof:

- **Basis step:** *1* is a power of a prime number, $1 = 2^0$.

- **Inductive step:**

  Inductive hypothesis: every positive integer that is $\leq k$ is a power of a prime or a product of powers of primes.

  If *k + 1* is a prime power, *P(k + 1)* is true. Otherwise, *k + 1* must be a composite, i.e., a product of two smaller positive integers, each of which is, by the inductive hypothesis, a power of a prime or the product of powers of primes. Therefore, *P(k + 1)* is true.

- Finally, by strong induction, every positive integer is a power of a prime or a product of powers of primes.

SUSTech

Mathematical Induction Summary

○ A typical proof by induction, showing that $P(n)$ is true for all integers $n \geq b$, consists of three steps:

  - **Basis step:** prove $P(b)$ is true

  - **Inductive step:** prove one of the following

    $\forall k \geq b, P(k) \rightarrow P(k+1)$ is true **OR**

    $\forall k \geq b, P(b) \wedge \cdots \wedge P(k) \rightarrow P(k+1)$ is true

  - **Conclusion:** based on the (second) principle of mathematical induction, we conclude that $P(n)$ is true for all $n \geq b$.

○ The assumption "$P(k)$ is true" **OR** "$P(1) \wedge P(2) \wedge \cdots \wedge P(k)$ is true" is called the inductive hypothesis (IH).

  - IH is used to prove "$P(k+1)$ is true".

SUSTech

Recursion

Recursion

○ **Recursion:** a method of solving a computational problem where its solution depends on solutions to smaller instances of the same problem.

○ Recursive computer programs or algorithms often lead to inductive analysis.

○ A classical example of recursion is the Towers of Hanoi Problem.
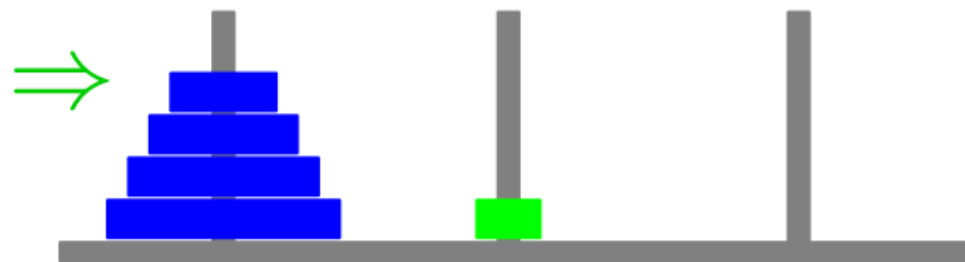
SUSTech

Towers of Hanoi

○ **Problem:** Find an efficient way to move all of the disks from one peg to another.

- *3* pegs and *n* disks of different sizes

- A legal move takes a disk from one peg and moves it onto another peg so that it is not on top of a smaller disk.

Towers of Hanoi

○ **Problem:** Find an efficient way to move all of the disks from one peg to another.

- *3* pegs and *n* disks of different sizes

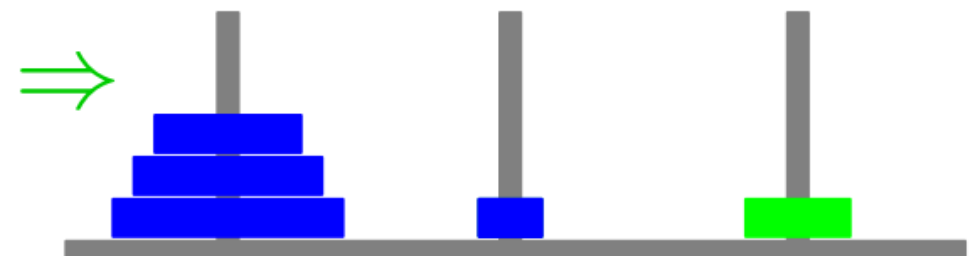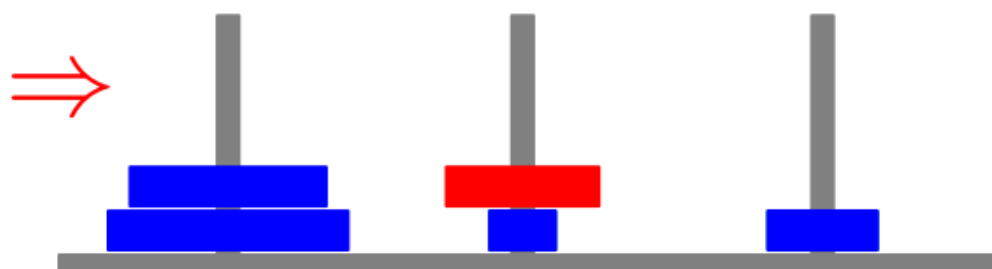- A legal move takes a disk from one peg and moves it onto another peg so that it is not on top of a smaller disk.

SUSTech

Towers of Hanoi

○ **Problem:** Find an efficient way to move all of the disks from one peg to another, using only legal moves.

