موسسه آموز ش عالی ماد

تحلیل و طراحی الگوریتم ها

فصل اول(قسمت دوم):

الگوریتم های بازگشتی
Recursive algorithms

سید جبار بارخدا
barkhoda@gmail.com

مهر ۹۲

رئوس مطالب درس

- الگوریتم های بازگشتی
- تحلیل الگوریتم های بازگشتی
- حل معادلات بازگشتی
  - استقرا(induction)
  - جایگذاری با تکرار
  - قضیه اصلی
  - درخت بازگشتی
  - معادله مشخصه برای حل معادلات همگن
  - سری مولد
  - حل روابط بازگشتی ناهمگن

الگوریتم های بازگشتی

- رویکردهای نوشتن الگوریتم های تکراری
  - تکرار و حلقه
  - الگوریتم بازگشتی

- الگوریتم بازگشتی
  - فرایندی تکراری که در آن، یک الگوریتم خودش را فراخوانی می کند.
- نحوه اجرای الگوریتم بازگشتی
  - عمل فراخوانی
    - قرارگیری متغیرهای محلی و مقادیر آنها و آدرس بازگشت در پشته
    - انتقال پارامترها
    - انتقال کنترل برنامه به ابتدای پردازه جدید
  - بازگشت از یک فراخوانی
    - آدرس بازگشت و ادامه اجرا

الگوریتم های بازگشتی

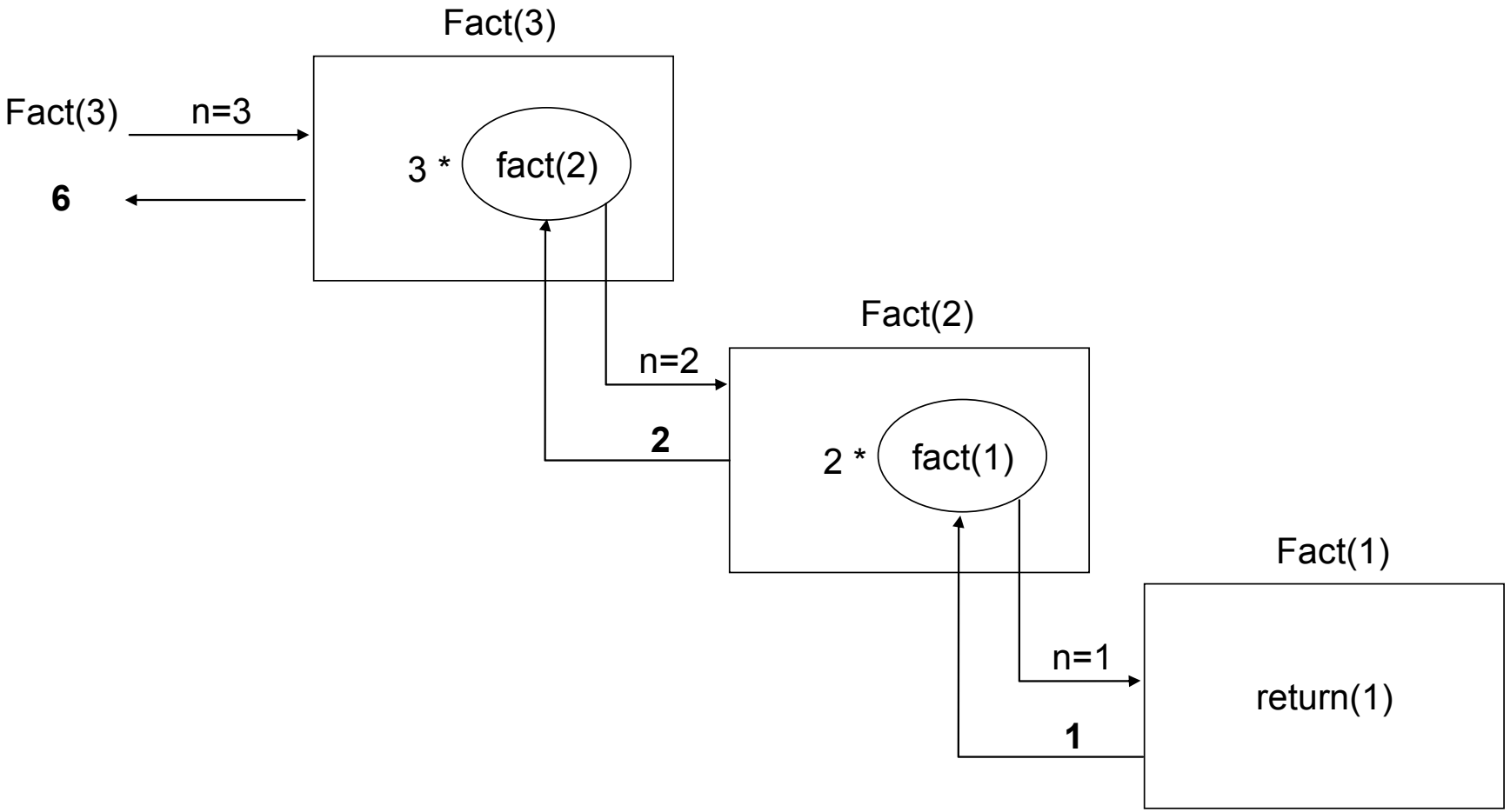
- مزایا
  - کد نویسی کوتاه و راحت
- معایب
  - فراخوانی های مکرر و نیاز به پشته
  - معمولا از نظر فضا و زمان بهینه نیستند.
- بعضی از مسائل را هم می توان به صورت غیر بازگشتی و هم بازگشتی حل کرد.
- اما راه حل بعضی از مسائل به طور ذاتی بازگشتی است.
- برای اینکه بتوان از روش بازگشتی برای حل یک مساله استفاده نمود، مساله باید قابلیت خرد شدن به زیرمساله‌هایی از همان نوع مساله اصلی و اندازه کوچکتر را داشته باشد.

فاکتوریل

$$f(n) = n! = \begin{cases} 1 & if \quad n = 1 \\ n \times (n-1)! & if \quad n > 1 \end{cases}$$

```
int fact (int n)
  {
    if (n == ١) then
  return(١);
    else
  return (n * fact(n-١));
  }
```

