[Knu22, Chapter 7.2.2.1]

Algorithm X

Lou & Anakin

February 20, 2023

Outline

Section 1

Review of Exact Cover

Exact Cover Problems

- The goal of Exact Cover is to select subsets of some list of items according to certain criterion:
  - Cover: Select subsets such that their **union** is all items
  - Exact: Each item is in **exactly one** subset
- In 1972, Richard Karp proved that Exact Cover, among 20 other problems, is NP-Complete
  - Easy to verify solutions in polynomial time
  - Hard to solve, best known solutions run in exponential time
  - Can simulate (or reduce) other problems in NP using Exact Cover

An Example of Exact Cover

Goal: Select rows such that each column in the selection has one 1

$$\begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

We can abstract this to options containing items

$$\begin{array}{lll} 1\colon [a, c] & 2\colon [e] & 3\colon [b, d] \\ 4\colon [c, e] & 5\colon [a, c, d] & \end{array}$$

Answer: Select options 1, 2, and 3

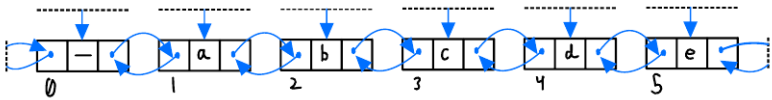
Recursively Solving Exact Cover Problems

In trying to solve the previous problem, you may have naturally found a recursive algorithm to find a solution

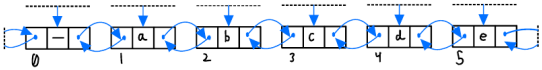
```
    FindCover(Options, Items, Cover, i):
 1:   if Cover is a cover:
 2:     terminate successfully
 3:   if no option in Options contains i:
 4:     terminate unsuccessfully
 5:
 6:   I ← options in Options that contain i
 7:   Options ← Options \I
 8:   for each O in I:
 9:     j ← an item still not covered
10:     FindCover(Options, Cover ∪ { O }, j)
```

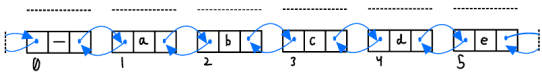
Section 2

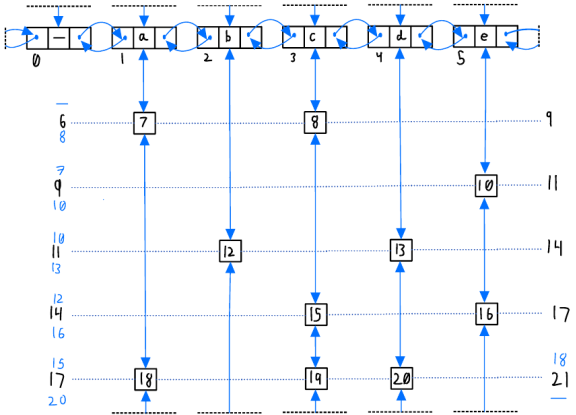
Data Structure



- We create a linked list of our *items*, where each *item* will connect to a linked list representing its *options*


a
b
c
d
e
0
1
2
3
4
5


a
b
c
d
e
0
1
2
3
4
5
6
7
8
9
9
10
11
11
12
13
14
14
15
16
17
17
18
19
20
21

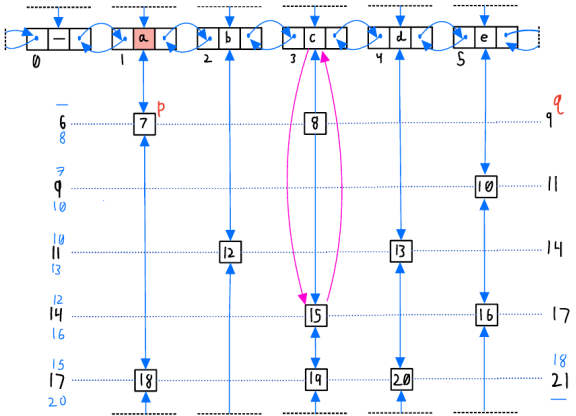

a
b
c
d
e
0
1
2
3
4
5
7
8
10
12
13
15
16
18
19
20

Section 3

Algorithm X

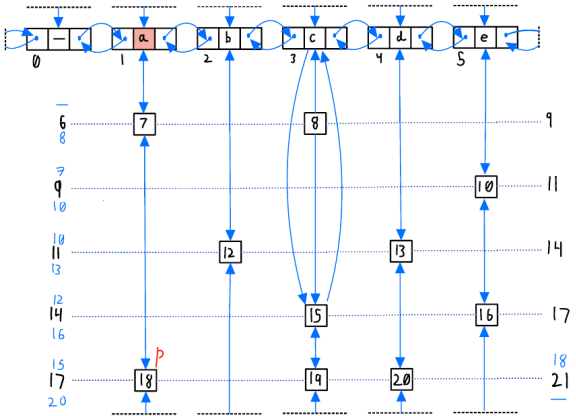
Hide(p)

