- 书面作业讲解
  - CS第5.5节问题8、11、14
  - TC第11.2节练习3、5、6
  - TC第11.3节练习3、4
  - TC第11.4节练习2、3
  - TC第11章问题1、2

CS第5.5节问题8

- hash n items into k locations
  - (a) probability that all n items hash to different locations
    - n>k: 0
    - n≤k: $\frac{A_k^n}{k^n}$
  - (b) probability that the i-th item is the first collision
    - $\frac{A_k^{i-1}}{k^{i-1}} \cdot \frac{i-1}{k}$
  - (c) expected number of items you hash until the first collision
    - $\sum_{i=2}^{\min(n,k+1)} i\left(\frac{A_k^{i-1}}{k^{i-1}} \cdot \frac{i-1}{k}\right)$

CS第5.5节问题14

- expected number of empty slots when you hash 2k items into a hash table with k slots
  - 定理5.15 $$k\left(1-\frac{1}{k}\right)^{2k}$$
- expected fraction of empty slots when k is reasonably large
  - $$\lim_{k\to+\infty}\frac{k\left(1-\frac{1}{k}\right)^{2k}}{k}=\left(\lim_{k\to+\infty}\left(1-\frac{1}{k}\right)^{k}\right)^{2}=\frac{1}{e^{2}}$$

TC第11.2节练习3

- modifying the chaining scheme to keep each list in sorted order
  - successful searches
  - unsuccessful searches
  - insertions
  - deletions

TC第11.2节练习5

- storing n keys drawn from |U|>nm into a hash table of size m
  - worst-case searching time: $\Theta(n)$

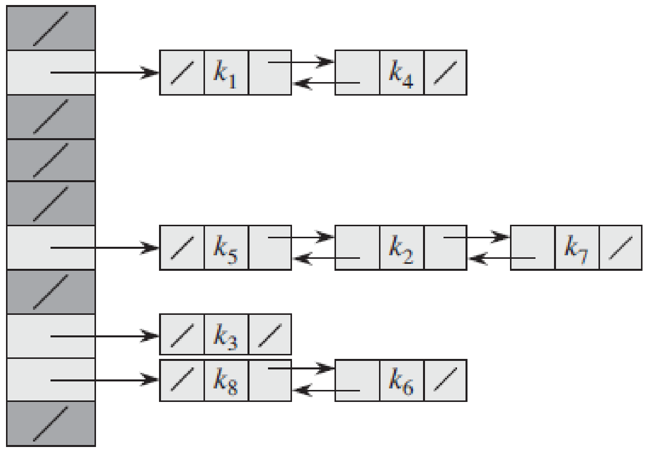
TC第11.2节练习6

- Selects a key uniformly at random from among n keys in the hash table of size m and returns it in expected time O(L(1+1/α)).
  - 方法1：随机等概率地选一个chain，再从中随机等概率地选一个key，这样可以吗？
  - 方法2：随机等概率地选一个序号，再找到对应的key
    - 先找chain，期望运行时间：

$$\sum_{i=1}^{m} \frac{L_i}{n} i = \cdots$$

    - 再在chain中找key，期望运行时间：

$$\sum_{i=1}^{m} \frac{L_i}{n} \frac{1+L_i}{2} = \cdots$$



TC第11.3节练习3

- character string interpreted in radix 2^p
  - $x_n \ldots x_1 x_0 \rightarrow \sum x_i (2^p)^i$
  - $\left(x_i\left(2^p\right)^i + x_j\left(2^p\right)^j\right) - \left(x_j\left(2^p\right)^i + x_i\left(2^p\right)^j\right)$
    $= \left(x_i - x_j\right)\left(\left(2^p\right)^i - \left(2^p\right)^j\right)$
    $= \left(x_i - x_j\right)\left(2^p - 1\right)\cdots$

TC第11章问题1

- (c) Show that Pr{X>2lgn}=O(1/n).
  - 这样对不对？

$$\Pr\{X > 2\lg n\} = \Pr\{\max_{1\le i\le n} X_i > 2\lg n\} = \sum_{1\le i\le n} \Pr\{X_i > 2\lg n\} = \sum_{1\le i\le n} O\left(\frac{1}{n^2}\right) = O\left(\frac{1}{n}\right)$$

- (d) Show that the expected length E[X] of the longest probe sequence is O(lgn).

