L10

4800

10.11.2016

abhi shelat

Scheduling

L10

CS4800 F16

abhi shelat

	start	end
sy3333	2	3.25
en1612	1	4
ma1231	3	4
cs4102	3.5	4.75
cs4800	4	5.25
cs6051	4.5	6
sy3100	5	6.5
cs1000	7	8

problem statement

$$(a_1, \ldots, a_n)$$

$$(s_1, s_2, \ldots, s_n)$$

$(f_1, f_2, \ldots, f_n)$ (sorted) $\quad s_i < f_i$

find largest subset of activities $C=\{a_i\}$ such that

(compatible)

problem statement

$$(a_1, \ldots, a_n)$$

$$(s_1, s_2, \ldots, s_n)$$

$(f_1, f_2, \ldots, f_n)$ (s0rted) $s_i < f_i$

find largest subset of activities C={a_i} such that

(compatible)

$$a_i, a_j \in C, i < j$$

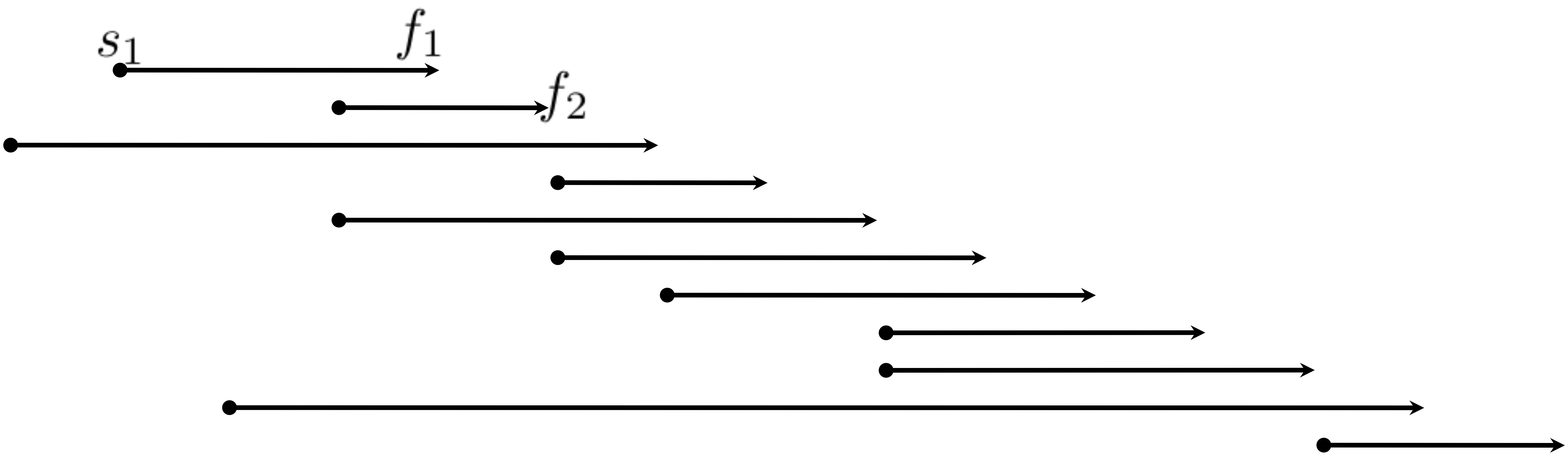
$$f_i \leq s_j$$

problem statement

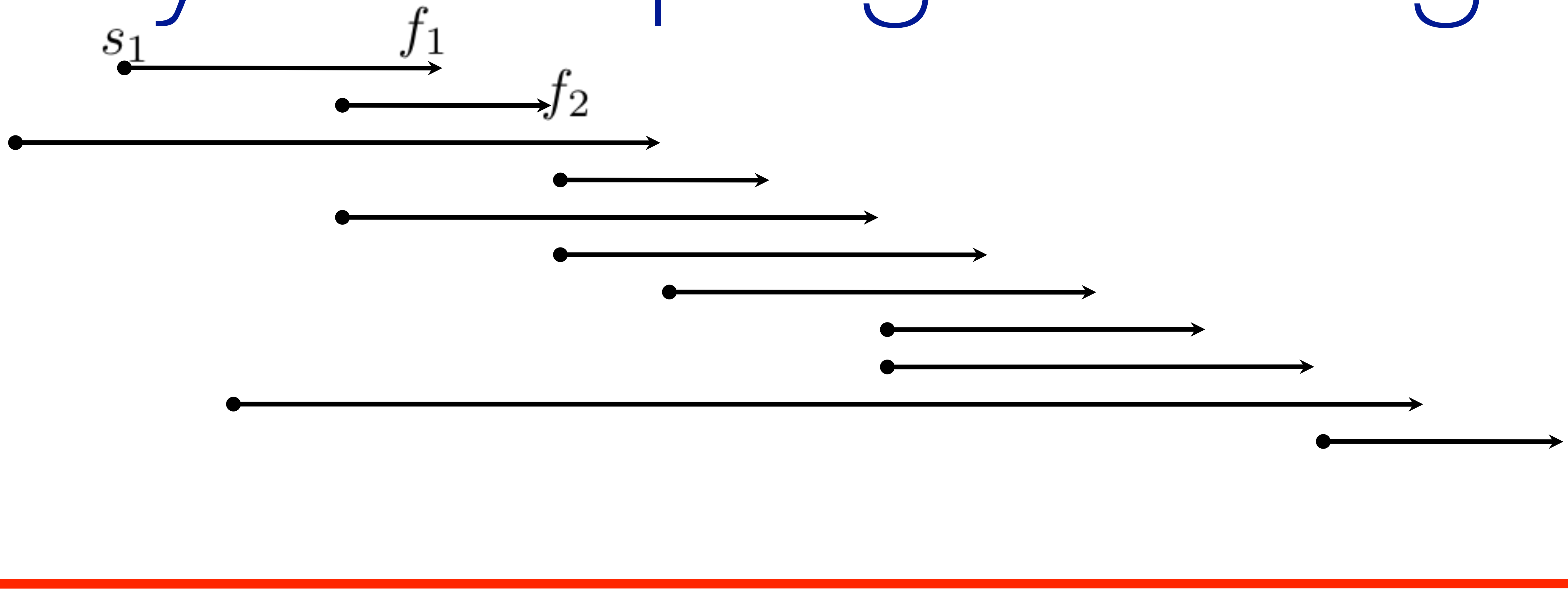
$$(a_1, \ldots, a_n)$$

$$(s_1, s_2, \ldots, s_n)$$

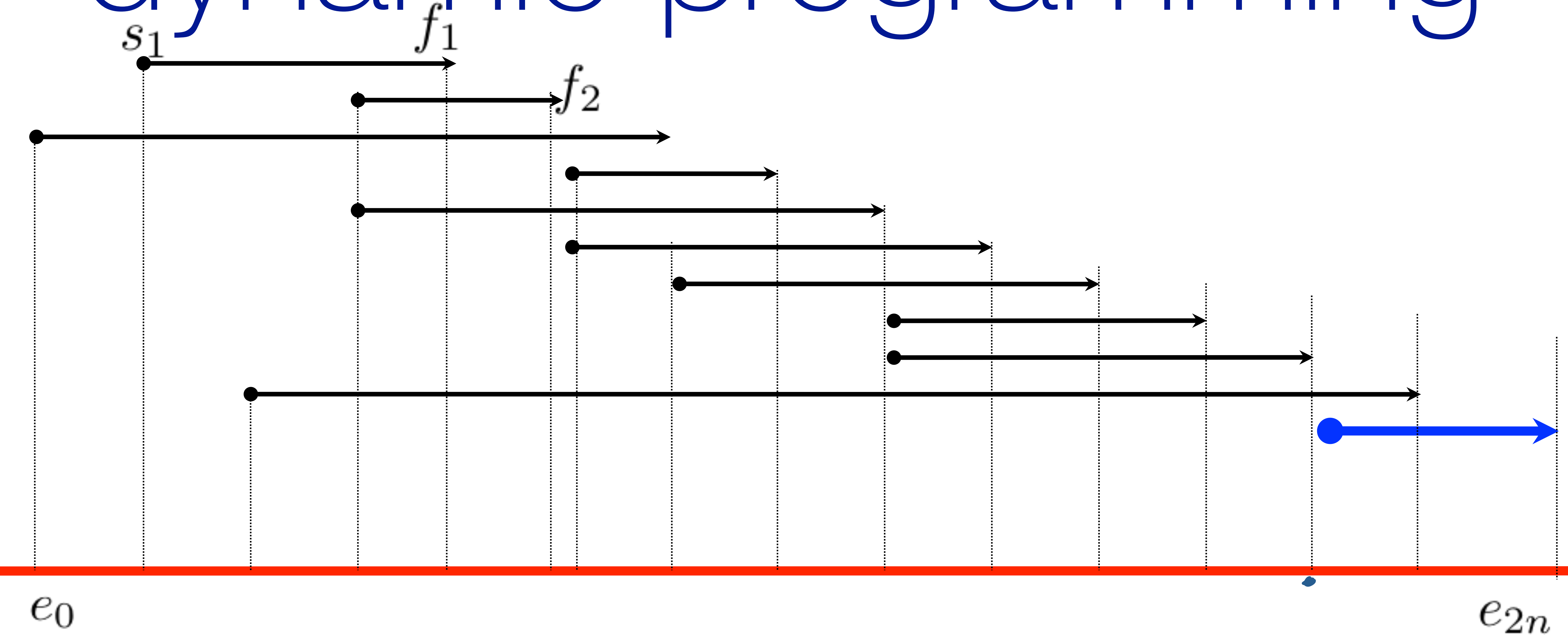
$(f_1, f_2, \ldots, f_n)$ (sorted) $\quad s_i < f_i$