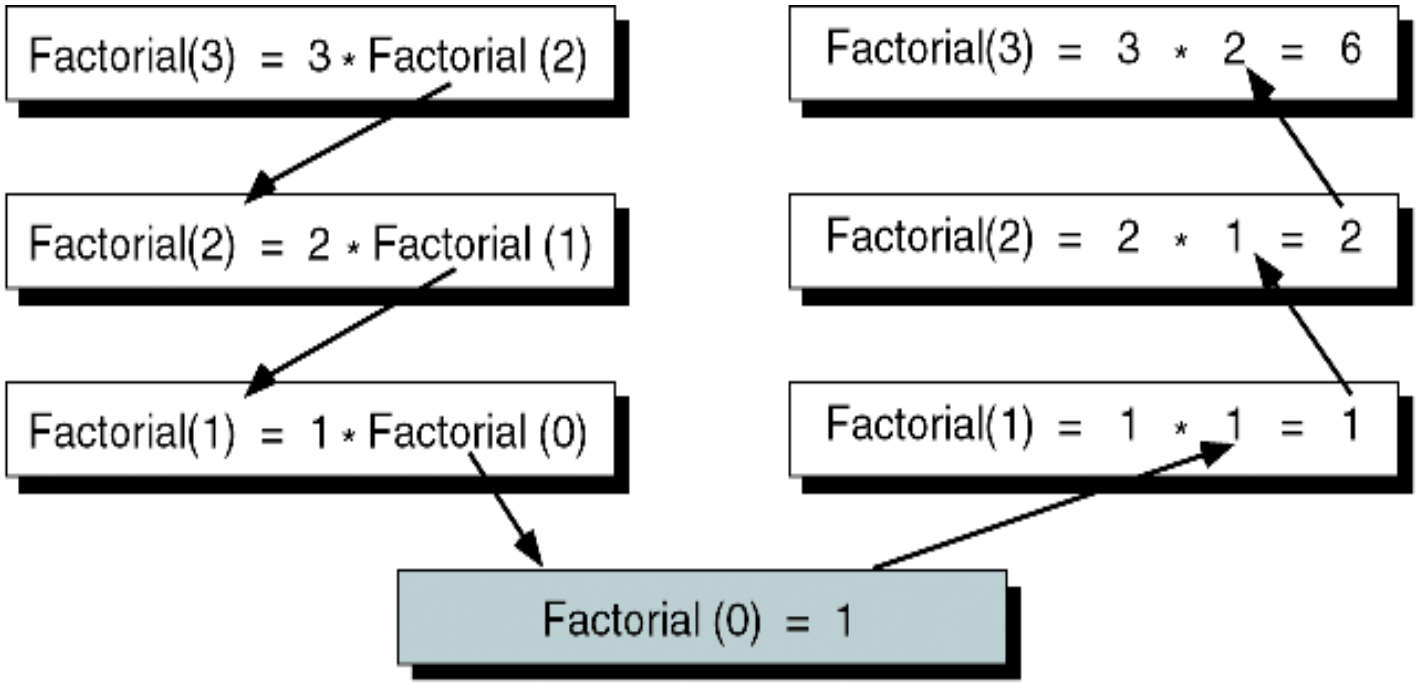
فاکتوریل
چگونگی پیمایش تابع fact



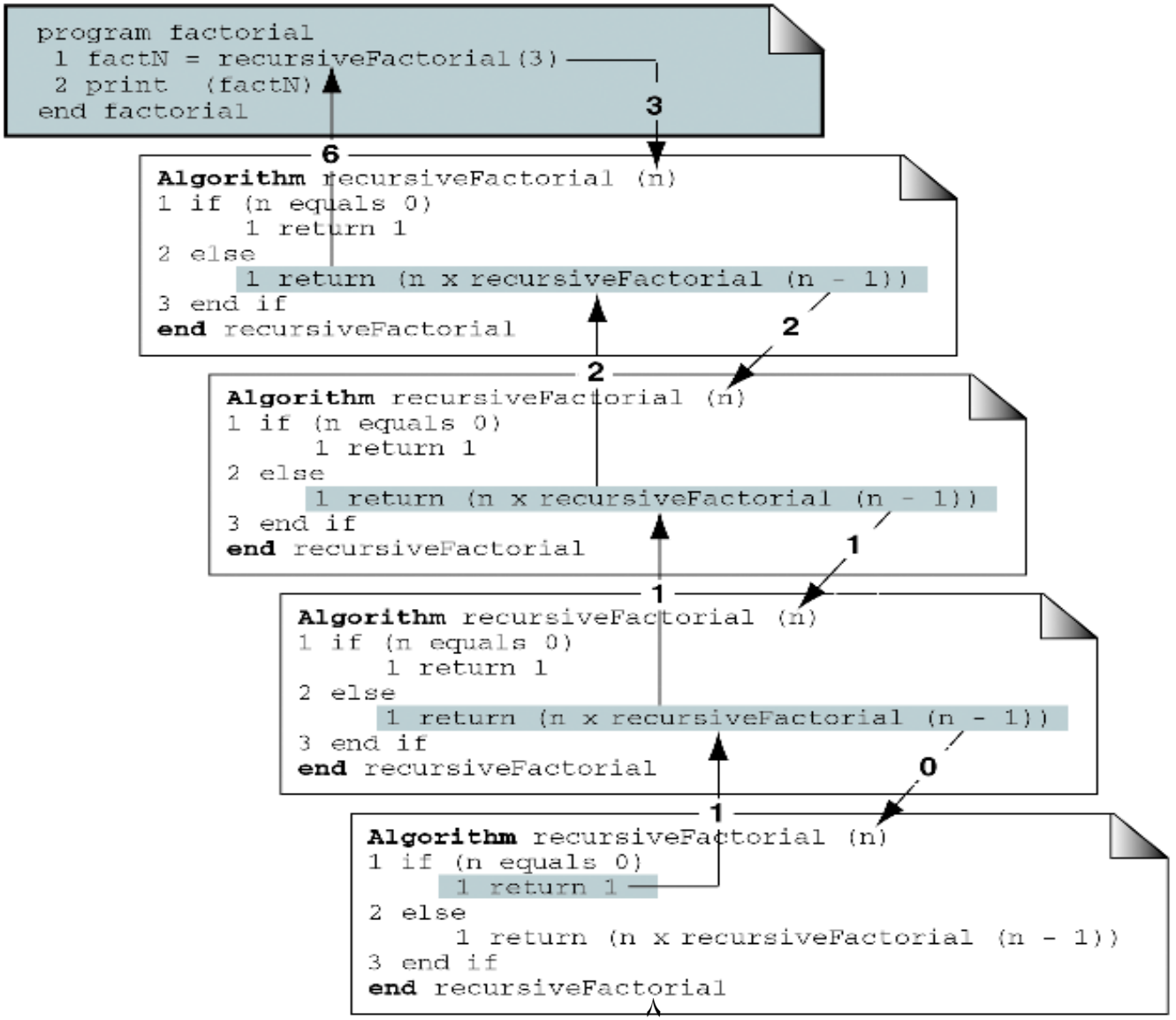
نحوه اجرای یک الگوریتم بازگشتی را درک کنید

❖ راه حل بازگشتی یک مساله شامل دو مرحله کلی است

- اول مساله را از بالا به پائین می شکند.
- سپس مساله را از پایین به بالا حل می کند.



فراخوانی الگوریتم های بازگشتی



طراحی الگوریتم های بازگشتی

- هر فراخوانی از الگوریتم بازگشتی هم قسمتی از مساله را حل می کند و هم اندازه مساله را کوچک می کند.
- قسمت اصلی راه حل، فراخوانی های بازگشتی است. در هر فراخوانی بازگشتی اندازه مساله کوچکتر می شود.
- عبارتی که مساله را حل می کند، حالت پایه(base case) نامیده می شود.
- هر الگوریتم بازگشتی باید یک حالت پایه داشته باشد.
- بقیه الگوریتم حالت عمومی نامیده می شود. این قسمت شامل منطق مورد نیاز برای کاهش اندازه مساله می باشد.
- طراحی الگوریتم بازگشتی
  - مشخص کردن حالت پایه
  - مشخص کردن حالت عمومی
  - ترکیب این دو در یک الگوریتم

تحلیل زمانی الگریتم بازگشتی

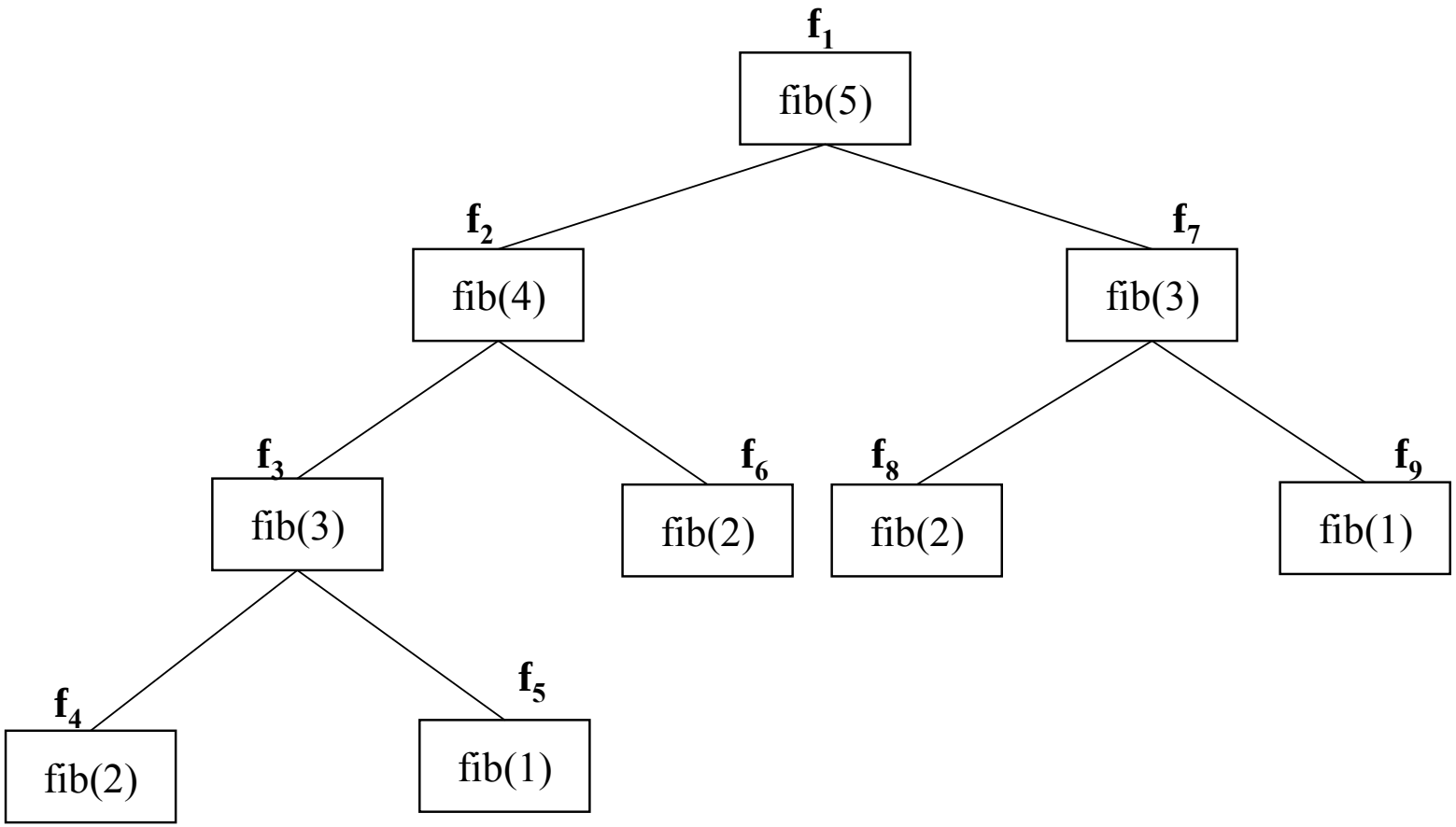
- محاسبه زمان اجرای الگوریتم بازگشتی
  - زمان حل زیر مساله
  - زمان شکستن مساله به زیر مسائل
  - زمان لازم برای ادغام جواب های زیر مسائل