SUSTech

Towers of Hanoi

○ **Problem:** Find an efficient way to move all of the disks from one peg to another, using only legal moves.

○ **Solution** by recursion:

- **Basis step:** If *n = 1*, moving one disk from one to another is easy.

- **Recursive step:** If *n > 1*, we need three steps:
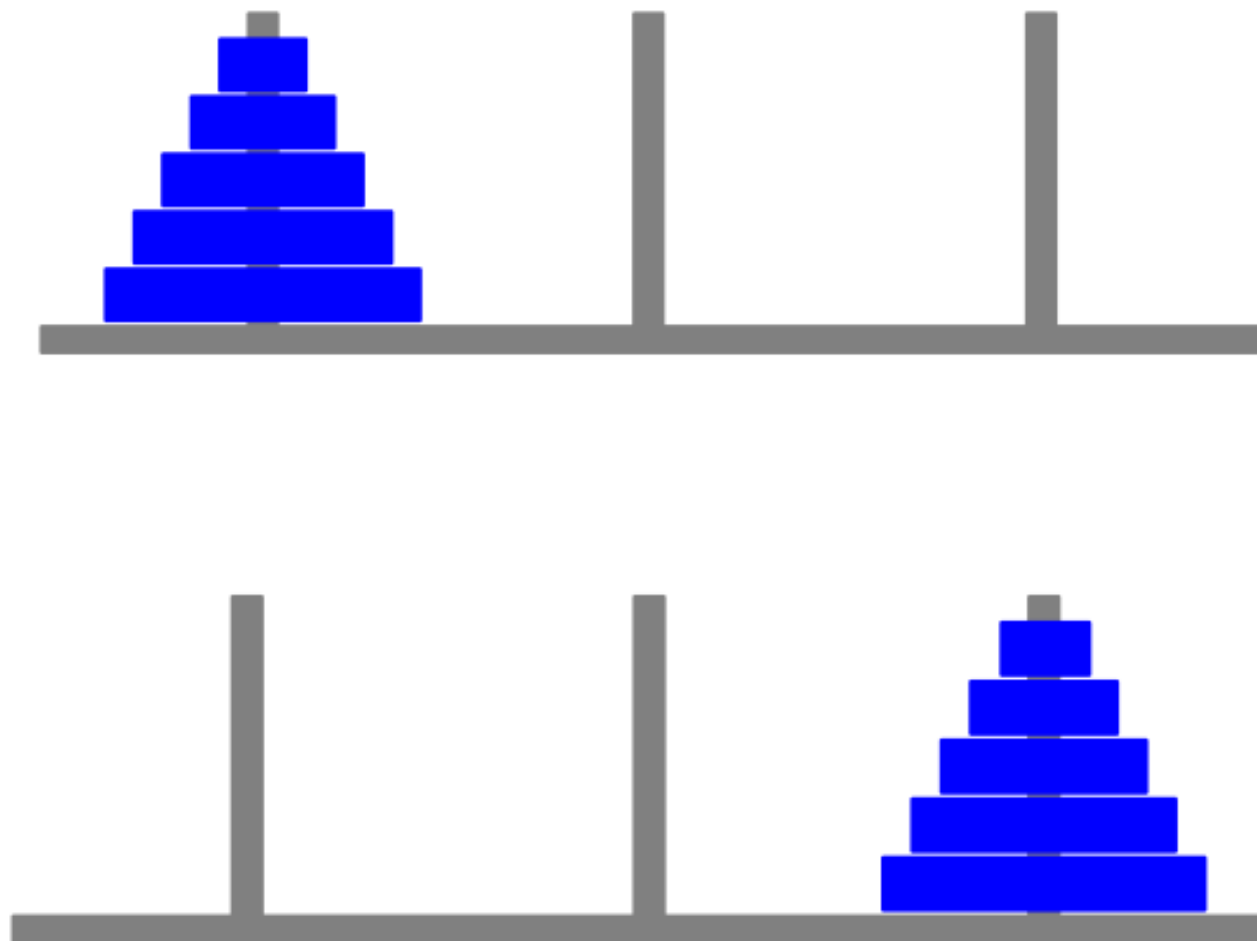
Towers of Hanoi

- **Problem:** Find an efficient way to move all of the disks from one peg to another, using only legal moves.

- **Solution** by recursion:

```java
3  public class Hanoi
4  {
5
6      public void move(int n, char a, char b, char c)
7      {
8          if (n == 1)
9              System.out.println("plate " + n + " from " + a + " to " + c);
10         else
11         {
12             move(n-1,a,c,b);
13             System.out.println("plate " + n + " from " + a + " to " + c);
14             move(n-1,b,a,c);
15         }
16     }
17 }
18
```

move n disks from peg a to peg c using peg b

SUSTech

Towers of Hanoi

○ **Problem:** Find an efficient way to move all of the disks from one peg to another, using only legal moves.

○ **Proof of correctness** by induction:

- Let *P(n)* be the proposition that the solution is correct for *n*.