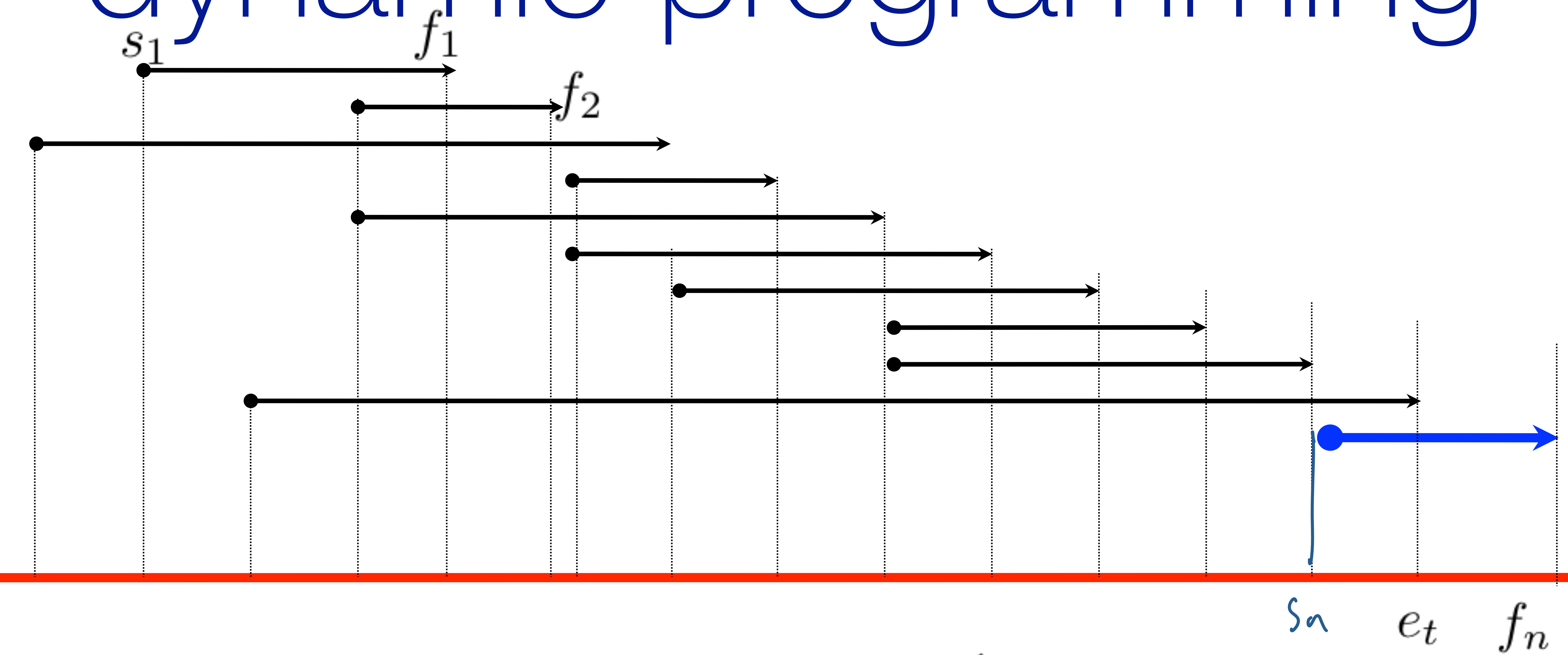
dynamic programming



dynamic programming

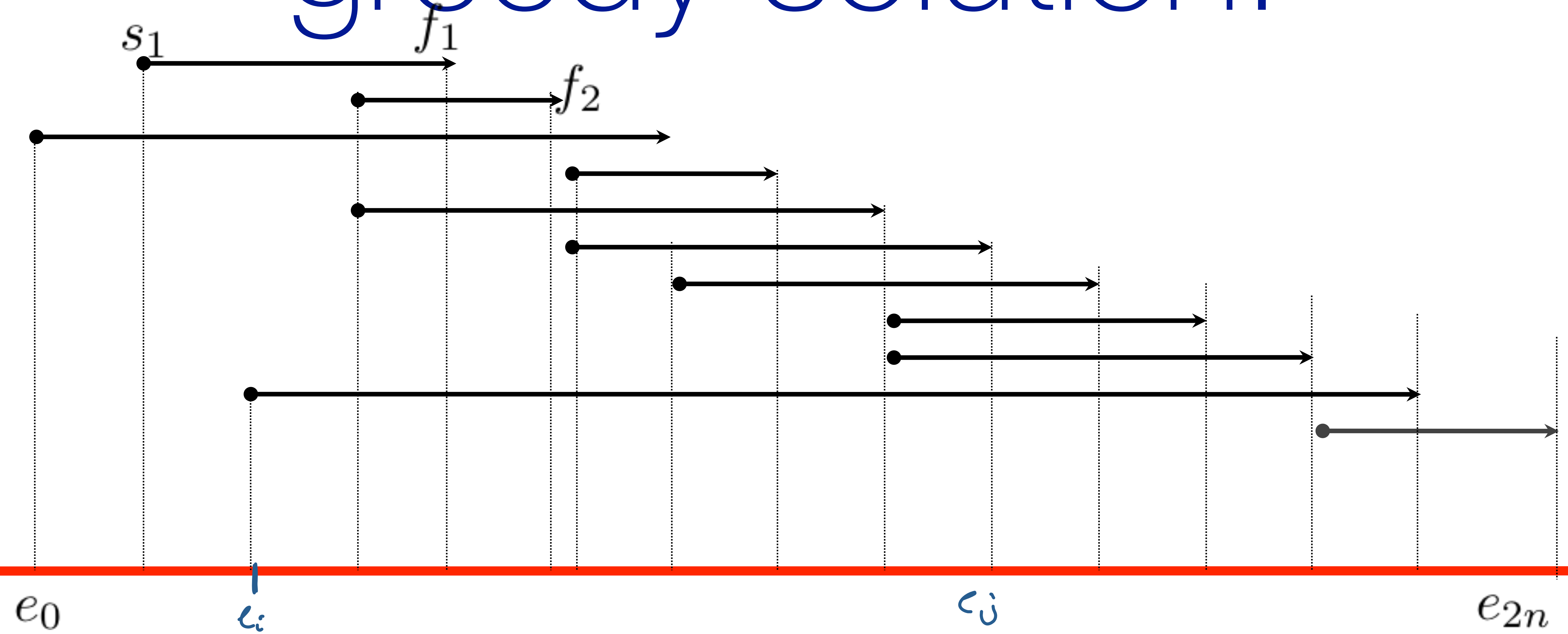


dynamic programming



$$\mathrm{BEST}_{f_n} = \max \begin{cases} \mathrm{BEST}_{s_n} + 1 & \text{in: } a_n \\ \mathrm{BEST}_{e_t} & \text{out: } a_n \end{cases}$$