$$\begin{aligned}
E[X] &= \sum_{i=1}^{n} i\Pr(X = i) = \sum_{i=1}^{2\lg n} i\Pr(X = i) + \sum_{i=2\lg n+1}^{n} i\Pr(X = i) \\
&\le 2\lg n \sum_{i=1}^{2\lg n} \Pr(X = i) + n \sum_{i=2\lg n+1}^{n} \Pr(X = i) \\
&= 2\lg n \Pr(X \le 2\lg n) + n\Pr(X > 2\lg n) \\
&\le 2\lg n + n O\left(\frac{1}{n}\right) \\
&= O(\lg n)
\end{aligned}$$

TC第11章问题2

- (b) Show that $P_k \le nQ_k$.
  - P_k=Pr(最多的一个恰为k)

    =Pr(存在一个恰为k且其余均≤k)

    ≤Pr(存在一个恰为k)

    ≤∑Pr(第i个恰为k)

    $=\sum Q_k$

    $=nQ_k$

- 教材答疑和讨论
  - TC第15章

问题1：dynamic programming

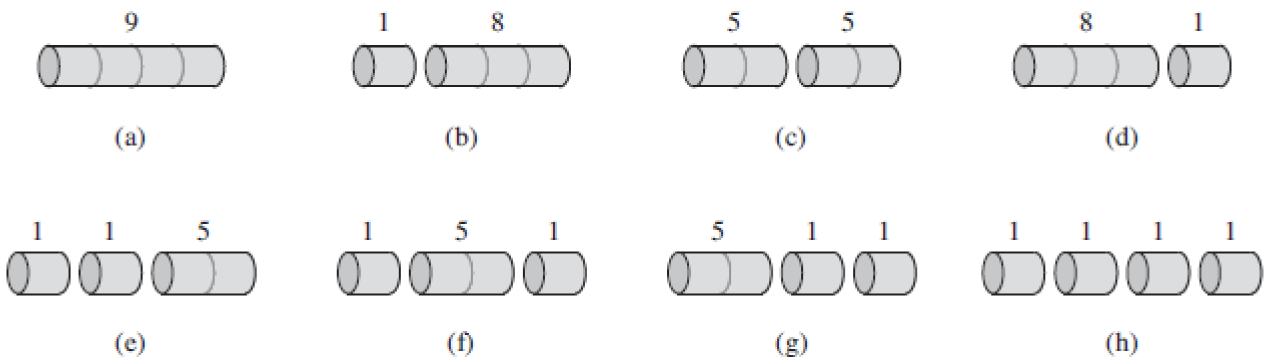
- dynamic programming常用来解决哪一类问题？
- 当问题具有什么特征时，可以使用dynamic programming？
- 当问题具有什么特征时，使用dynamic programming能够提高效率？为什么？
- 付出的代价是什么？

问题1：dynamic programming (续)

- dynamic programming的四个步骤分别是什么含义？
  1. Characterize the structure of an optimal solution.
  2. Recursively define the value of an optimal solution.
  3. Compute the value of an optimal solution.
  4. Construct an optimal solution from computed information.

问题2：rod cutting

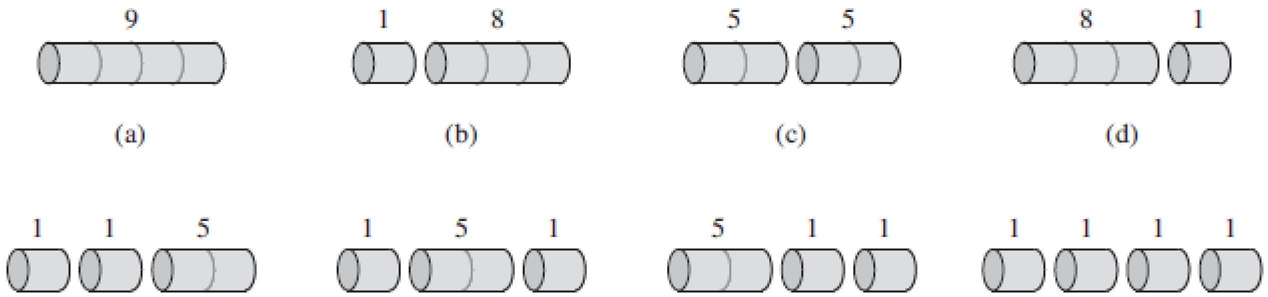
- rod cutting要解决什么问题？



length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

问题2： rod cutting (续)