- **Basis step:** *P(1)* is obviously true, i.e., the solution is correct when there is only one disk.

- **Inductive step:** From the inductive hypothesis, i.e., *P(k)* is true for an arbitrary positive integer *k*, we need to show that *P(k + 1)* is true. That is, if our solution works for *k* disks, then we can build a correct solution for *k + 1* disks, which is true by the recursive step:

- By mathematical induction, *P(n)* is true for all positive integer *n*.

Towers of Hanoi

○ **Problem:** Find an efficient way to move all of the disks from one peg to another, using only legal moves.

○ **Solving the running time:** *# disk moves M(n) = ?*

- **Basis step:** If *n = 1*, moving one disk from one to another is easy.

M(1) = 1

- **Recursive step:** If *n > 1*, we need three steps:

1) 2) 3)

M(n) = 2M(n − 1) + 1 for n > 1

SUSTech

Towers of Hanoi

○ **Problem:** Find an efficient way to move all of the disks from one peg to another, using only legal moves.

○ **Solving the running time:**

$$M(1) = 1 \qquad M(n) = 2M(n - 1) + 1 \text{ for } n > 1$$

- Iterating the above function gives:

  $M(1) = 1, M(2) = 3, M(3) = 7, M(4) = 15, M(5) = 31, \dots$

- We can guess that $M(n) = 2^n - 1$ and prove it by induction:

  Let $P(n)$ denote the above equation.

  Basis step: $P(1)$ is true, because $M(1) = 1 = 2^1 - 1$.

  Inductive step: Assume $P(k)$ is true for $k \geq 1$, i.e., $M(k) = 2^k - 1$.
  Then $P(k + 1)$ is true: $M(k + 1) = 2M(k) + 1 = 2(2^k - 1) + 1 = 2^{k+1} - 1$

  By mathematical induction, $P(n)$ is true for all positive n.

SUSTech

Towers of Hanoi

○ **Problem:** Find an efficient way to move all of the disks from one peg to another, using only legal moves.

○ Note that we applied induction twice:

- first time to prove **correctness** of the solution

- second time to derive the **closed-form running time** $M(n) = 2^n - 1$

SUSTech

Recurrence Relations

Recurrence Relations

- A **recurrence relation** or **recurrence** for a function defined on the set of integers $\geq b$ tells us how to compute the n-th value from some or all the first $n - 1$ values.

- To completely specify a function defined by a recurrence, we have to give the initial condition(s) (as known as the base case(s)) for the recurrence.

- Example: running time for Towers of Hanoi

$$M(n) = \begin{cases} 1 & \text{if } n = 1 \\ 2M(n-1) + 1 & \text{otherwise} \end{cases}$$

SUSTech

Example

○ Let *S(n)* be the number of subsets of a set of size *n*. We already learned that *S(n) = 2^n*, but now let us think about this recursively:

- Consider the *8* subsets of *{1, 2, 3}*:

$$\varnothing \qquad \{1\} \qquad \{2\} \qquad \{1, 2\}$$

$$\{3\} \qquad \{1, 3\} \qquad \{2, 3\} \qquad \{1, 2, 3\}$$

- The top *4* subsets are exactly the subsets of *{1, 2}*, while the bottom *4* subsets are the subsets of *{1, 2}* with *3* added into each.

- So, we get a subset of *{1, 2, 3}* either by taking a subset of *{1, 2}* or by adding *3* to a subset of *{1, 2}*.

- This suggests that the recurrence relation for the number of subsets of an *n*-element set *{1, 2, …, n}* is

$$S(n) = \begin{cases} 1 & \text{if } n = 0 \\ 2S(n-1) & \text{if } n \geq 1 \end{cases}$$

SUSTech

Example

- Let *S(n)* be the number of subsets of a set of size *n*. We already learned that *S(n) = 2^n*, but now let us think about this recursively:

- **Proof of correctness** of the recurrence:

$$S(n) = \begin{cases} 1 & \text{if } n = 0 \\ 2S(n-1) & \text{if } n \geq 1 \end{cases}$$

- The subsets of *{1, 2, . . . , n}* can be partitioned according to whether or not they contain the element *n*.

  Each subset *S* containing *n* can be constructed in a unique fashion by adding *n* to the subset *S – {n}* that does not contain *n*.

  Each subset *S* not containing *n* can be constructed by removing *n* from the unique set *S ∪ {n}* that contains n.

  So, the number of subsets containing *n* is exactly the same as the number of subsets not containing n.

- Therefore, if *n ≥ 1*, then *S(n) = 2S(n – 1)*.

SUSTech

Example

○ Let $S(n)$ be the number of subsets of a set of size n. We already learned that $S(n) = 2^n$, but now let us think about this recursively:

○ **Proof of the closed form** $S(n) = 2^n$ for recurrence:

$$S(n) = \begin{cases} 1 & \text{if } n = 0 \\ 2S(n-1) & \text{if } n \geq 1 \end{cases}$$

- *the proof (by induction) is left as an exercise*

SUSTech

Iterating a Recurrence

○ Let $T(n) = rT(n - 1) + a$, where r and a are constants.