- Removes the option a node p is from (so that option can no longer be chosen)


0
1
2
3
4
5
a
b
c
d
e
p
q
6
7
8
9
9
10
11
9
10
11
12
13
14
10
13
12
14
15
16
17
16
15
17
18
19
20
21
18
20

Cover(i)

- Hides all options that could cover item i
  - Once we choose an option for i, we cannot choose any other options including i

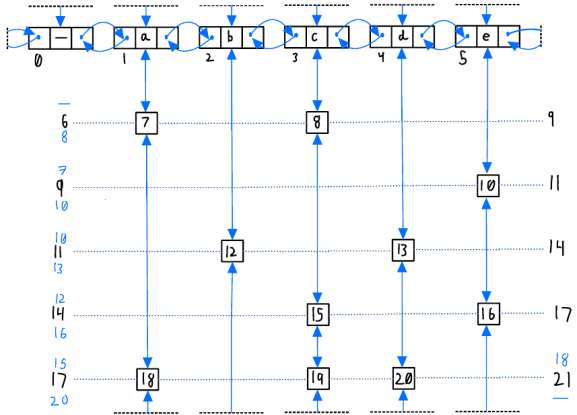

a
b
c
d
e
0
1
2
3
4
5
7
8
9
10
12
13
15
16
18
19
20
21

Algorithm X

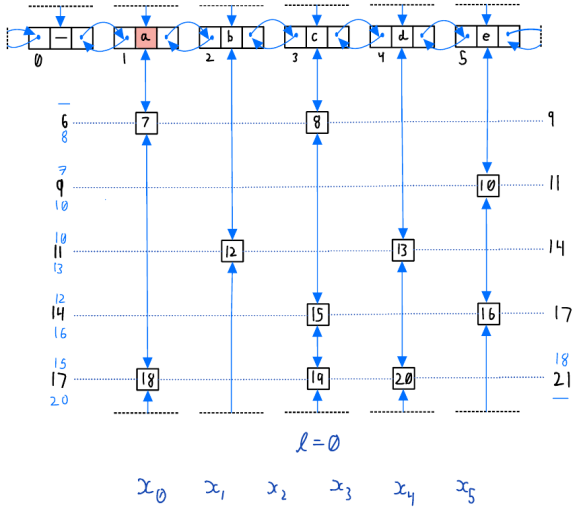
ALGORITHM X(*Options, Items*):

1: Set up the dancing links, $\ell \leftarrow 0$ ⟨⟨*ℓ is our level*⟩⟩
2: **if** all items have been covered:
3: Report success, visit answer, and **goto** Line 13
4: $i \leftarrow$ item not yet covered
5: COVER(i) then $x_\ell \leftarrow i$.down
6: **if** $x_\ell = i$:
7: **goto** Line 12 ⟨⟨*no options left to try*⟩⟩
8: **else**:
9: $O \leftarrow$ option corresponding to x_ℓ
10: COVER every item in O, then **goto** Line 2
11: UNCOVER items $\neq i$ in option corresponding to x_ℓ, **goto** Line 6
12: UNCOVER(i)
13: **if** $\ell = 0$, terminate, **else** $\ell = \ell - 1$, **goto** Line 11