الگوریتم باز گشتی
محاسبه جمله nام سری فیبوناچی

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

```
int fib (int n)
  {
    if (n == ١ or n == ٢) then  return(١);
    else
  return (fib(n - ١) + fib(n - ٢));
  }
```

چگونگی فراخوانی بازگشتی تابع fib



کارایی الگوریتم بازگشتی فیبوناچی

Fib(۵)
= Fib(۴) + Fib(۳)
= Fib(۳) + Fib(۲) + Fib(۳)
= Fib(۲) + Fib(۱) + Fib(۲) + Fib(۳)
= Fib(۱) + Fib(۰) + Fib(۱) + Fib(۲) + Fib(۳)
= Fib(۱) + Fib(۰) + Fib(۱) + Fib(۱) + Fib(۰) + Fib(۳)
= Fib(۱) + Fib(۰) + Fib(۱) + Fib(۱) + Fib(۰) + Fib(۲) + Fib(۱)
= Fib(۱) + Fib(۰) + Fib(۱) + Fib(۱) + Fib(۰) + Fib(۱) + Fib(۰) + Fib(۱)

- مشکل: محاسبه تکراری جملات
- راه حل
  - برنامه نویسی پویا (Dynamic programming) و استفاده از آرایه برای ذخیره جملات قبلی

الگوریتم بازگشتی پیدا کردن یک عنصر از لیست (جستجوی ترتیبی)

```
Alg rsearch(i)
  {
      case
  i > n : return (-١) ;  // x not found
  a[i] = x : return (i) ;  // x is a[i]
  else return (rsearch(i + ١)) ;  // recursive
       end case ;
   }
```

مساله برج هانوی (Hanoi tower)

- تعریف مساله

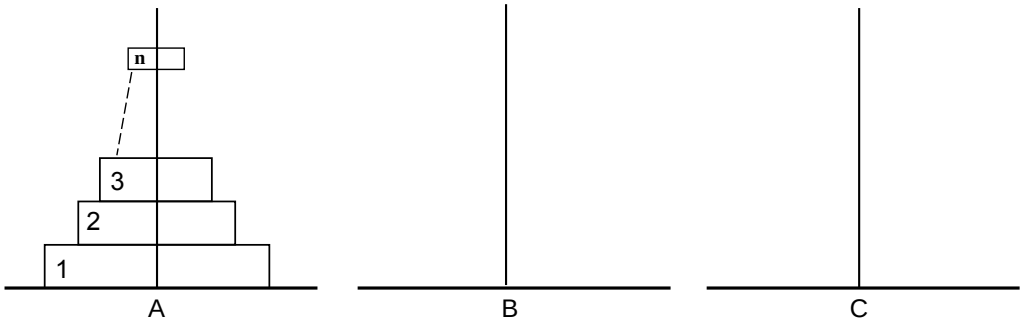
- سه میله - میله مبدا (A)، میله کمکی (B) و میله مقصد (C) و تعدادی دیسک در میله مبدا داریم. هدف انتقال تمام دیسک‌ها از میله مبدا به میله مقصد با رعایت دو شرط زیر است:
  - در هر زمان فقط یک دیسک را می‌توان جابجا نمود.
  - نباید در هیچ زمانی دیسکی بر روی دیسک با اندازه کوچکتر قرار بگیرد.

- اما هدف ما ارائه الگوریتمی برای انتقال دیسک‌ها با کمترین جابجایی ممکن است.

حل بازگشتی مساله برج هانوی

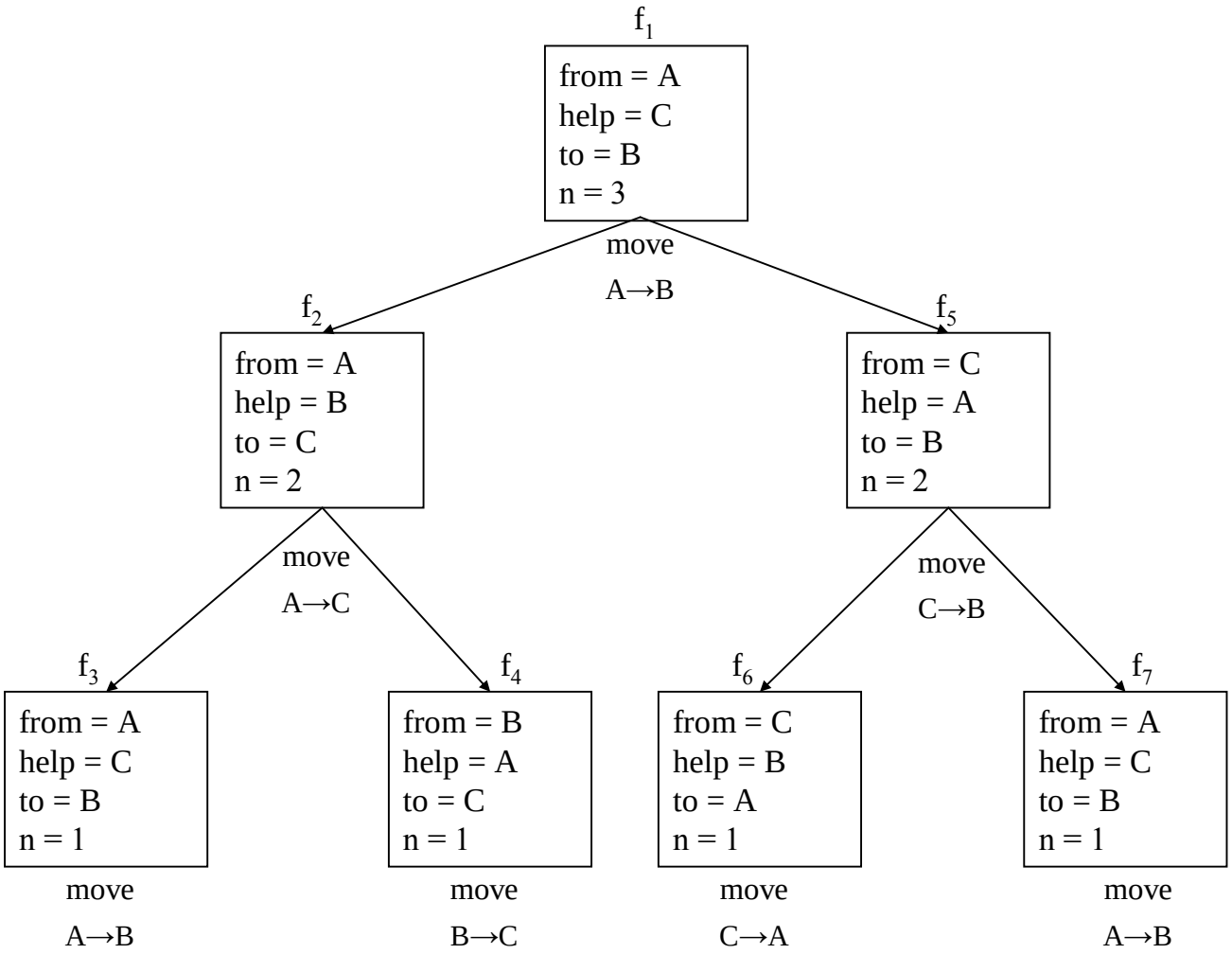
- برای جابجا کردن بزرگترین دیسک از میله A به میله C، ابتدا باید تمامی دیسک‌های کوچکتر به میله B منتقل شوند. پس از تمام شدن این مرحله، دیسک بزرگ را از میله A به میله Cمنتقل کرده و مجددا به کمک میله A تمامی دیسک‌های میله Bرا به میله Cمنتقل می‌کنیم. پس به طور خلاصه می‌توان گفت:
  - مرحله اول: n - ۱ دیسک بالایی میله مبدا با شرایط ذکر شده و به کمک میله C به میله Bمنتقل می‌شوند.
  - مرحله دوم: بزرگترین دیسک از میله مبدا به میله مقصد منتقل می‌شود.
  - مرحله سه: n - ۱ دیسک میله B با کمک گرفتن از میله A به میله مقصد منتقل می‌شوند.
- توانستیم عملیات جابجا کردن n دیسک را به دو عملیات مشابه ولی با اندازه کمتر و یک عملیات ساده تقسیم کنیم.