- Characterize the structure of an optimal solution.
  - 最优子结构是什么？



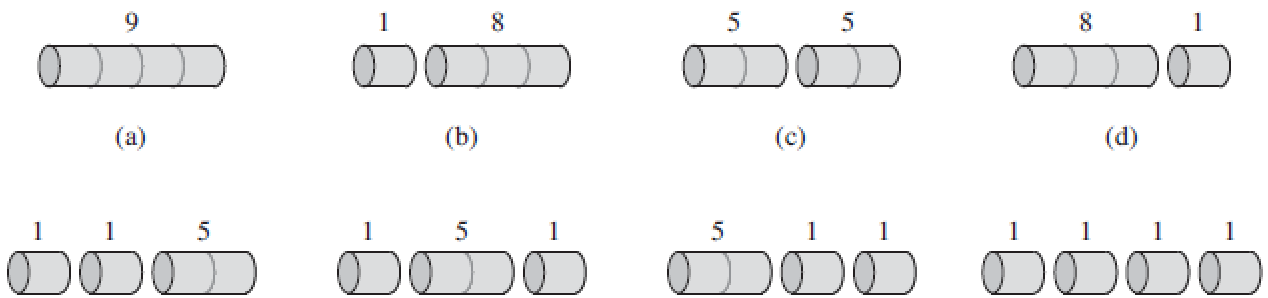
(a) (b) (c) (d)
(e) (f) (g) (h)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

问题2：rod cutting (续)

- Recursively define the value of an optimal solution.

$$r_n = \max_{1 \le i \le n} (p_i + r_{n-i})$$



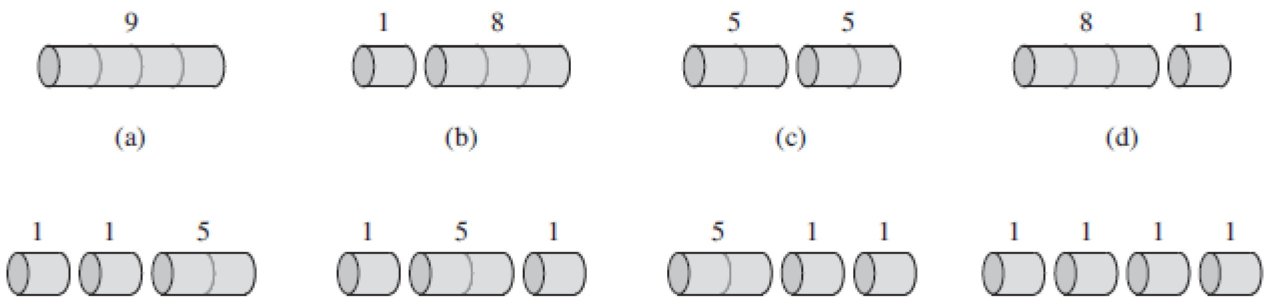
(a) (b) (c) (d)
(e) (f) (g) (h)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

问题2： rod cutting (续)

- Compute the value of an optimal solution.
  - 计算的顺序是什么？

```
for j = 1 to n
    q = -∞
    for i = 1 to j
        q = max(q, p[i] + r[j - i])
    r[j] = q
```



(a) (b) (c) (d)
(e) (f) (g) (h)

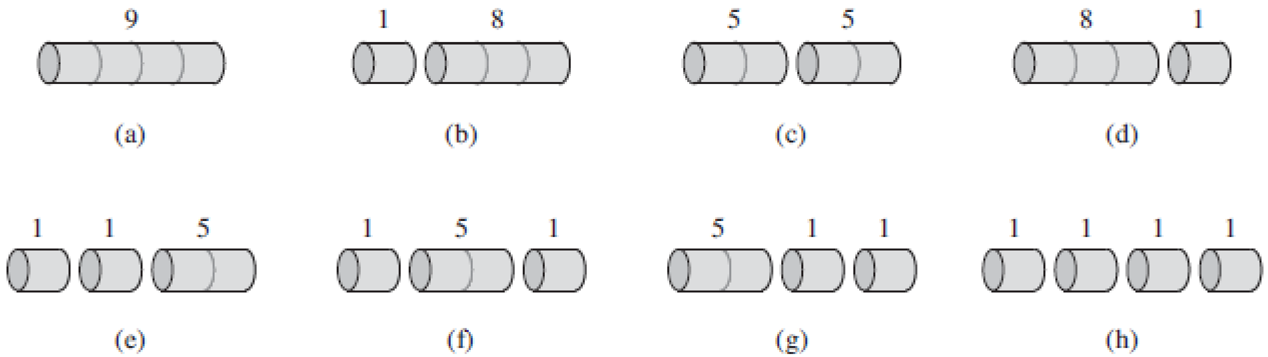
length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