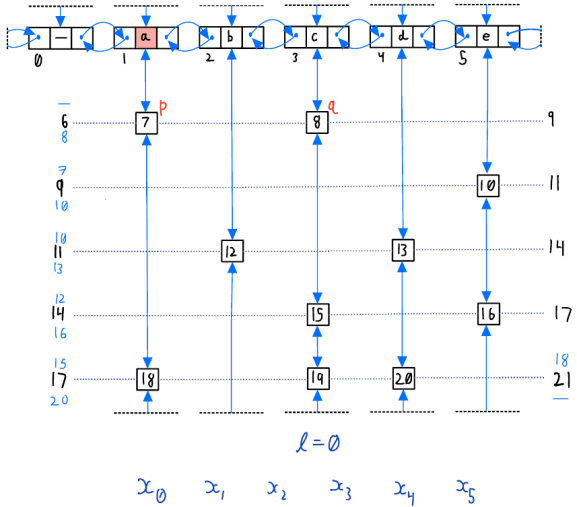
Initialize Data Structure



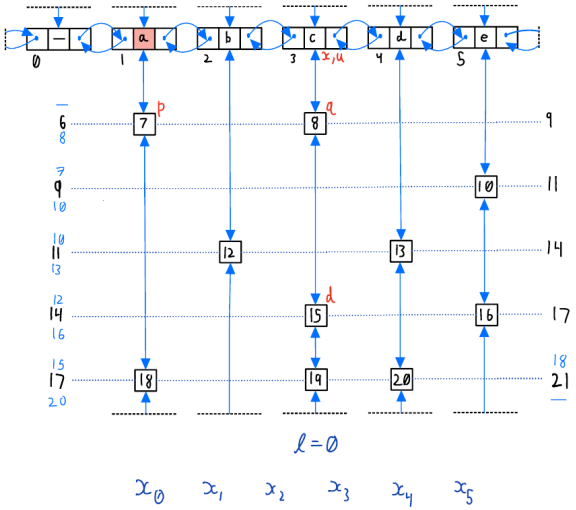
Initialize ℓ and x_ℓ's



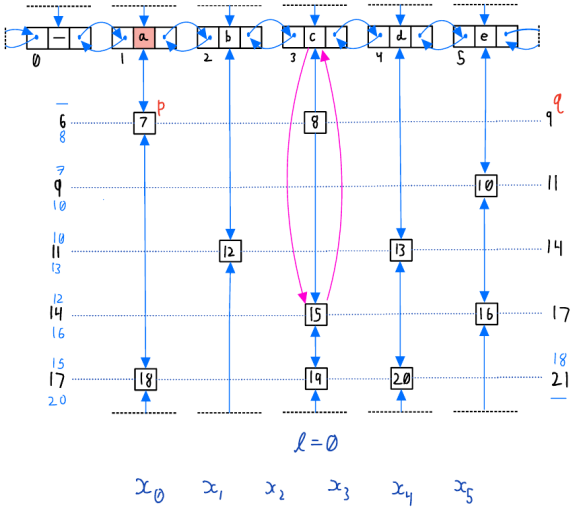
Select $i = a$, cover(a)



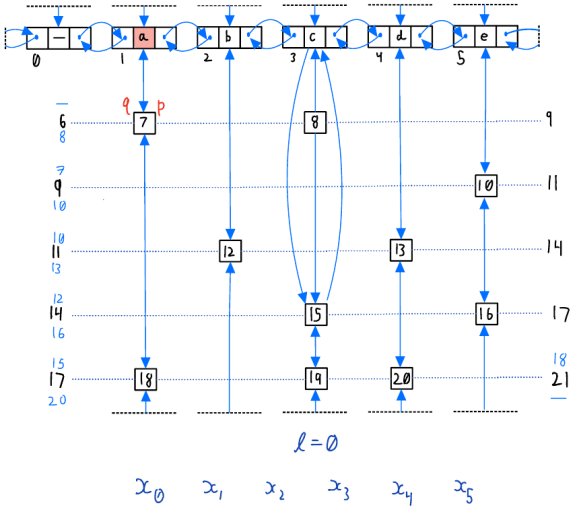
hide(7)



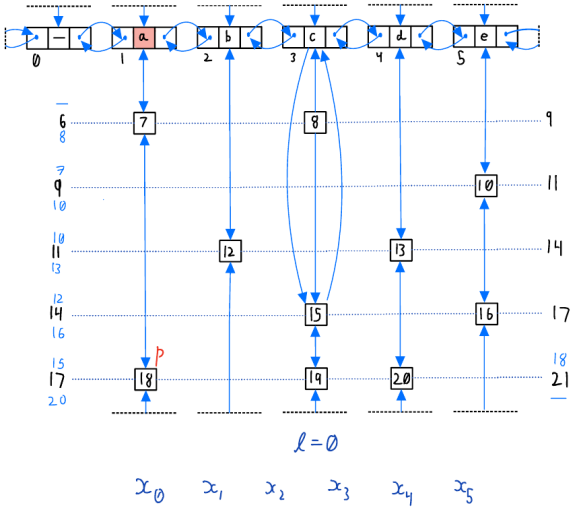
hide(7)


0
1
2
3
4
5
a
b
c
d
e
p
q
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
$\ell = 0$
x_0 x_1 x_2 x_3 x_4 x_5

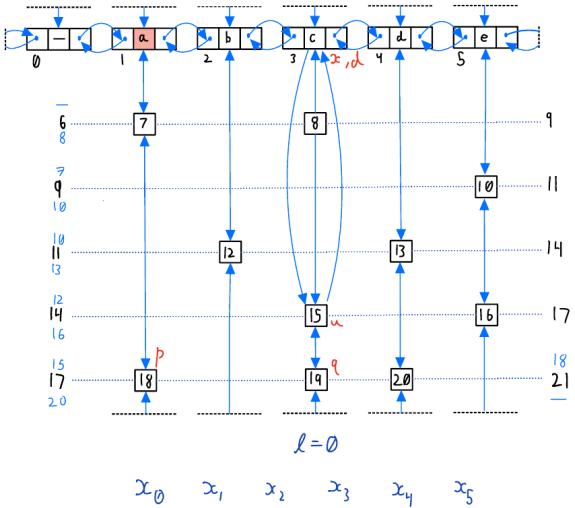
hide(7)