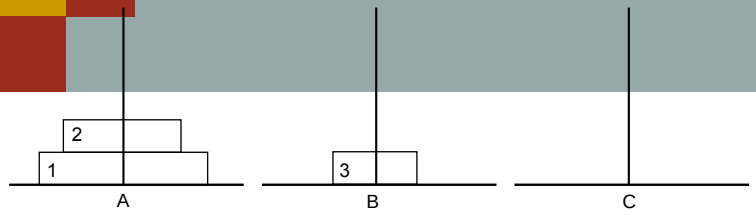
مسئله برج هانوی

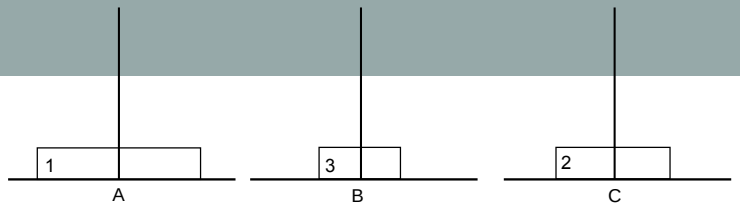


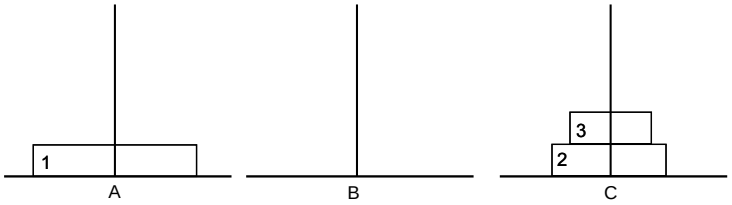

```
Alg Hanoy (n , from , help , to)
  {
     if (n == ۱) then
           move top disk from (from → to)  ;
     else
        {
           Hanoy(n – ۱ ,  from  ,  to  ,  help) ;
           move top disk from  (from → to)  ;
           Hanoy (n – ۱  ,  help  ,  from  ,  to) ;
        }
  }
```

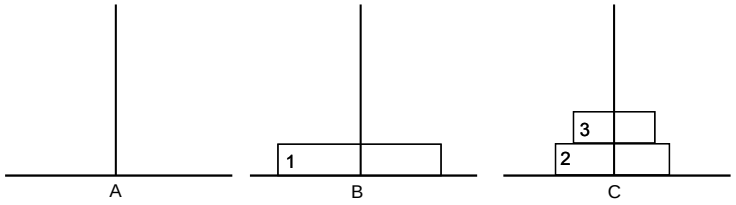
چگونگی فراخوانی تابع هانوی

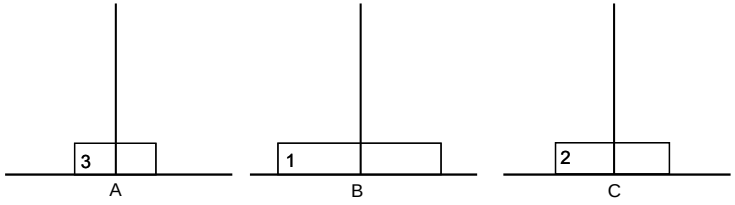


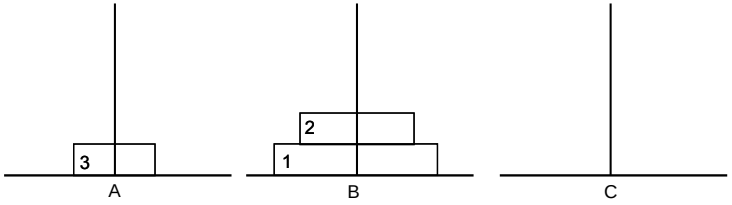

2
1
A
3
B
C

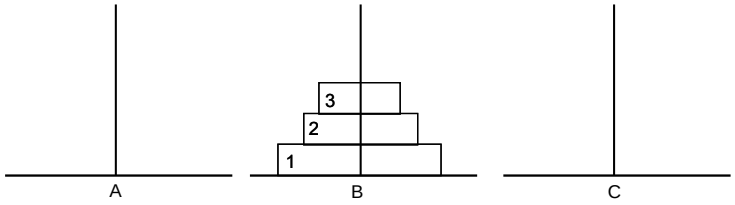

1
A
3
B
2
C


1
A
B
3
2
C


A
1
B
3
2
C


3
A
1
B
2
C


3
A
2
1
B
C


A
3
2
1
B
C

تحیل پیچیدگی زمانی مساله برج هانوی

- در حالت کلی می‌خواهیم بدانیم اگر تعداد دیسک‌ها nباشد، کمترین تعداد حرکت برای جابجا نمودن دیسک‌ها چقدر است؟
- فرض کنید (T(n تعداد حرکت‌های لازم جهت انتقال nدیسک به مقصد باشد. بر اساس توضیحات فوق، تعداد (T(n - ۱ حرکت برای انتقال n - ۱دیسک به میله کمکی، یک حرکت برای انتقال بزرگترین دیسک به میله مقصد، و مجددا (T(n - ۱ حرکت برای انتقال n - ۱دیسک موجود در میله کمکی به میله مقصد نیاز است. پس:

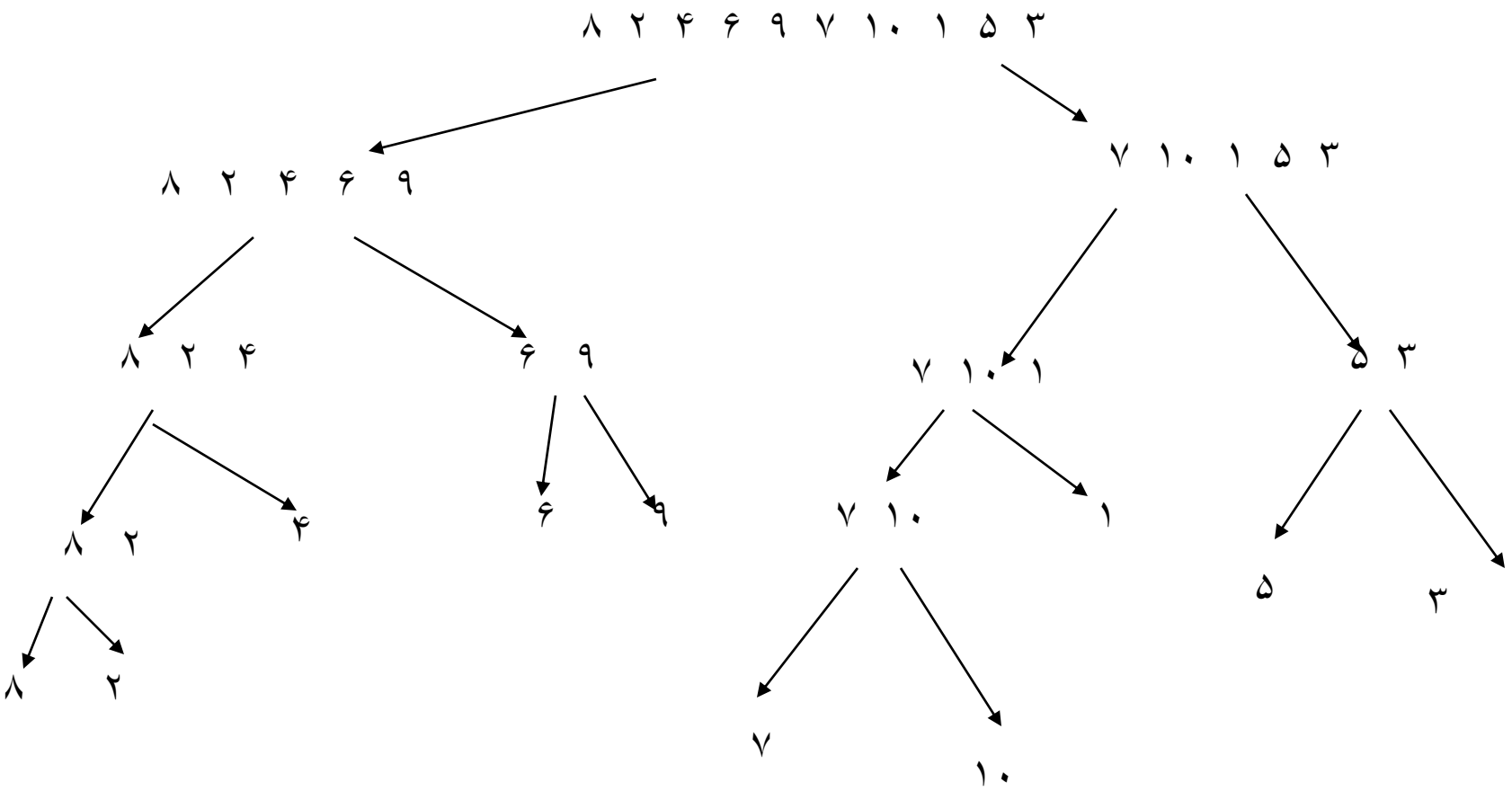
$$T(n) = ۲\,T(n-۱) + ۱$$

- با حل این رابطه بازگشتی به روش‌های ریاضی، به نتیجه زیر می‌رسیم:

$$T(n) = ۲^{n} - ۱$$

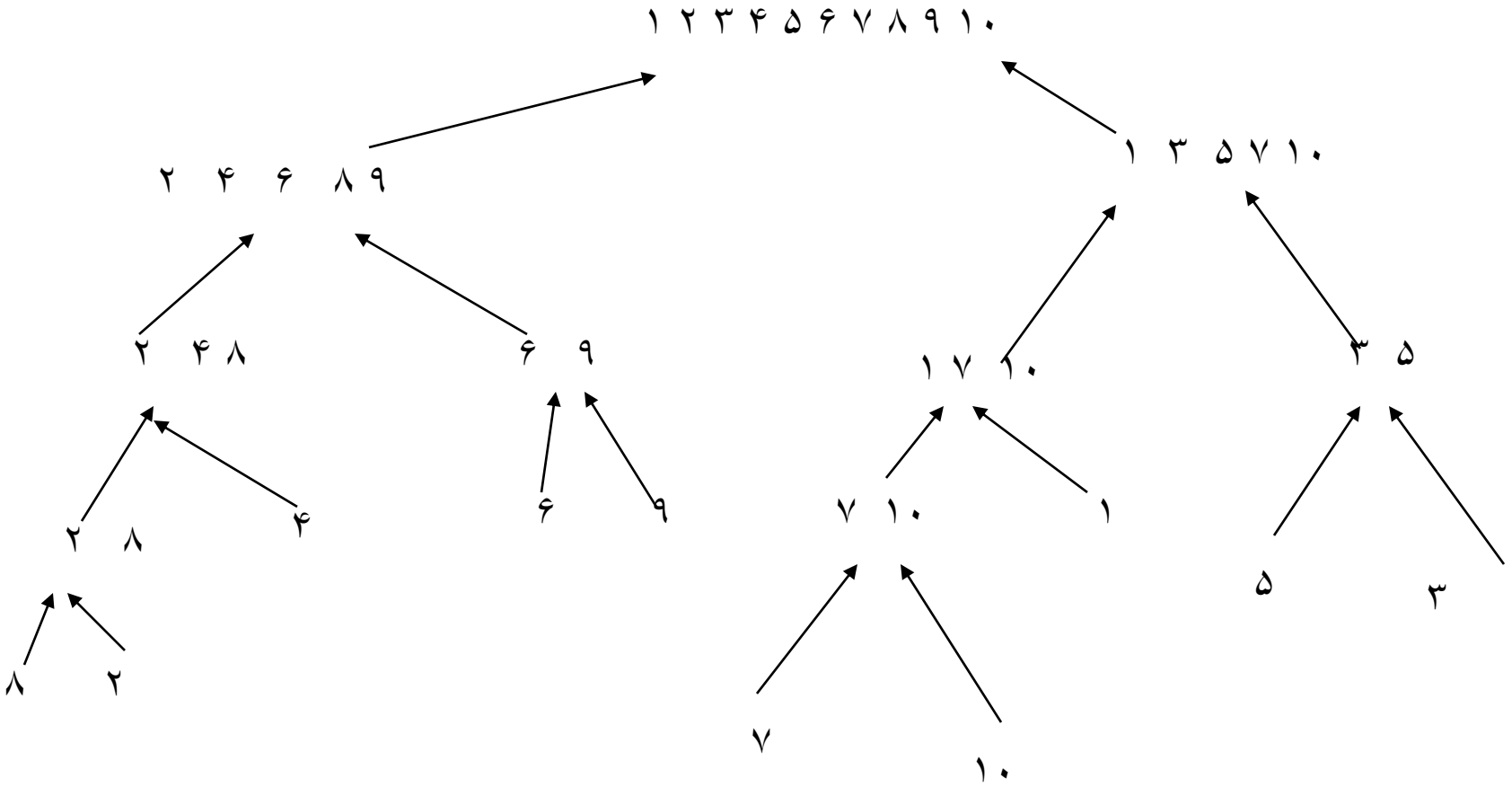
- مرتبه اجرایی این الگوریتم (O(۲n است که چندان مرتبه مطلوبی به نظر نمی‌رسد. اما همانگونه که بحث شد، این روش حداقل تعداد حرکت‌های ممکن را می‌دهد؛ و هرگز نمی‌توان روش دیگری با مرتبه پایین‌تر برای حل آن یافت.

مرتب سازی ادغام



The time complexity if O (n log n)

مرتب سازی ادغام



روش‌های حل معادلات بازگشتی

- استفاده از استقرای ریاضی(induction)
- روش جایگذاری و تکرار
- استفاده از قضیه اصلی
- درخت بازگشتی
- استفاده از درخت بازگشتی
- حل معادلات بازگشتی خطی با ضرایب ثابت
  - همگن(استفاده از معادله مشخصه)
  - غیر همگن
- حل معادلات بازگشتی با استفاده از سری مولد
- حل معادلات بازگشتی به روش تغییر متغیر

حل معادلات بازگشتی با استفاده از استقرای ریاضی

- ایجاد یک حدس اولیه برای تابع بازگشتی
- **هدف:** اثبات درستی حدس با استفاده از استقراء

- استقراء سه مرحله دارد:
  - **مبنای** استقراء: حدس برای شرط اولیه برقرار است.
  - **فرض** استقراء: فرض می شود که حدس به ازای n صحیح است.
  - **گام** استقراء: باید اثبات شود که حدس برای جمله $n+1$ ام نیز برقرار می باشد.
- استقرای قوی؟

- بازگشتی زیر را به روش استقرای ریاضی حل کنید.

$$\begin{cases} t(n) = t(\frac{n}{2}) + 1 \\ t(1) = 1 \end{cases}$$

- **حل** : ابتدا باید یک حل کاندیدا برای معادله فوق پیدا کنیم و سپس درستی این حل کاندیدا را به روش استقرا اثبات نماییم:

$$t(2) = t(\frac{2}{2}) + 1 = t(1) + 1 = 1 + 1 = 2$$

$$t(4) = t(\frac{4}{2}) + 1 = t(2) + 1 = 2 + 1 = 3$$

$$t(8) = t(\frac{8}{2}) + 1 = t(4) + 1 = 3 + 1 = 4$$

$$t(16) = t(\frac{16}{2}) + 1 = t(8) + 1 = 4 + 1 = 5$$

- با توجه به معادلات فوق می توان گفت که حل کاندیدا به صورت زیر است:

$$t(n) = \log(n) + 1$$

با استفاده از استقراء ثابت می کنیم که این حل درست است.