问题2： rod cutting (续)

- Construct an optimal solution from computed information.
  - 怎样记住得到最优解的过程？
  - 如何基于此输出最优解？

```
if q < p[i] + r[j − i]
    q = p[i] + r[j − i]
    s[j] = i
```

```
while n > 0
    print s[n]
    n = n − s[n]
```



length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

问题3： matrix-chain multiplication

- matrix-chain multiplication要解决什么问题？

$$
\begin{aligned}
&(A_1(A_2(A_3A_4))) \;, \\
&(A_1((A_2A_3)A_4)) \;, \\
&((A_1A_2)(A_3A_4)) \;, \\
&((A_1(A_2A_3))A_4) \;, \\
&(((A_1A_2)A_3)A_4) \;.
\end{aligned}
$$

问题3： matrix-chain multiplication (续)

- Characterize the structure of an optimal solution.
  - 最优子结构是什么？

$$
\begin{array}{l}
(A_1(A_2(A_3A_4)))\ , \\
(A_1((A_2A_3)A_4))\ , \\
((A_1A_2)(A_3A_4))\ , \\
((A_1(A_2A_3))A_4)\ , \\
(((A_1A_2)A_3)A_4)\ .
\end{array}
$$

问题3：matrix-chain multiplication (续)

- Recursively define the value of an optimal solution.

$$m[i,j] = \begin{cases} 0 & \text{if } i = j\ , \\ \min\limits_{i \le k < j} \{m[i,k] + m[k+1,j] + p_{i-1}p_k p_j\} & \text{if } i < j\ . \end{cases}$$

$$\begin{aligned}
&(A_1(A_2(A_3A_4)))\ , \\
&(A_1((A_2A_3)A_4))\ , \\
&((A_1A_2)(A_3A_4))\ , \\
&((A_1(A_2A_3))A_4)\ , \\
&(((A_1A_2)A_3)A_4)\ .
\end{aligned}$$

问题3： matrix-chain multiplication (续)

- Compute the value of an optimal solution.
  - 计算的顺序是什么？

```
for l = 2 to n                    // l is the chain length
    for i = 1 to n − l + 1
        j = i + l − 1
        m[i, j] = ∞
        for k = i to j − 1
            q = m[i, k] + m[k + 1, j] + p_{i−1} p_k p_j
            if q < m[i, j]
                m[i, j] = q
```

$$
\begin{aligned}
&(A_1(A_2(A_3A_4)))\ ,\\
&(A_1((A_2A_3)A_4))\ ,\\
&((A_1A_2)(A_3A_4))\ ,\\
&((A_1(A_2A_3))A_4)\ ,\\
&(((A_1A_2)A_3)A_4)\ .
\end{aligned}
$$

问题3：matrix-chain multiplication (续)

- Construct an optimal solution from computed information.
  - 怎样记住得到最优解的过程？
  - 如何基于此输出最优解？

```
for l = 2 to n                // l is the chain length
    for i = 1 to n − l + 1
        j = i + l − 1
        m[i, j] = ∞
        for k = i to j − 1
            q = m[i, k] + m[k + 1, j] + p_{i−1} p_k p_j
            if q < m[i, j]
                m[i, j] = q
                s[i, j] = k
```

```
PRINT-OPTIMAL-PARENS(s, i, j)
if i == j
    print "A"_i
else print "("
    PRINT-OPTIMAL-PARENS(s, i, s[i, j])
    PRINT-OPTIMAL-PARENS(s, s[i, j] + 1, j)
    print ")"
```

$$
\begin{aligned}
&(A_1(A_2(A_3A_4)))\ ,\\
&(A_1((A_2A_3)A_4))\ ,\\
&((A_1A_2)(A_3A_4))\ ,\\
&((A_1(A_2A_3))A_4)\ ,\\
&(((A_1A_2)A_3)A_4)\ .
\end{aligned}
$$

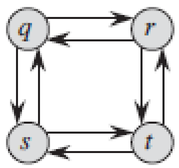
问题4：dynamic programming的运行时间

- 你认为，决定dynamic programming运行时间的要素有哪些？
  - number of subproblems overall
  - number of choices we look at for each subproblem