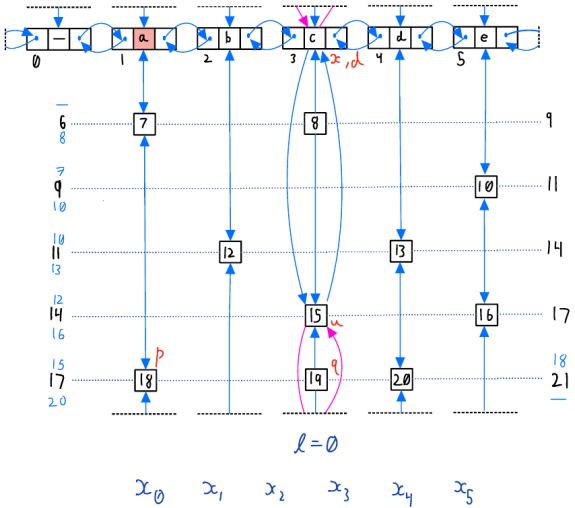
hide(18)


0
1
2
3
4
5
a
b
c
d
e
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
$\ell = 0$
x_0 x_1 x_2 x_3 x_4 x_5

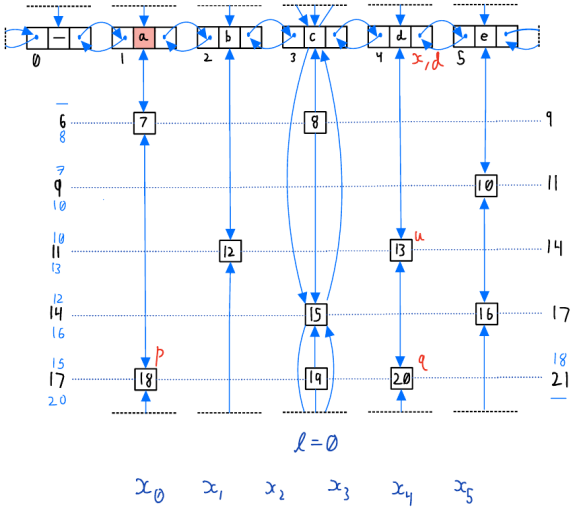
hide(18)


0
1
2
3
4
5
a
b
c
d
e
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
$\ell = 0$
x_0 x_1 x_2 x_3 x_4 x_5

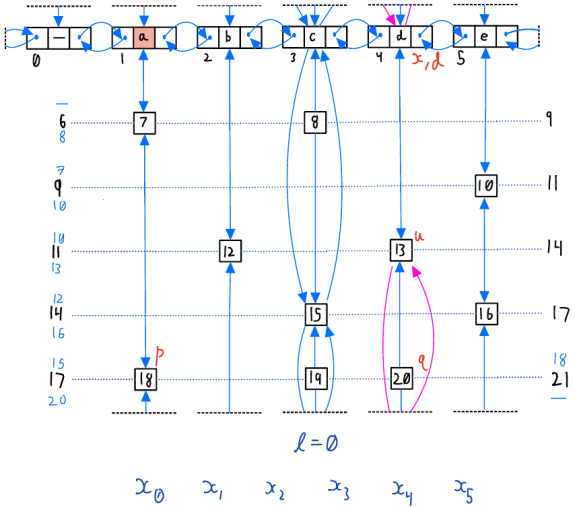
hide(18)



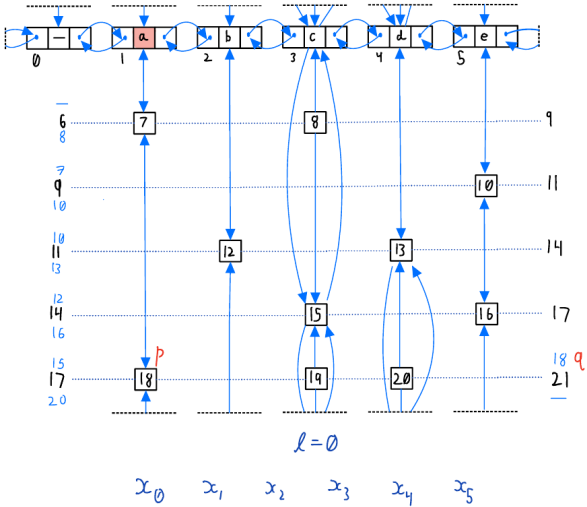
hide(18)


0
1
2
3
4
5
x1 d
6
8
7
8
9
9
10
11
10
11
13
12
13
u
14
12
14
16
15
16
17
12
14
16
15
17
20
18
p
19
20
q
18
21
ℓ = 0
x0
x1
x2
x3
x4
x5