مبنای استقراء:

برای $n = ۱$ داریم:

$$t(1) = 1$$

فرض استقراء:

فرض کنید برای مقدار دلخواه $n > ۰$ که n توانی از ۲ است، داشته باشیم:

$$t(n) = \log(n) + 1$$

گام استقراء:

چون رابطه بازگشتی برای n های توانی از ۲ است، عدد بعدی که باید در نظر گرفته شود، n۲می‌باشد لذا باید نشان دهیم:

$$t(2n) = \log(2n) + 1$$

ابتدا n^2 را در رابطه بازگشتی مسأله قرار داده و با استفاده از فرض استقراء ، حکم را اثبات می‌کنیم:

$$t(2n) = t(\frac{2n}{2}) + 1 = t(n) + 1 = \log(n) + 1 + 1$$

$$t(2n) = \log(n) + \log(2) + 1 \qquad \log_x^x = 1$$

$$t(2n) = \log(2n) + 1 \longrightarrow t(2n) = \log(2n) + 1$$

چون حکم استقرا با گام استقرا اثبات شد بنابراین حل کاندیدا صحیح میباشد . یعنی داریم :

$$t(n) = \log(n) + 1$$

حل معادلات بازگشتی با استفاده از استقرای ریاضی
مثال ۲

$$T(n)=\begin{cases}1 & n=0\\ n*T(n-1) & n>0\end{cases}$$

✓چند جمله را محاسبه می کنیم:

T(۰) = ۱
T(۱) = ۱*T(۰) = ۱*۱ = ۱
T(۲) = ۲*T(۱) = ۲*۱ = ۲
T(۳) = ۳*T(۲) = ۳*۲ = ۶
T(۴) = ۴*T(۳) = ۴*۶ = ۲۴
T(k) = K*T(k-۱) = k!

حدس می زنیم که: T(n) = n!

حدس خود را با استقراء ثابت می کنیم:

مبنای استقراء:	برای n=۰: T(۰) = ۰! = ۱
فرض استقراء:	برای مقادیری از n داریم: T(n) = n!
گام استقراء:	باید نشان دهیم: T(n+۱) = (n+۱)!
	T(n) = n* T(n-۱) then T(n+۱) = (n+۱) * T(n) ➔ T(n+۱) = (n + ۱)* n! = (n+۱) !

بازگشتی زیر را با استفاده از استقرای ریاضی حل کنید.

$$\begin{cases} t(n) = 7t(\frac{n}{2}) \\ t(1) = 1 \end{cases}$$

جواب: چند مقدار اولیه عبارتند از:

$$t(2) = 7t(\frac{2}{2}) = 7t(1) = 7$$

$$t(4) = 7t(\frac{4}{2}) = 7t(2) = 7^2$$

$$t(8) = 7t(\frac{8}{2}) = 7t(4) = 7^3$$

$$t(16) = 7t(\frac{16}{2}) = 7t(8) = 7^4$$

نتیجه می شود که:

$$t(n) = 7^{\log n}$$

با استفاده از استقراء ثابت می‌کنیم که حل درست است.

مبنای استقراء: برای n = ۱ داریم: $t(1) = 1 = 7^0 = 7^{\log 1}$

فرض استقراء: فرض کنید برای هر مقدار دلخواه n > ۰ که n توانی از ۲ است، داریم: $t(n) = 7^{\log n}$

گام استقراء: باید نشان دهیم: $t(2n) = 7^{\log(2n)}$

اگر $2n$ را در بازگشتی قرار دهیم، بدست می‌آوریم:

$$t(2n) = 7t(\frac{2n}{2}) = 7t(n) = 7 \times 7^{\log n} = 7^{1+\log n} = 7^{\log 2+\log n} = 7^{\log(2n)}$$

به این ترتیب، استقراء ثابت می‌شود. سرانجام چون: $7^{\log n} = n^{\log 7}$

معمولا حل این بازگشتی را به صورت زیر ارائه می‌کنیم:

$$t(n) = n^{\log 7} \approx n^{2.81}$$

حل معادلات بازگشتی به روش جایگذاری و تکرار

$$\begin{cases} t(n) = t(n-1) + n & \text{for } n > 1 \\ t(1) = 1 \end{cases}$$

- با توجه به اینکه روش جایگذاری عکس روش استقراء است، داریم:

$$t(n) = t(n-1) + n$$
$$t(n-1) = t(n-2) + n - 1$$
$$t(n-2) = t(n-3) + n - 2$$

کلیه مقادیر را در معادله اصلی جایگزین می کنیم.

$$t(2) = t(1) + 2$$
$$t(1) = 1$$

$$t(n) = t(n-1) + n$$
$$= t(n-2) + n - 1 + n$$
$$= t(n-3) + n - 2 + n - 1 + n$$
$$= t(1) + 2 + \ldots n - 2 + n - 1 + n$$
$$= 1 + 2 + \ldots n - 2 + n - 1 + n$$
$$= \sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

$$\Rightarrow T(n) = \frac{n(n+1)}{2} \in O(n^2)$$

حل معادلات بازگشتی به روش جایگذاری و تکرار

$$T(n)=\begin{cases}2 & n=1\\ T(n-1)+5 & n>1\end{cases}$$

T(n) = T(n-۱) + ۵

T(n-۱) = T(n-۲) + ۵

T(n-۲) = T(n-۳) +۵

…

T(n-k) = T(n-k-۱)+۵

…

T(۱) = ۲

T(n) = T(n-۱) + ۵

= T(n-۲) + ۵ + ۵

= T(n-۳) + ۵ + ۵ + ۵

=T(n-۴) + ۵ + ۵ + ۵ + ۵

…

=T(n-k) + ۵k

باید به $T(1)$ برسیم بنابراین:

$$T(n-k)=T(1)\Rightarrow$$

$$n-k=1\Rightarrow k=n-1$$

با جایگذاری مقدار k داریم:

$$T(n)=T(n-k)+5k$$

$$T(n)=T(1)+5(n-1)=2+5n-5=5n-3$$

تمرین در کلاس

- پیچیدگی زمانی الگوریتم برج هانوی را به روش جایگذاری و تکرار محاسبه کنید:

الگوریتم بازگشتی پیدا کردن ماکزیمم عضو یک لیست، به صورت زیر می‌باشد:

```
Alg max (i , n)   {
    if (n == 1) then
      return (A[i]) ;
    else
      return (maximum (max (i , ⌊n/2⌋) , max (i + ⌊n/2⌋ , n - ⌊n/2⌋ )))
  }
```