- Compute the value of an optimal solution有哪两种策略？
  - Top-down with memoization
  - Bottom-up method
- 它们在运行时间上有什么区别？

问题5：unweighted shortest/longest simple path

- unweighted longest simple path为什么不具有最优子结构？
- unweighted shortest simple path为什么不存在这个问题？



问题6：longest common subsequence

- longest common subsequence要解决什么问题？

$$S_1 = \mathrm{ACCGGTCGAGTGCGCGGAAGCCGGCCGAA}$$

$$S_2 = \mathrm{GTCGTTCGGAATGCCGTTGCTCTGTAAA}$$

$$\mathrm{GTCGTCGGAAGCCGGCCGAA}$$

问题6：longest common subsequence (续)

- Characterize the structure of an optimal solution.
  - 最优子结构是什么？

1. If $x_m = y_n$, then $z_k = x_m = y_n$ and Z_{k-1} is an LCS of X_{m-1} and Y_{n-1}.
2. If $x_m \neq y_n$, then $z_k \neq x_m$ implies that Z is an LCS of X_{m-1} and Y.
3. If $x_m \neq y_n$, then $z_k \neq y_n$ implies that Z is an LCS of X and Y_{n-1}.

$$S_1 = \text{ACCGGTCGAGTGCGCGGAAGCCGGCCGAA}$$

$$S_2 = \text{GTCGTTCGGAATGCCGTTGCTCTGTAAA}$$

GTCGTCGGAAGCCGGCCGAA

问题6：longest common subsequence (续)

- Recursively define the value of an optimal solution.

$$c[i,j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0\,, \\ c[i-1,j-1]+1 & \text{if } i,j > 0 \text{ and } x_i = y_j\,, \\ \max(c[i,j-1], c[i-1,j]) & \text{if } i,j > 0 \text{ and } x_i \neq y_j\,. \end{cases}$$

$$S_1 = \mathtt{ACCGGTCGAGTGCGCGGAAGCCGGCCGAA}$$

$$S_2 = \mathtt{GTCGTTCGGAATGCCGTTGCTCTGTAAA}$$

GTCGTCGGAAGCCGGCCGAA

问题6：longest common subsequence (续)

- Compute the value of an optimal solution.
  - 计算的顺序是什么？

```
for i = 1 to m
    for j = 1 to n
        if x_i == y_j
            c[i, j] = c[i − 1, j − 1] + 1
        elseif c[i − 1, j] ≥ c[i, j − 1]
            c[i, j] = c[i − 1, j]
        else c[i, j] = c[i, j − 1]
```

$$S_1 = \text{ACCGGTCGAGTGCGCGGAAGCCGGCCGAA}$$

$$S_2 = \text{GTCGTTCGGAATGCCGTTGCTCTGTAAA}$$

GTCGTCGGAAGCCGGCCGAA

问题6：longest common subsequence (续)

- Construct an optimal solution from computed information.
  - 怎样记住得到最优解的过程？
  - 如何基于此输出最优解？

```
for i = 1 to m
    c[i,0] = 0
for j = 0 to n
    c[0,j] = 0
for i = 1 to m
    for j = 1 to n
        if x_i == y_j
            c[i,j] = c[i-1,j-1] + 1
            b[i,j] = "↖"
        elseif c[i-1,j] ≥ c[i,j-1]
            c[i,j] = c[i-1,j]
            b[i,j] = "↑"
        else c[i,j] = c[i,j-1]
            b[i,j] = "←"
```

```
PRINT-LCS(b, X, i, j)
if i == 0 or j == 0
    return
if b[i,j] == "↖"
    PRINT-LCS(b, X, i-1, j-1)
    print x_i
elseif b[i,j] == "↑"
    PRINT-LCS(b, X, i-1, j)
else PRINT-LCS(b, X, i, j-1)
```