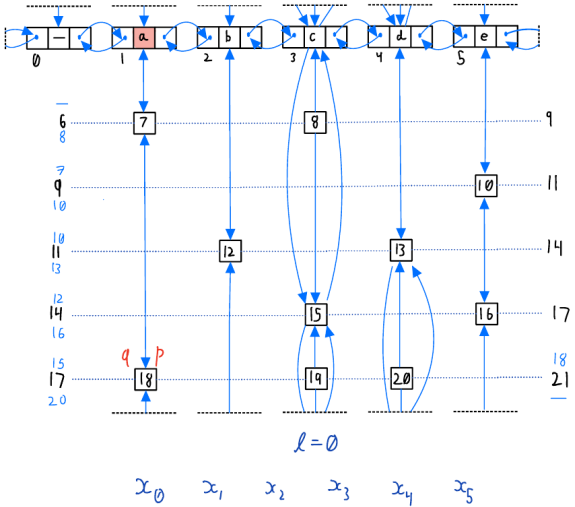
hide(18)


0
1
2
3
4
5
a
b
c
d
e
x1 d
6
8
9
10
11
13
12
14
16
15
17
20
18
21
7
8
9
10
11
12
13
14
15
16
17
18
19
20
p
q
u
ℓ = 0
x0 x1 x2 x3 x4 x5

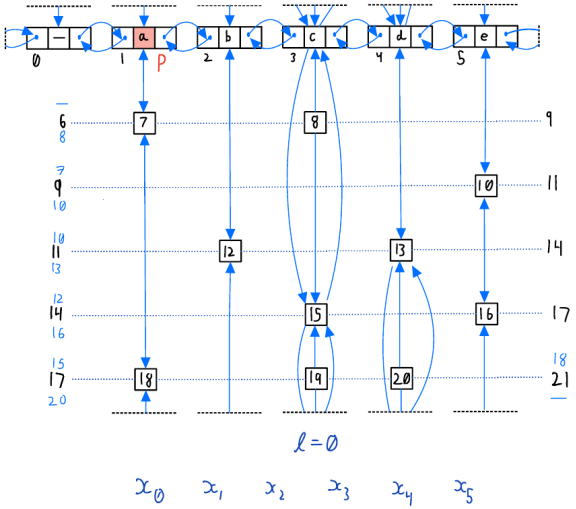
hide(18)



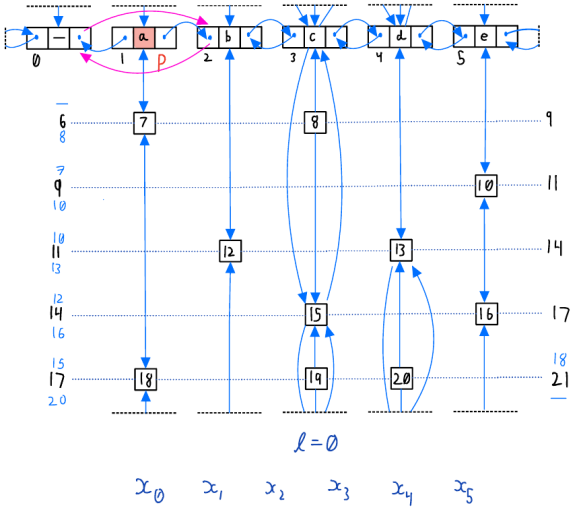
hide(18)


0
1
2
3
4
5
a
b
c
d
e
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
q
p
ℓ = 0
x_0 x_1 x_2 x_3 x_4 x_5