i : اندیس خانه اول لیست

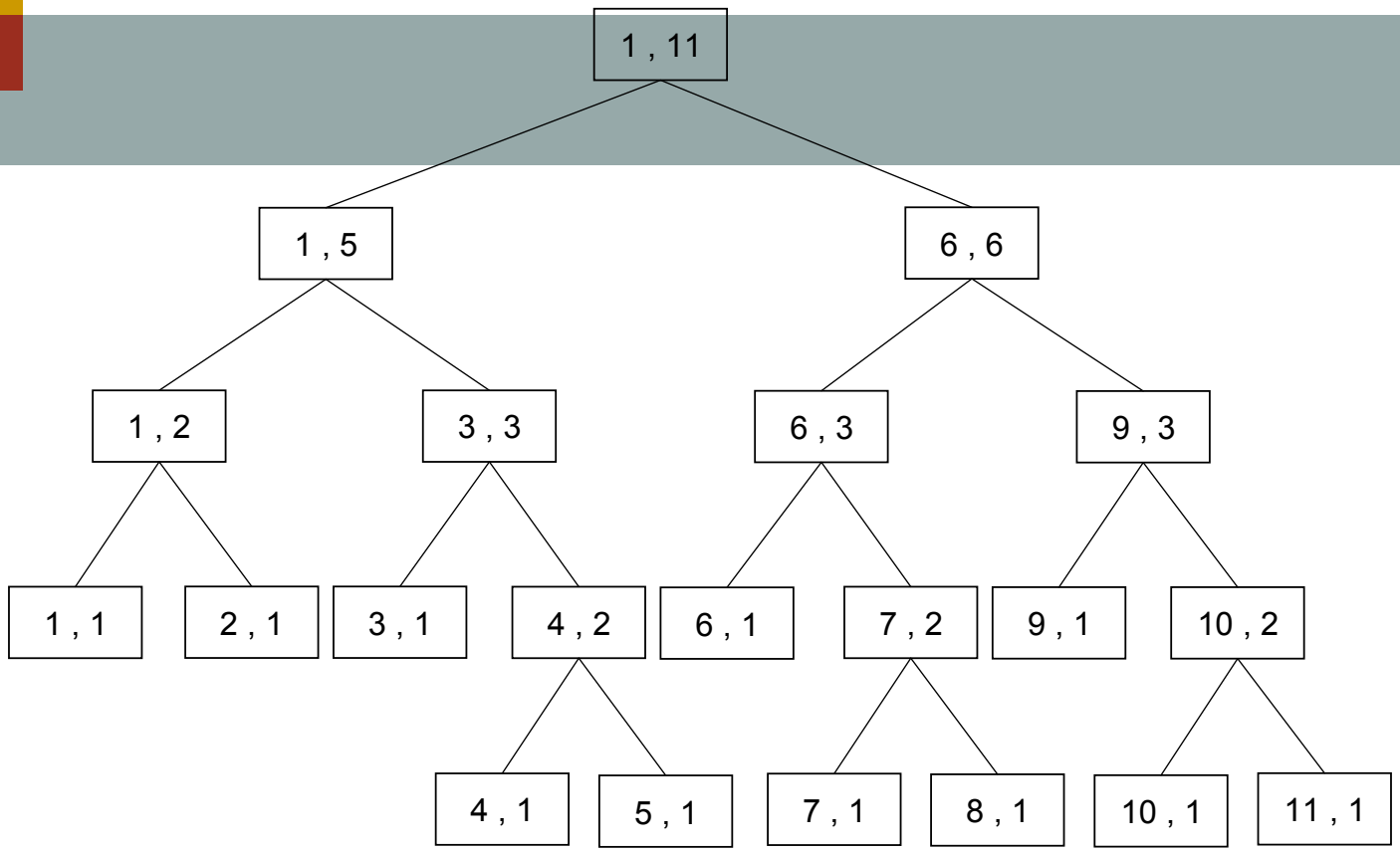
n : تعداد عناصر لیست

الف) درخت فراخوانی الگوریتم فوق را برای لیست زیر رسم کنید.

7	16	10	9	5	4	3	1	12	19	6

ب) معادله بازگشتی الگوریتم پیدا کردن ماکزیمم عضو یک لیست را، نوشته و آن را بصورت جایگزینی حل کنید.

n = ۱۱



تعداد فراخوانی برابر با ۲۱ می‌باشد.

تعداد اجرای تابع maximumبرابر با ۱۰ می‌باشد.

تعداد فراخوانی‌های با n = ۱برابر با ۱۱ می‌باشد (تعداد برگها و تعداد عناصر لیست).

ب) معادله بازگشتی الگوریتم بصورت زیر خواهد بود:

$$t(n)=\begin{cases} a & n=1 \\ 2t(\frac{n}{2})+b & n>1 \end{cases}$$

حل معادله بازگشتی فوق به روش جایگزینی: (با فرض $n=2^k$)

$$t(n)=2t(\frac{n}{2})+b$$

$$=2(2t(\frac{n}{4})+b)+b=4t(\frac{n}{4})+2b$$

$$=4(2t(\frac{n}{8})+b)+2b+b=8t(\frac{n}{8})+4b+2b+b$$

.

.

.

$$=2^k t(\frac{n}{2^k})+(2^{k-1}+2^{k-2}+\ \dots +2^2+2^1+2^0)b$$

$$=na+nb-b \quad \rightarrow \quad T(n)=n(a+b)-b$$

$$T(n)\in O(n)$$

$$\sum_{i=0}^{n} 2^i = 2^{n+1}-1$$

حل روابط بازگشتی با استفاده از قضیه اصلی

$$T(n) = aT(\frac{n}{b}) + Cn^k$$

اگر a>۱ و b>=۱

$$T(n) = \begin{cases} \theta\left(n^{\log_b^a}\right) & a > b^k \\ \theta\left(n^k \log n\right) & a = b^k \\ \theta\left(n^k\right) & a < b^k \end{cases}$$

مثال

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

$$T(n) = 4T\left(\frac{n}{2}\right) + n^2 \quad \Rightarrow \quad T(n) \in \Theta\left(n^2 \lg n\right)$$

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

قضیه اصلی – حالت کلی

- if T(n) = aT(n/b) + f(n) then

$$T(n)=\begin{cases}\Theta\left(n^{\log_b a}\right) & f(n)=O\left(n^{\log_b a-\varepsilon}\right) \\ \Theta\left(n^{\log_b a}\log n\right) & f(n)=\Theta\left(n^{\log_b a}\right) \\ \Theta\left(f(n)\right) & f(n)=\Omega\left(n^{\log_b a+\varepsilon}\right)\text{AND} \\ & af(n/b)<cf(n)\ \text{ for large } n\end{cases}\quad \begin{matrix}\varepsilon>0\\ c<1\end{matrix}$$

❖ بازگشتی زیر را با استفاده از روش قضیه اصلی حل کنید.

$$T(n)=\begin{cases}1 & n=1\\ 2T(\frac{n}{2})+n & \text{else}\end{cases}$$

جواب: با توجه به معادله بازگشتی فوق، موارد زیر را داریم:

$$a=2 \quad , \quad b=2 \quad , \quad f(n)=n \quad , \quad T(1)=1$$

$$T(n) \in O(n \log n)$$

$$T(n) = 4T(\frac{n}{2}) + \log n$$

$$n^{\log_b^a} = n^{\log_2^4} = n^2 \quad , \quad O(f(n)) = \log n \Rightarrow n^2 > \log n$$

$$T(n) = O(n^2)$$

$$T(n) = 3T(\frac{n}{4}) + n \log n$$

$$n^{\log_b^a} = n^{\log_4^3} = n^{0.79} \quad , \quad O(f(n)) = n \log n \Rightarrow T(n) = O(n \log n)$$

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

$$a = b = 2 \Rightarrow T(n) = \log n \times O(f(n)) = \log n . n \log n \Rightarrow T(n) = n \log^2 n$$

نکته : در رابطه بازگشتی $T(n)=T(\frac{a}{b}n)+T(\frac{c}{d}n)+...+f(n)$ اگر $\frac{a}{b}+\frac{c}{d}+...=1$ باشد آنگاه داریم: $T(n)=O(\log n \times O(f(n))$.

مثال ۳۷ : پیچیدگی رابطه بازگشتی $T(n)=T(\frac{n}{3})+T(\frac{2}{3}n)+n^2$ را به دست آورید.

حل :

$$\frac{1}{2}+\frac{2}{3}=1 \Rightarrow T(n)=O(\log n \times (n)^2)=O(n^2 \log n)$$

حل معادلات بازگشتی به کمک درخت بازگشت

- مراحل
  - ساخت درخت بازگشت
  - تعیین ارتفاع درخت بازگشت بر حسب اندازه ورودی
  - تعیین هزینه هر سطح از درخت بازگشت
  - محاسبه مجموع هزینه های تمامی سطوح

رابطه های بازگشتی خطی

- بازگشتی خطی همگن درجه k:

$$a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = 0$$

  - رابطه به صورت زیر یک رابطه بازگشتی مرتبه دوم همگن نامیده می شود.

$$T(n) = aT(n-1) + bT(n-2) \qquad n > 1$$
$$T(0) = C_0 \quad , T(1) = C_1$$

  - در این رابطه a,b ثابت هستند

- بازگشتی خطی ناهمگن

$$a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = f(k)$$

حل بازگشتی های خطی همگن

❖ **تعریف:** بازگشتی زیر را درنظر بگیرید که در آن k و a_iها ثابتند:

$$a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = 0$$

این بازگشتی، معادله خطی همگن با ضرایب ثابت از مرتبه kاُم نام دارد.