○ Find a recurrence relation that expresses

$T(n)$ in terms of $T(n - 2)$

$T(n)$ in terms of $T(n - 3)$

$T(n)$ in terms of $T(n - 4)$

…

○ Can we generalize this to find a closed-form solution to $T(n)$?

SUSTech

Iterating a Recurrence

○ Note that $T(n) = rT(n-1) + a$ implies that

- $\forall i < n, \ T(n-i) = rT((n-i)-1)) + a$

○ Then, we have

$$
\begin{aligned}
T(n) &= rT(n-1) + a \\
&= r(rT(n-2) + a) + a \\
&= r^2T(n-2) + ra + a \\
&= r^2(rT(n-3) + a) + ra + a \\
&= r^3T(n-3) + r^2a + ra + a \\
&= r^3(rT(n-4) + a) + r^2a + ra + a \\
&= r^4T(n-4) + r^3a + r^2a + ra + a.
\end{aligned}
$$

○ Guess: $T(n) = r^nT(0) + a\sum_{i=0}^{n-1} r^i$

SUSTech

Iterating a Recurrence

○ The method we used to guess the solution is called iterating the recurrence, because we iteratively use the recurrence.

○ Another approach is to iterate from the "bottom-up" instead of "top-down".

- E.g., $T(n) = rT(n-1) + a$, where r and a are constants.

$$T(0) = b$$
$$T(1) = rT(0) + a = rb + a$$
$$T(2) = rT(1) + a = r(rb + a) + a = r^2 b + ra + a$$
$$T(3) = rT(2) + a = r^3 b + r^2 a + ra + a$$

- This would lead to the same guess: $T(n) = r^n T(0) + a \sum_{i=0}^{n-1} r^i$

SUSTech

Formula of Recurrences

○ **Theorem:** If *T(n) = rT(n − 1) + a*, *T(0) = b*, and *r ≠ 1*, then for all non-negative integers *n*, we have:

$$T(n) = r^n b + a\frac{1 - r^n}{1 - r}$$

○ Proof by induction:

- **Basis step:** The formula is true for *n = 0*: $T(0) = r^0 b + a\frac{1 - r^0}{1 - r} = b.$

- **Inductive step:**   ?

SUSTech

Formula of Recurrences

- **Theorem:** If $T(n) = rT(n-1) + a$, $T(0) = b$, and $r \neq 1$, then for all non-negative integers n, we have:

$$T(n) = r^n b + a\frac{1 - r^n}{1 - r}$$

- Proof by induction:

  - **Basis step:** The formula is true for $n = 0$: $T(0) = r^0 b + a\frac{1 - r^0}{1 - r} = b$.

  - **Inductive step:**
  $$
  \begin{aligned}
  T(n) &= rT(n-1) + a \\
  &= r\left(r^{n-1}b + a\frac{1 - r^{n-1}}{1 - r}\right) + a \\
  &= r^n b + \frac{ar - ar^n}{1 - r} + a \\
  &= r^n b + \frac{ar - ar^n + a - ar}{1 - r} \\
  &= r^n b + a\frac{1 - r^n}{1 - r}.
  \end{aligned}
  $$

SUSTech

Formula of Recurrences

○ **Theorem:** If $T(n) = rT(n-1) + a$, $T(0) = b$, and $r \neq 1$, then for all non-negative integers n, we have:

$$T(n) = r^n b + a \frac{1 - r^n}{1 - r}$$

○ Example: $T(n) = 3T(n-1) + 2$, $T(0) = 5$

  • Plugging $r = 3, a = 2, b = 5$ in the formula, we have:

$$T(n) = 3^n \cdot 5 + 2 \frac{1 - 3^n}{1 - 3} = 3^n \cdot 6 - 1$$

SUSTech

First-Order Linear Recurrences

○ A recurrence of the form $T(n) = f(n)T(n-1) + g(n)$ is called a **first-order linear** recurrence.

  • **first order**: $T(n)$ depends upon going back one step, i.e., $T(n-1)$
    e.g., $T(n) = T(n-1) + 2T(n-2)$ is a second-order recurrence

  • **linear**: the $T(n-1)$ only appears to the first power.
    e.g., $T(n) = (T(n-1))^2 + 3$ is a non-linear first-order recurrence

○ When $f(n)$ is a constant, say r, the general solution is almost as easy as the one we derived before. Iterating the recurrence gives

$$
\begin{aligned}
T(n) &= rT(n-1) + g(n) \\
&= r(rT(n-2) + g(n-1)) + g(n) \\
&= r^2 T(n-2) + rg(n-1) + g(n) \\
&\vdots \\
&= r^n T(0) + \sum_{i=0}^{n-1} r^i g(n-i)
\end{aligned}
$$