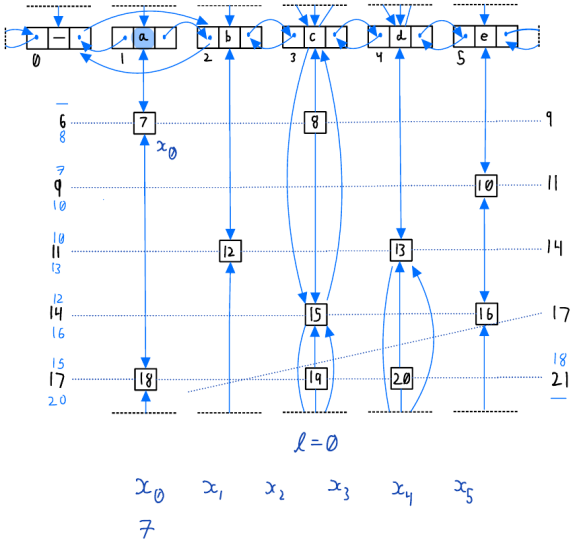
remove a



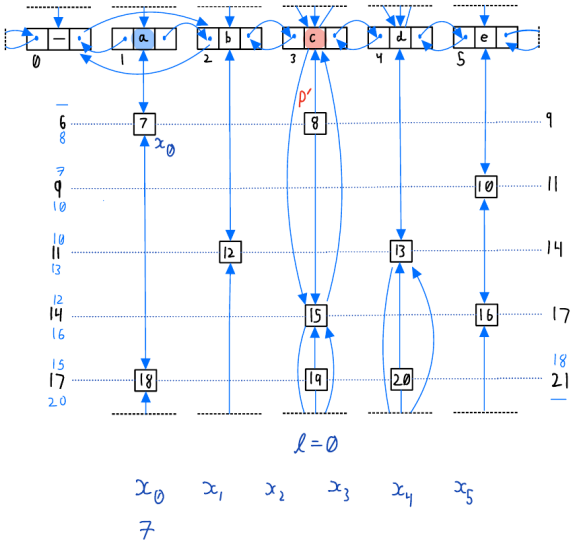
remove a



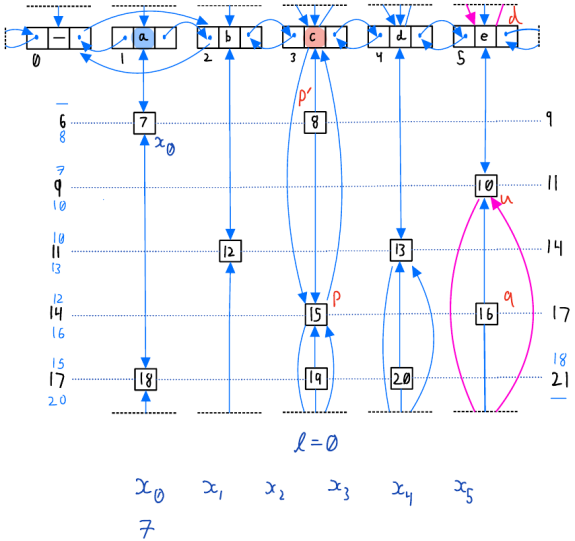
$\ell = 0,\ x_0 = 7$


0
1
2
3
4
5
a
b
c
d
e
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
x_0
$\ell = 0$
x_0 x_1 x_2 x_3 x_4 x_5
7

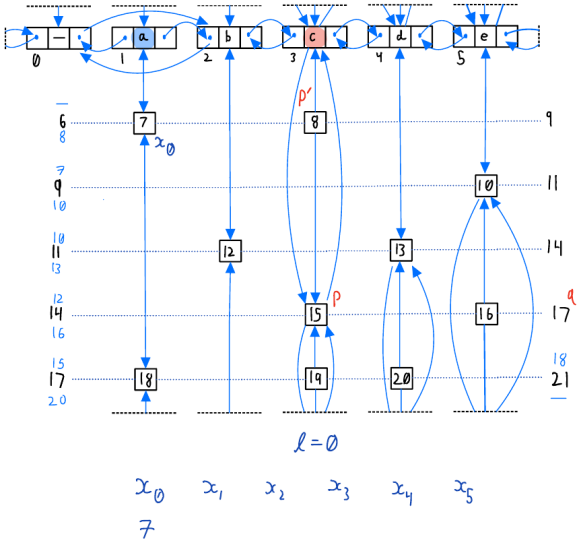
$\mathrm{cover}(c)$



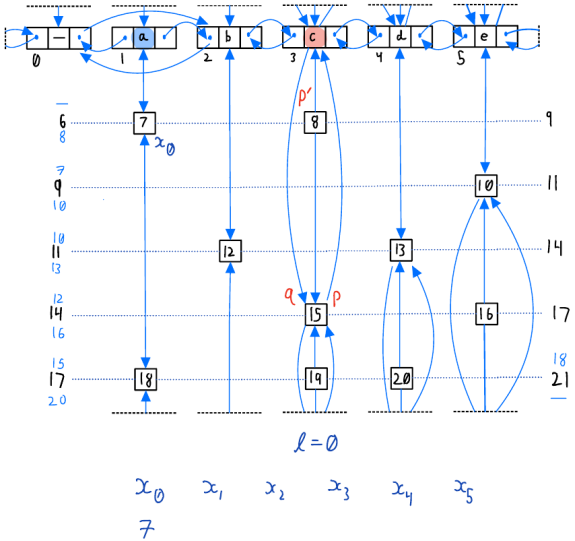
hide(15)


0
1
2
3
4
5
a
b
c
d
e
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
x_0
p'
p
q
u
$\ell = 0$
x_0 x_1 x_2 x_3 x_4 x_5
7

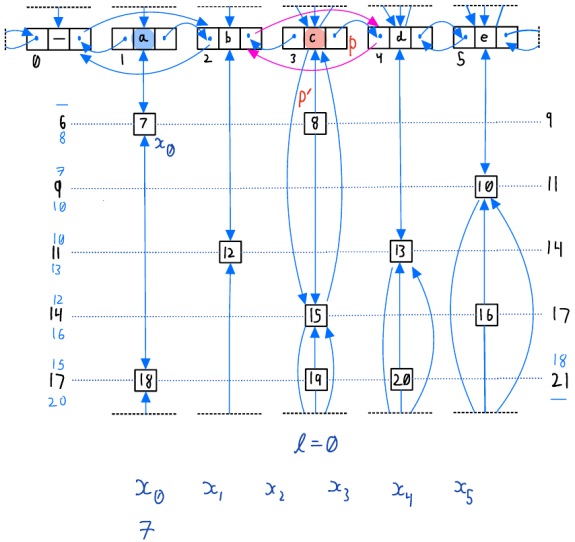
hide(15)