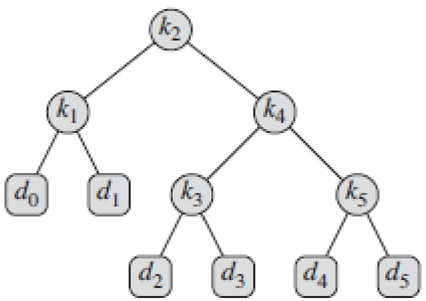
$$S_1 = \mathrm{ACCGGTCGAGTGCGCGGAAGCCGGCCGAA}$$

$$S_2 = \mathrm{GTCGTTCGGAATGCCGTTGCTCTGTAAA}$$

GTCGTCGGAAGCCGGCCGAA

问题7：optimal binary search trees

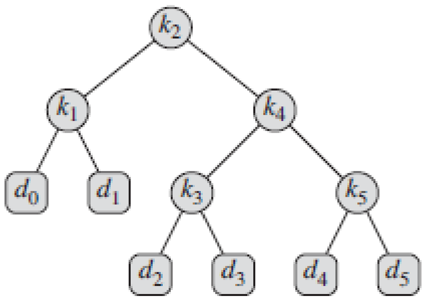
- optimal binary search trees要解决什么问题？



i	0	1	2	3	4	5
p_i		0.15	0.10	0.05	0.10	0.20
q_i	0.05	0.10	0.05	0.05	0.05	0.10

问题7：optimal binary search trees (续)

- Characterize the structure of an optimal solution.
  - 最优子结构是什么？



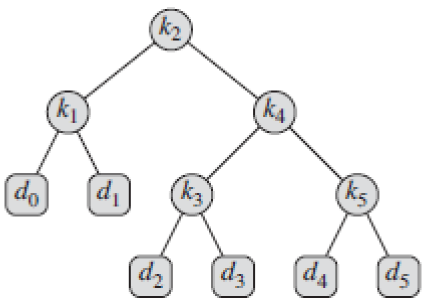
i	0	1	2	3	4	5
p_i		0.15	0.10	0.05	0.10	0.20
q_i	0.05	0.10	0.05	0.05	0.05	0.10

问题7：optimal binary search trees (续)

- Recursively define the value of an optimal solution.

$$e[i,j] = \begin{cases} q_{i-1} & \text{if } j = i-1\,, \\ \min\limits_{i \le r \le j} \{e[i,r-1] + e[r+1,j] + w(i,j)\} & \text{if } i \le j\,. \end{cases}$$

$$w(i,j) = w(i,r-1) + p_r + w(r+1,j)$$

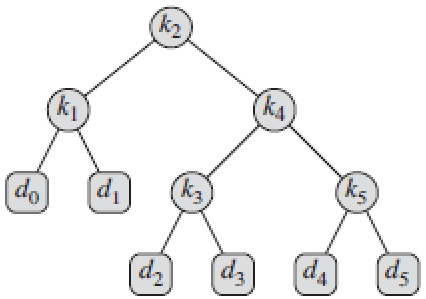


i	0	1	2	3	4	5
p_i		0.15	0.10	0.05	0.10	0.20
q_i	0.05	0.10	0.05	0.05	0.05	0.10

问题7：optimal binary search trees (续)

- Compute the value of an optimal solution.
  - 计算的顺序是什么？

```
for l = 1 to n
    for i = 1 to n − l + 1
        j = i + l − 1
        e[i, j] = ∞
        w[i, j] = w[i, j − 1] + p_j + q_j
        for r = i to j
            t = e[i, r − 1] + e[r + 1, j] + w[i, j]
            if t < e[i, j]
                e[i, j] = t
```

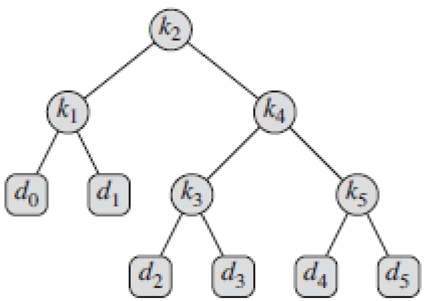


i	0	1	2	3	4	5
p_i		0.15	0.10	0.05	0.10	0.20
q_i	0.05	0.10	0.05	0.05	0.05	0.10

问题7：optimal binary search trees (续)

- Construct an optimal solution from computed information.
  - 怎样记住得到最优解的过程？

```
for l = 1 to n
    for i = 1 to n − l + 1
        j = i + l − 1
        e[i, j] = ∞
        w[i, j] = w[i, j − 1] + p_j + q_j
        for r = i to j
            t = e[i, r − 1] + e[r + 1, j] + w[i, j]
            if t < e[i, j]
                e[i, j] = t
                root[i, j] = r
```



i	0	1	2	3	4	5
p_i		0.15	0.10	0.05	0.10	0.20
q_i	0.05	0.10	0.05	0.05	0.05	0.10