SUSTech

First-Order Linear Recurrences

○ **Theorem:** For any positive constants a and r, and any function g defined on positive integers, the solution to the first-order linear recurrence

$$T(n) = \begin{cases} rT(n-1) + g(n) & \text{if } n > 0 \\ a & \text{if } n = 0 \end{cases}$$

is

$$T(n) = r^n a + \sum_{i=1}^{n} r^{n-i} g(i).$$

Exercise *(5 mins)*

- Solve *T(n) = 4T(n − 1) + 2^n (n > 0)* with *T(0) = 6*

  Hint: write T(n) in terms of 4^n and 2^n

- **Theorem:** For any positive constants *a* and *r*, and any function *g* defined on nonnegative integers, the solution to the first-order linear recurrence

$$T(n) = \begin{cases} rT(n-1) + g(n) & \text{if } n > 0 \\ a & \text{if } n = 0 \end{cases}$$

is

$$T(n) = r^n a + \sum_{i=1}^{n} r^{n-i} g(i).$$

SUSTech

Exercise *(5 mins)*

○ Solve *T(n) = 4T(n − 1) + 2^n (n > 0)* with *T(0) = 6*

Hint: write T(n) in terms of 4^n and 2^n

$$
\begin{aligned}
T(n) &= 6 \cdot 4^n + \sum_{i=1}^{n} 4^{n-i} \cdot 2^i \\
&= 6 \cdot 4^n + 4^n \sum_{i=1}^{n} 4^{-i} \cdot 2^i \\
&= 6 \cdot 4^n + 4^n \sum_{i=1}^{n} \left(\frac{1}{2}\right)^i \\
&= 6 \cdot 4^n + \left(1 - \frac{1}{2^n}\right) \cdot 4^n \\
&= 7 \cdot 4^n - 2^n.
\end{aligned}
$$

SUSTech

Divide-and-Conquer Recurrences

Divide and Conquer

○ **Divide-and-conquer:** recursively breaks down a problem into two or more sub-problems of the same or related type, until these become simple enough to be solved directly; the solutions to the sub-problems are then combined to give a solution to the original problem.

○ Divide-and-conquer recurrence relations are usually of the form:

$$T(n) = \begin{cases} \text{something given} & \text{if } n \leq n_0 \\ r \cdot T(n/m) + g(n) & \text{if } n > n_0 \end{cases}$$
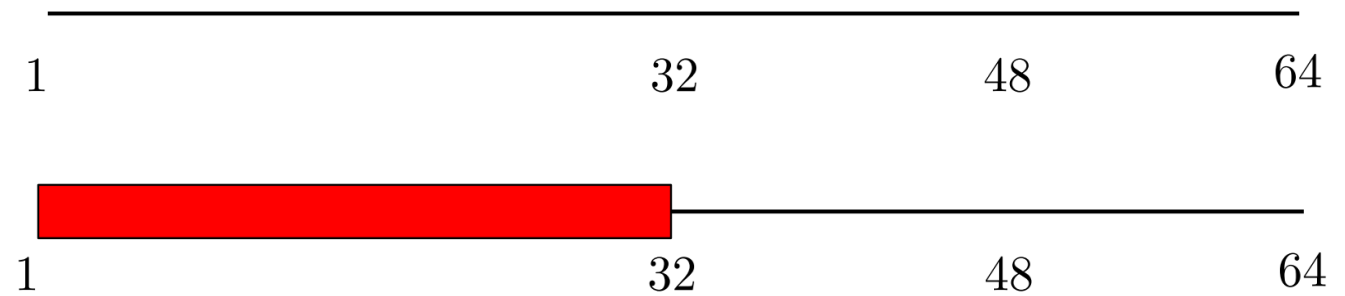
SUSTech

Binary Search

○ Someone has chosen a number *x* between *1* and *n*. We need to discover *x*.

○ We only need to ask two types of questions:

- Is *x* greater than *k*?

- Is *x* equal to *k*?

○ Our strategy will be to always ask greater than questions, at each step halving our search range, until the range only contains one number, when we ask a final equal to question.
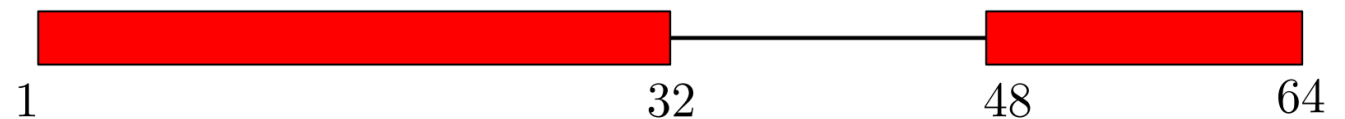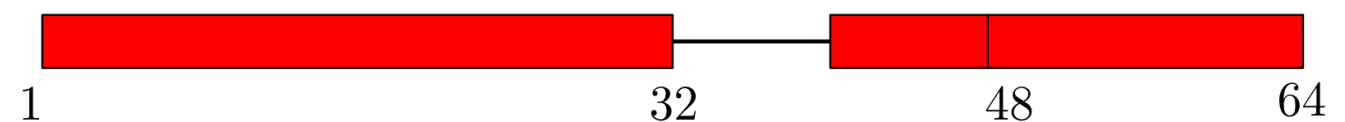
Binary Search