0
1
2
3
4
5
a
b
c
d
e
p'
p
q
x_0
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
$\ell = 0$
x_0 x_1 x_2 x_3 x_4 x_5
7

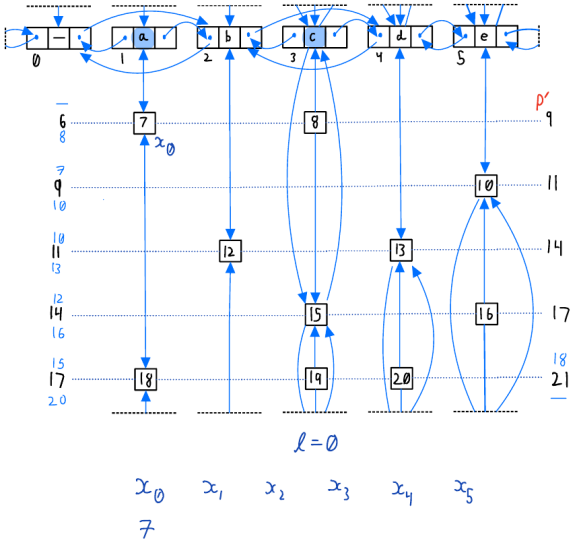
hide(15)


0
1
2
3
4
5
a
b
c
d
e
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
x_0
p'
q
p
$\ell = 0$
x_0 x_1 x_2 x_3 x_4 x_5
7