❖ یک بازگشتی خطی همگن از مرتبه kاُم، نیازمند k شرط اولیه می باشد.

❖ دنباله فیبوناچی

$$\begin{cases} t_n = t_{n-1} + t_{n-2} \\ t_0 = 0 \\ t_1 = 1 \end{cases}$$

حل معادلات بازگشتی همگن
با استفاده از معادله مشخصه

$$a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = 0$$

جواب معادله به فرم x^n می‌باشد. پس با قرار دادن $t(n) = x^n$ داریم:

$$a_0 x^n + a_1 x^{n-1} + \ \ldots \ + a_k x^{n-k} = 0$$

حال طرفین را بر x^{n-k} (کوچکترین توان x)تقسیم می کنیم و خواهیم داشت:

$$a_0 x^k + a_1 x^{k-1} + \ \ldots \ + a_k x^0 = 0$$

به این معادله، معادله مشخصه رابطه بازگشتی گویند که k جواب $x_1, x_2, \ \ldots \ , x_k$ دارد و در حالت کلی حل نهایی رابطه بازگشتی با توجه به این جوابها به صورت زیر می‌باشد:

$$t(n) = c_1 x_1^n + c_2 x_2^n + \ \ldots \ + c_k x_k^n$$

که در آن c_i ها ثابت هستند و می توان آنها را از طریق مقدیر اولیه بدست آورد.

مثال (دنباله فیبوناچی)

$$\begin{cases} t_n = t_{n-۱} + t_{n-۲} \\ t_۰ = ۰ \\ t_۱ = ۱ \end{cases}$$

معادله مشخصه: $r^۲ - r - ۱ = ۰ \Rightarrow r_۱ = (۱+\sqrt{۵})/۲ \quad , \quad r_۲ = (۱-\sqrt{۵})/۲$

$$t_n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n$$

معادله بازگشتی خطی همگن زیر را حل کنید.

جواب:

$$\begin{cases} t(n)-3t(n-1)-4t(n-2)=0 & \text{for } n>1 \\ t(0)=0 \\ t(1)=1 \end{cases}$$

۱- معادله مشخصه رابطه بازگشتی را بدست می‌آوریم:

$$t(n)-3t(n-1)-4t(n-2)=0$$

با قرار دادن $T(n)=x^n$ داریم: $x^n-3x^{n-1}-4x^{n-2}=0$ طرفین را بر x^{n-2} تقسیم می کنیم:

۲- معادله مشخصه را حل می‌کنیم: $x^2-3x-4=(x-4)(x+1)=0$ ریشه‌های آن عبارتند از:

$$\begin{cases} x-4=0 \rightarrow x=4 \\ x+1=0 \rightarrow x=-1 \end{cases}$$

۳- پس از اعمال قضیه معادله زیر به عنوان حل رابطه بازگشتی بدست می‌آید:

$$t(n)=c_1 4^n+c_2(-1)^n$$

۴- مقادیر ثوابت را از طریق اعمال مقادیر اولیه در رابطه فوق بدست می‌آوریم:

$$\begin{cases} t(0)=0=c_1 4^0+c_2(-1)^0 \\ t(1)=1=c_1 4^1+c_2(-1)^1 \end{cases} \rightarrow \begin{cases} c_1+c_2=0 \\ 4c_1-c_2=1 \end{cases} \rightarrow \begin{cases} c_1=\frac{1}{5} \\ c_2=\frac{-1}{5} \end{cases}$$

$$t(n)=\frac{1}{5}4^n-\frac{1}{5}(-1)^n$$

حل بازگشتی های خطی همگن

- **قضیه:** فرض کنید r یک ریشه m-گانه معادله مشخصه یک بازگشتی خطی همگن باشد، قسمتی از جواب بازگشتی که مربوط به این ریشه می باشد به صورت زیر در حل بازگشتی اعمال می شود:

$$(c_0+c_1n+c_2n^2+\ldots+c_{m-1}n^{m-1})r^n$$

در معادله بازگشتی صدق خواهد کرد و ضرایب c_k از شرایط اولیه محاسبه می شوند.

- در معادله همگن درجه ۲ در صورتی که ریشه مضاعف داشته باشیم

$$(c_0+c_1n)r^n$$

مرتبه زمانی الگوریتمی با تابع زمانی زیر برابر کدام گزینه است(ارشد سراسری ۸٤)

$$T(n)=\begin{cases}0 & n=0\\ 1 & n=1\\ 3T(n-1)+4T(n-2) & Otherwise\end{cases}$$

الف- n^2

ب- 4^n

ج- $2^n \log n$

د- $n^4 \log n$

حل بازگشتی های خطی ناهمگن

- **تعریف:** بازگشتی زیر را درنظر بگیرید که در آن k و a_iها ثابت و $f(n)$ تابعی غیر صفر می باشد:

$$a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = f(n)$$

این بازگشتی، معادله خطی ناهمگن با ضرایب ثابت از مرتبه kاُم نام دارد.

- حل بازگشتی های خطی ناهمگن را در حالتی خاص که $f(n)$ یک چندجمله ای باشد در ادامه شرح می دهیم.

حل بازگشتی های خطی ناهمگن

- **قضیه:** بازگشتی ناهمگن زیر را درنظر بگیرید:

$$a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = b^n p(n)$$

این شکل از بازگشتی، می تواند به بازگشتی همگنی تبدیل شود که معادله مشخصه آن به صورت زیر است:

$$(a_0 r^k + a_1 r^{k-1} + \ldots + a_k r^0)(r - b)^{d+1} = 0$$

که d برابر با درجه $p(n)$ است.

- اگر بیش از یک جمله مانند $b^n p(n)$ در سمت راست وجود داشته باشد، هر یک از آنها در معادله مشخصه ظاهر می شود.
- حل بازگشتی:

$$a_n = a_n^h + a_n^p$$

حل بازگشتی های خطی ناهمگن

$t_n - ۳t_{n-۱} = ۴^n(۲n+۱)$ for $n>۱$

$t_۰ = ۰$, $t_۱ = ۱۲$

معادله مشخصه: $(r-۳)(r-۴)^{۱+۱} = ۰$

حل بازگشتی: $t_n = c_۱۳^n + c_۲۴^n + c_۳n۴^n$

$$=> t_n = ۲۰(۳^n) - ۲۰(۴^n) + ۸n۴^n$$

حل روابط بازگشتی با روش تغییر متغیر

- با تغییر متغیر می توان رابطه بازگشتی را به یک رابطه ساده تر برای حل تبدیل کرد.

$$T(n) = T(\sqrt{n}) + 1$$

$n = ۲^m$

$T(۲^m) = T(۲^{m/۲}) + ۱$

$T(۲^m) = S(m)$

$S(m) = S(m/۲) + ۱$

با روش جایگذاری حل می شود:

$S(۱) = c \rightarrow S(m) = c + \lg m$

در نهایت داریم:

$m = \lg n \rightarrow T(n) = c + \lg \lg n$

تمرین

۱-۱۴-غیر از مسائل مطرح شده در کلاس پنج مساله ارائه دهید که به صورت بازگشتی قابل حل هستند. الگوریتم بازگشتی انها را بنویسید. رابطه بازگشتی الگوریتم را بنویسید و زمان اجرای آن را به دست اورید.

تمرین

۱-۱۵- پیچیدگی توابع زیر را به روشهای خواسته شده، بدست آورید:

- a , b: روش جایگذاری با تکرار،
- c و d : روش قضیه اصلی
- e و f: روش معادله مشخصه
- d: روش تغییر متغیر

$$a)\ T(n)=\begin{cases}1 & n=1\\ T(n-1)+n & n>1\end{cases}$$

$$b)T(n)=\begin{cases}3 & n=1\\ 4T(n-1) & n>1\end{cases}$$

c) $T(n)=۹T(n/۳)+n$

$$d)T(n)=8T(\frac{n}{9})+n\log n$$

e) $T(n)=۳T(n-۱)+۴T(n-۲)$

f) $a_n = ۶a_{n-۱} - ۹a_{n-۲}$

$a_۰ = ۰$

$a_۱ = ۱$

$$d)\ T(n)=2T(\sqrt{n})+\lg n \qquad T(1)=1$$