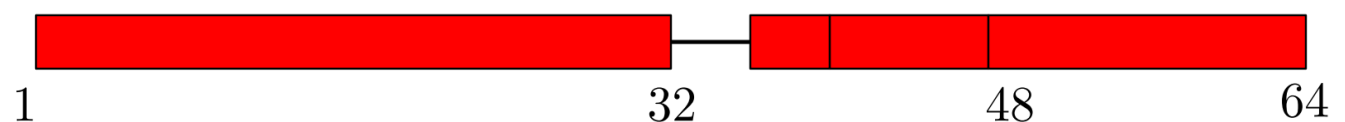○ Example: *n = 64, x = 35*

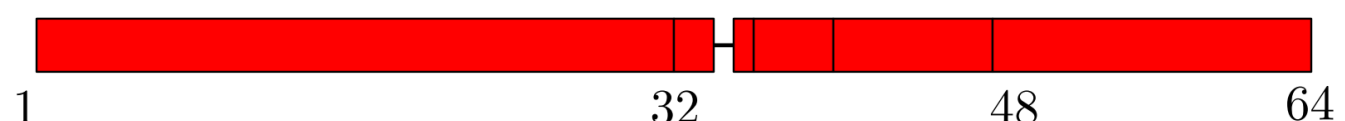| x > 32? | Yes |
x > 48?	No
x > 40?	No
x > 36?	No
x > 34?	Yes
x > 35?	No
x = 35?	Yes
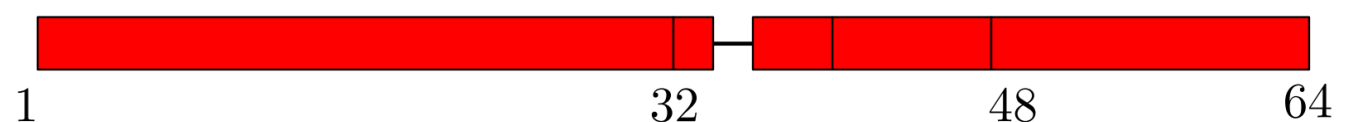
Binary Search

- **Idea:** each guess reduces the size of the problem to only half as big, then we can (recursively) conquer this smaller problem.

- When *n is a power of 2*, the number of comparisons *T(n)* in a binary search on *[1, n]*, satisfies

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \\ T(n/2) + 1 & \text{if } n \geq 2 \end{cases}$$

- The correctness proof is similar to that of Towers of Hanoi:

  - **Basis step** (*n = 1*): only one "equal to" comparison is needed

  - **Inductive step** (*n ≥ 2*): one "great than" comparison + time to perform binary search on the remaining *n/2 terms*

SUSTech

Iterating Recurrences

- **Example 1:**

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \\ T(n/2) + 1 & \text{if } n \geq 2 \end{cases}$$

- Solve it by algebraically iterating the recurrence:

$$T(n) = T\left(\frac{n}{2}\right) + 1 \qquad = \left(T\left(\frac{n}{2^2}\right) + 1\right) + 1$$

$$= T\left(\frac{n}{2^2}\right) + 2 \qquad = \left(T\left(\frac{n}{2^3}\right) + 1\right) + 2$$

$$= T\left(\frac{n}{2^3}\right) + 3$$

$$\vdots \qquad \vdots$$

$$= T\left(\frac{n}{2^i}\right) + i$$

end when $i = log_2\, n$

$$\vdots \qquad \vdots$$

$$= T\left(\frac{n}{2^{\log_2 n}}\right) + \log_2 n \ = 1 + \log_2 n$$

SUSTech

Iterating Recurrences

○ **Example 1:**

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \\ T(n/2) + 1 & \text{if } n \geq 2 \end{cases}$$

○ We just showed by algebraically iterating the recurrence that the solution is *T(n) = 1 + log₂ n*.

○ **Note:** Technically, we still need to use induction to prove that our solution is correct. Practically, we almost never explicitly perform this step, since it is obvious how the induction would work.

SUSTech

Iterating Recurrences

- **Example 2:**

$$T(n) = \begin{cases} T(1) & \text{if } n = 1 \\ 2T(n/2) + n & \text{if } n \geq 2 \end{cases}$$

- This corresponds to solving a problem of size *n*, by

    (i) solving *2* subproblems of size *n/2*

    (ii) doing *n* units of additional work

    or using *T(1)* work for "bottom" case of *n = 1*

    * this is exactly how merge sort (from an algorithm course) works

- How to solve it by algebraically iterating the recurrence?