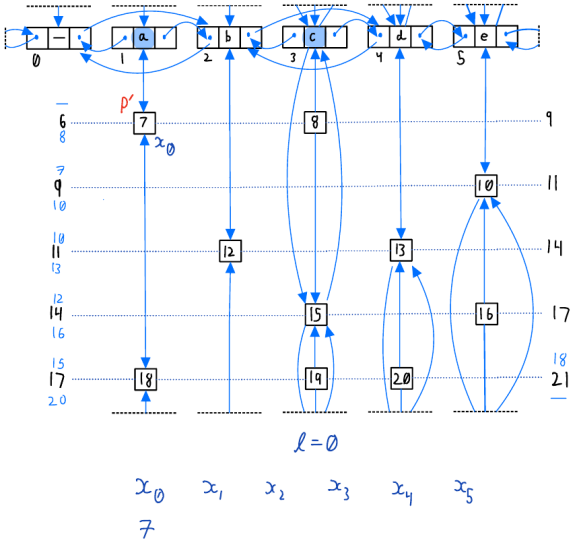
remove c



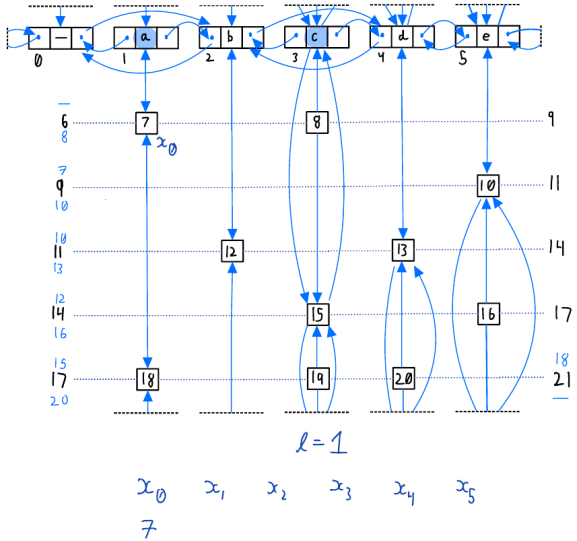
9 is a spacer, go back to 7



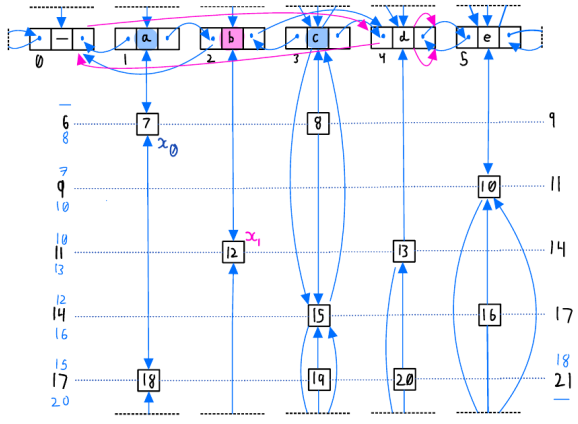
9 is a spacer, go back to 7



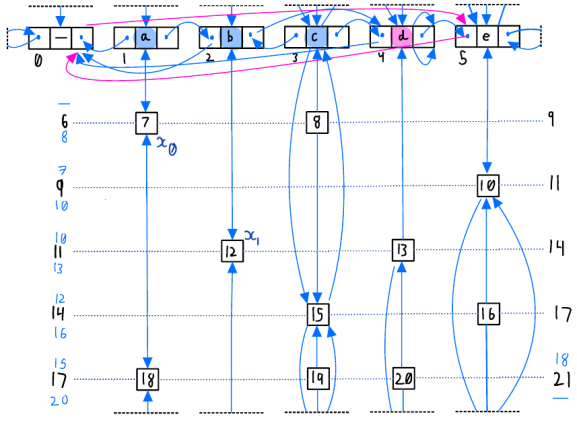
$\ell = 1$, attempt to cover b



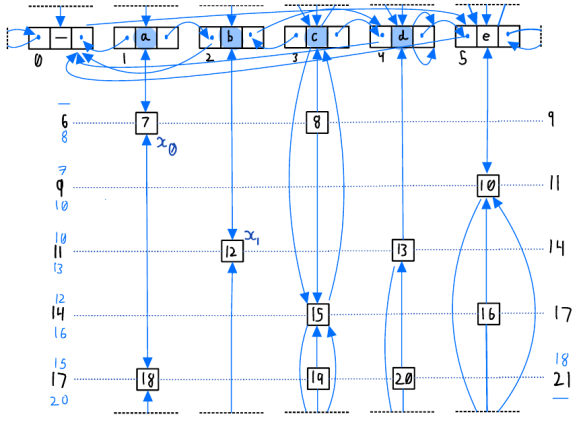
$\ell = 1$, attempt to cover b



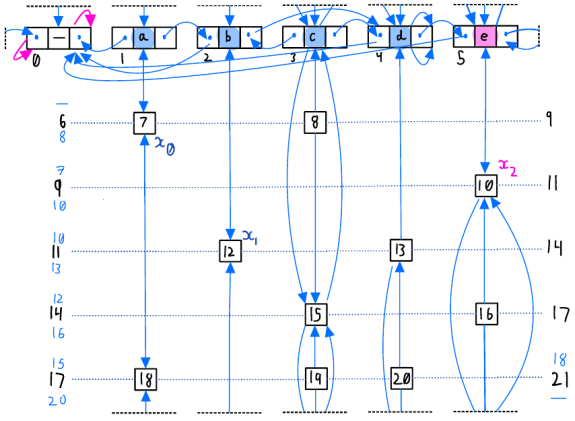
This option covers d as well



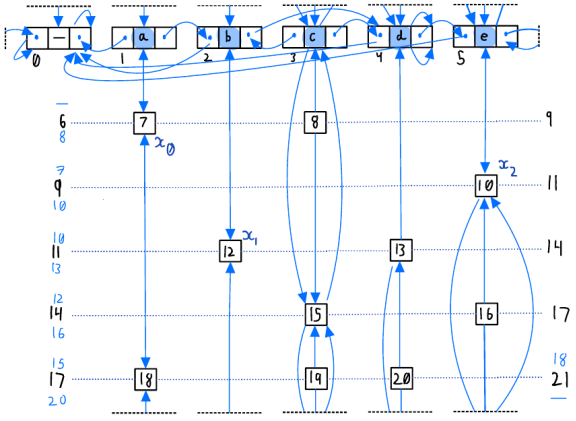
Only remaining item is e



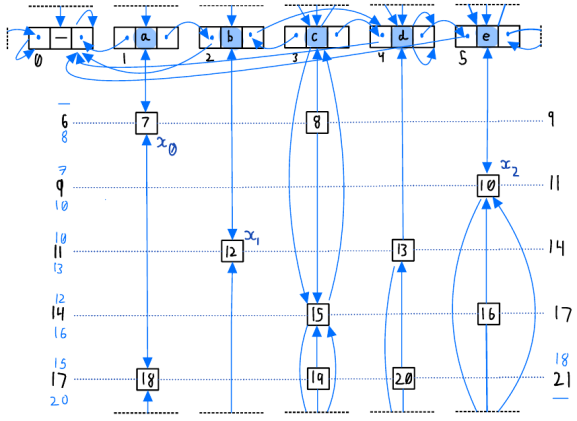
$\mathrm{cover}(e)$



$\mathrm{cover}(e)$



No options left to cover



Recovering The Answer

- Now our list contains links 7, 12, and 10.
- How do we recover what options we selected?
- Each link contains a field pointing to the corresponding option
  - Data structure design is half the battle

Questions?

Questions!

- Try to implement the Dancing Links and Algorithm X [Knu22, Chapter 7.2.2.1]
  - If anyone does this, I'll put the code on cstheory.org
- Walk through your own instance of an exact cover problem like we did by following Knuth's algorithm and go further by also walking through the UNCOVER / UNHIDE routines
  - Yes, we're serious. It is the **best** way to intuit this algorithm

Bibliography

Donald E. Knuth.
The Art of Computer Programming, Volume 4B: Combinatorial Algorithms, Part 2.
Addison-Wesley Professional, 1st edition, 2022.