SUSTech

Iterating Recurrences

- **Example 2:**

$$T(n) = \begin{cases} T(1) & \text{if } n = 1 \\ 2T(n/2) + n & \text{if } n \geq 2 \end{cases}$$

- Solve it by algebraically iterating the recurrence:

$$T(n) = 2T\left(\frac{n}{2}\right) + n \quad = 2\left(2T\left(\frac{n}{4}\right) + \frac{n}{2}\right) + n$$

$$= 4T\left(\frac{n}{4}\right) + 2n \quad = 4\left(2T\left(\frac{n}{8}\right) + \frac{n}{4}\right) + 2n$$

$$= 8T\left(\frac{n}{8}\right) + 3n$$

$$\vdots \qquad \vdots$$

$$= 2^i T\left(\frac{n}{2^i}\right) + in$$

$$\vdots \qquad \vdots$$

$$= 2^{\log_2 n} T\left(\frac{n}{2^{\log_2 n}}\right) + (\log_2 n)n = nT(1) + n\log_2 n$$

SUSTech

Iterating Recurrences

○ **Example 3:**

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \\ T(n/2) + n & \text{if } n \geq 2 \end{cases}$$

○ Solve it by algebraically iterating the recurrence:

$$T(n) = T\left(\frac{n}{2}\right) + n$$

$$= T\left(\frac{n}{2^2}\right) + \frac{n}{2} + n$$

$$= T\left(\frac{n}{2^3}\right) + \frac{n}{2^2} + \frac{n}{2} + n$$

$$\vdots \qquad \vdots$$

$$= T\left(\frac{n}{2^i}\right) + \frac{n}{2^{i-1}} + \cdots + \frac{n}{2^2} + \frac{n}{2} + n$$

$$\vdots \qquad \vdots$$

$$= T\left(\frac{n}{2^{\log_2 n}}\right) + \frac{n}{2^{\log_2 n - 1}} + \cdots + \frac{n}{2^2} + \frac{n}{2} + n$$

$$= 1 + 2 + 2^2 + \cdots + \frac{n}{2^2} + \frac{n}{2} + n = 2n - 1$$

SUSTech

Exercise *(5 mins)*

○ Solve this recurrence by algebraically iterating the recurrence:

$$T(n) = \begin{cases} 1 & \text{if } n < 3 \\ 3T(n/3) + n & \text{if } n \geq 3 \end{cases}$$

○ Example 3:

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \\ T(n/2) + n & \text{if } n \geq 2 \end{cases}$$

○ Solve it by algebraically iterating the recurrence:

$$\begin{aligned} T(n) &= T\left(\frac{n}{2}\right) + n \\ &= T\left(\frac{n}{2^2}\right) + \frac{n}{2} + n \\ &= T\left(\frac{n}{2^3}\right) + \frac{n}{2^2} + \frac{n}{2} + n \\ &\quad\vdots\qquad\vdots \\ &= T\left(\frac{n}{2^i}\right) + \frac{n}{2^{i-1}} + \cdots + \frac{n}{2^2} + \frac{n}{2} + n \\ &\quad\vdots\qquad\vdots \\ &= T\left(\frac{n}{2^{\log_2 n}}\right) + \frac{n}{2^{\log_2 n - 1}} + \cdots + \frac{n}{2^2} + \frac{n}{2} + n \\ &= 1 + 2 + 2^2 + \cdots + \frac{n}{2^2} + \frac{n}{2} + n = 2n - 1 \end{aligned}$$

SUSTech

Exercise *(5 mins)*

○ Solve this recurrence by algebraically iterating the recurrence:

$$T(n) = \begin{cases} 1 & \text{if } n < 3 \\ 3T(n/3) + n & \text{if } n \geq 3 \end{cases}$$

○ Solution:

$$T(n) = 3T\left(\frac{n}{3}\right) + n \qquad = 3\left(3T\left(\frac{n}{3^2}\right) + \frac{n}{3}\right) + n$$

$$= 3^2 T\left(\frac{n}{3^2}\right) + 2n \qquad = 3^2\left(3T\left(\frac{n}{3^3}\right) + \frac{n}{3^2}\right) + 2n$$

$$= 3^3 T\left(\frac{n}{3^3}\right) + 3n$$

$$\vdots \qquad \vdots$$

$$= 3^i T\left(\frac{n}{3^i}\right) + in$$

$$\vdots \qquad \vdots$$

$$= 3^{\log_3 n} T\left(\frac{n}{3^{\log_3 n}}\right) + n\log_3 n \qquad = n + n\log_3 n$$

x

SUSTech

Iterating Recurrences

○ **Example 4:**

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \\ 4T(n/2) + n & \text{if } n \geq 2 \end{cases}$$

○ Solve it by algebraically iterating the recurrence:

$$T(n) = 4T\left(\frac{n}{2}\right) + n \qquad\qquad = 4\left(4T\left(\frac{n}{2^2}\right) + \frac{n}{2}\right) + n$$

$$= 4^2 T\left(\frac{n}{2^2}\right) + \frac{4}{2}n + n \quad = 4^2\left(4T\left(\frac{n}{2^3}\right) + \frac{n}{2^2}\right) + \frac{4}{2}n + n$$

$$= 4^3 T\left(\frac{n}{2^3}\right) + \frac{4^2}{2^2}n + \frac{4}{2}n + n$$

$$\vdots \qquad \vdots$$

$$= 4^i T\left(\frac{n}{2^i}\right) + \frac{4^{i-1}}{2^{i-1}}n + \cdots + \frac{4^2}{2^2}n + n$$

$$\vdots \qquad \vdots$$

$$= 4^{\log_2 n} T\left(\frac{n}{2^{\log_2 n}}\right) + \frac{4^{\log_2 n - 1}}{2^{\log_2 n - 1}}n + \cdots + \frac{4}{2}n + n = 2n^2 - n$$

SUSTech

Three Different Behaviors

○ Compare the iteration for the following recurrences ($T(1) = 1$):

- $T(n) = T(n/2) + n$ $T(n) = 2n - 1 = \Theta(n)$

- $T(n) = 2T(n/2) + n$ $T(n) = n + n \log_2 n = \Theta(n \log n)$

- $T(n) = 4T(n/2) + n$ $T(n) = 2n^2 - n = \Theta(n^2)$

- All three recurrences iterate $log_2\ n$ times. The size of subproblem in next iteration is half the size in the preceding iteration level.

○ **Theorem:** Suppose that we have a recurrence $T(n) = aT(n/2) + n$, where $a \geq 1$ and $T(1) = \Theta(1)$. Then we have the following big Θ bounds on the solution:

- If $1 \leq a < 2$, then $T(n) = \Theta(n)$. *this proof is left as an exercise*

- If $a = 2$, then $T(n) = \Theta(n \log n)$. *already proved in Example 2*

- If $a > 2$, then $T(n) = \Theta(n^{\log_2 a})$. *now let us prove this*

SUSTech

Three Different Behaviors

- Let $n = 2^i$. Prove the last case: If $a > 2$, then $T(n) = \Theta(n^{\log_2 a})$.

- Iterating $T(n) = aT(n/2) + n$ as in **Example 4** gives:

$$T(n) = a^i T\left(\frac{n}{2^i}\right) + \left(\frac{a^{i-1}}{2^{i-1}} + \frac{a^{i-2}}{2^{i-2}} + \cdots \frac{a}{2} + 1\right) n$$

$$= a^{\log_2 n} T(1) + n \sum_{i=0}^{\log_2 n - 1} \left(\frac{a}{2}\right)^i$$

- For $r > 1$, the sum $1 + r + r^2 + \ldots + r^n$ grows as fast as its largest item r^n * *think why?* Therefore, for $a/2 > 1$, we have:

$$n \sum_{i=0}^{\log_2 n - 1} \left(\frac{a}{2}\right)^i = n\Theta\left((a/2)^{\log_2 n - 1}\right)$$

SUSTech

Three Different Behaviors

○ Let $n = 2^i$. Prove the last case: If $a > 2$, then $T(n) = \Theta(n^{\log_2 a})$.

○ Iterating $T(n) = aT(n/2) + n$ as in **Example 4** gives:

$$T(n) = a^{\log_2 n} T(1) + n\Theta((a/2)^{\log_2 n - 1})$$

○ Now, for $a > 2$, we also have:

$$n\left(\frac{a}{2}\right)^{\log_2 n - 1} = \frac{2}{a} \cdot \frac{n \cdot a^{\log_2 n}}{2^{\log_2 n}} = \frac{2}{a} \cdot \frac{n \cdot a^{\log_2 n}}{n} = \frac{2}{a} \cdot a^{\log_2 n}$$

$$\boxed{a^{\log_2 n} = \left(2^{\log_2 a}\right)^{\log_2 n} = \left(2^{\log_2 n}\right)^{\log_2 a} = n^{\log_2 a}}$$

○ Finally: $T(n) = a^{\log_2 n} T(1) + n\Theta((a/2)^{\log_2 n - 1}) = \Theta\left(n^{\log_2 a}\right)$

SUSTech

Three Different Behaviors

○ **Theorem:** Suppose that we have a recurrence $T(n) = aT(n/2) + n$, where $a \geq 1$ and $T(1) = \Theta(1)$. Then we have the following big Θ bounds on the solution:

- If $1 \leq a < 2$, then $T(n) = \Theta(n)$. ** this proof is left as an exercise*

- If $a = 2$, then $T(n) = \Theta(n \log n)$. ** already proved in Example 2*

- If $a > 2$, then $T(n) = \Theta(n^{\log_2 a})$. ** just proved*

○ **Master theorem:** Consider a recurrence $T(n) = aT(n/b) + cn^d$, where $a \geq 1, c > 0, d \geq 0$, integer $b \geq 2$, and $T(1) = \Theta(1)$. Then we have the following big Θ bounds on the solution:

- If $1 \leq a < b^d$, then $T(n) = \Theta(n^d)$.

- If $a = b^d$, then $T(n) = \Theta(n^d \log n)$.

- If $a > b^d$, then $T(n) = \Theta(n^{\log_b a})$.

SUSTech

07 Counting

To be continued…

Announcement

○ Assignment 4 was already released and is due on Dec 7:

- 100 points maximum but 110 in total

- DO NOT CHEAT!

SUSTech