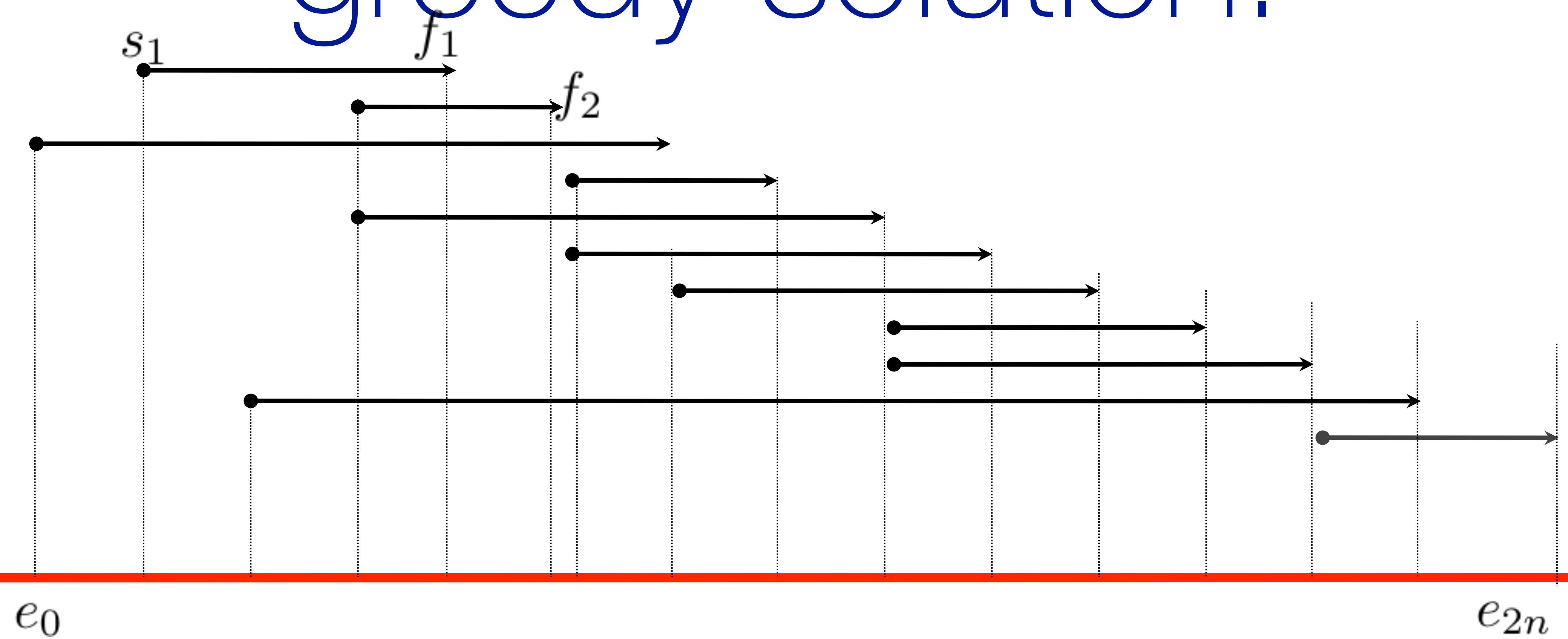
greedy solution:



definition:

$\mathrm{SOLTN}_{i,j}$

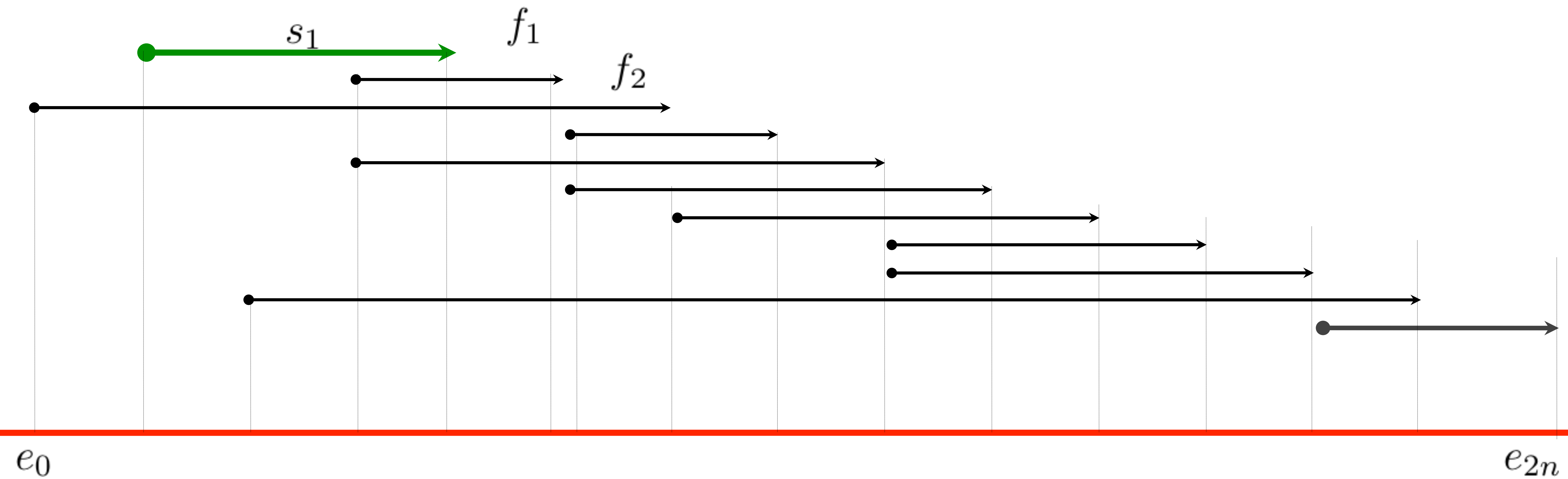
greedy solution:



$\mathrm{SOLTN}_{i,j}$

goal: $\mathrm{SOLTN}_{0,2n}$

greedy solution:

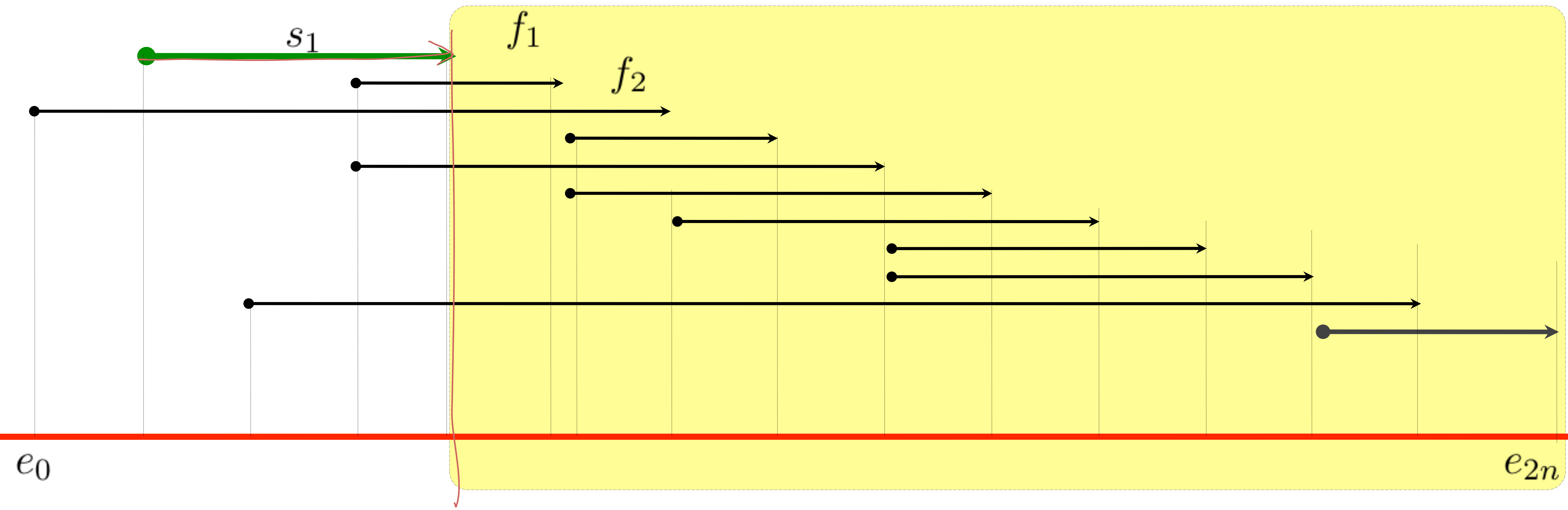


claim: the first action to finish in e[i,j] is always part of some $\textsc{soltn}_{i,j}$

claim: the first action to finish in e[i,j] is always part of some $\textsc{soltn}_{i,j}$

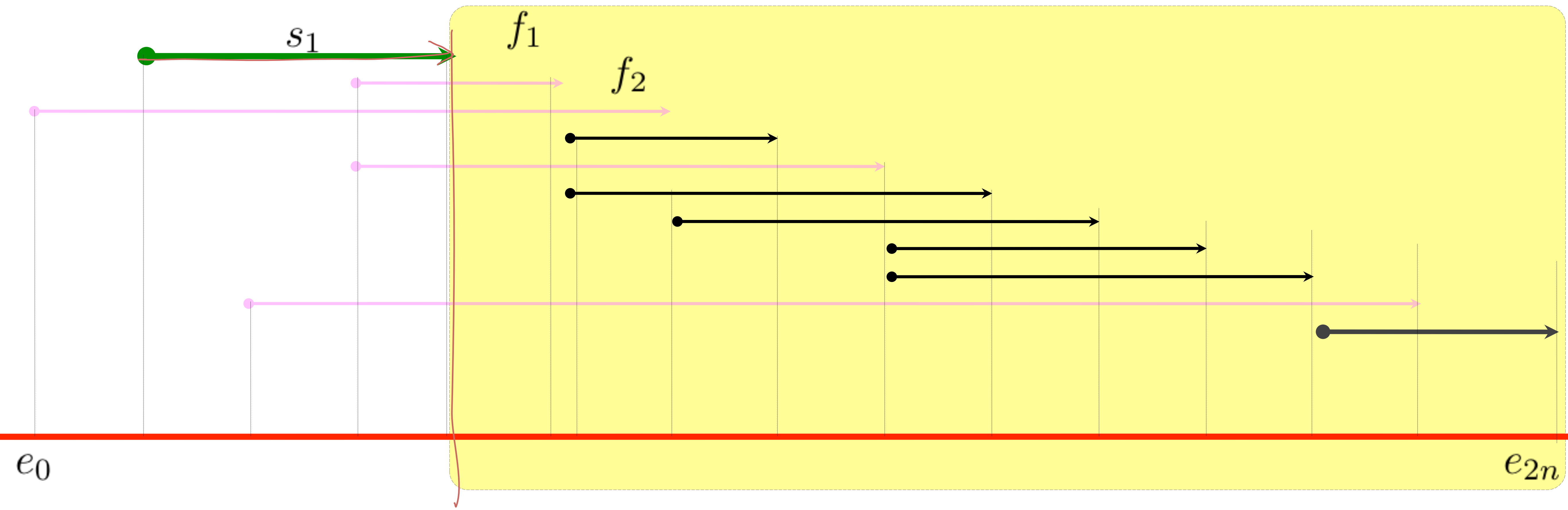
proof:

greedy solution:



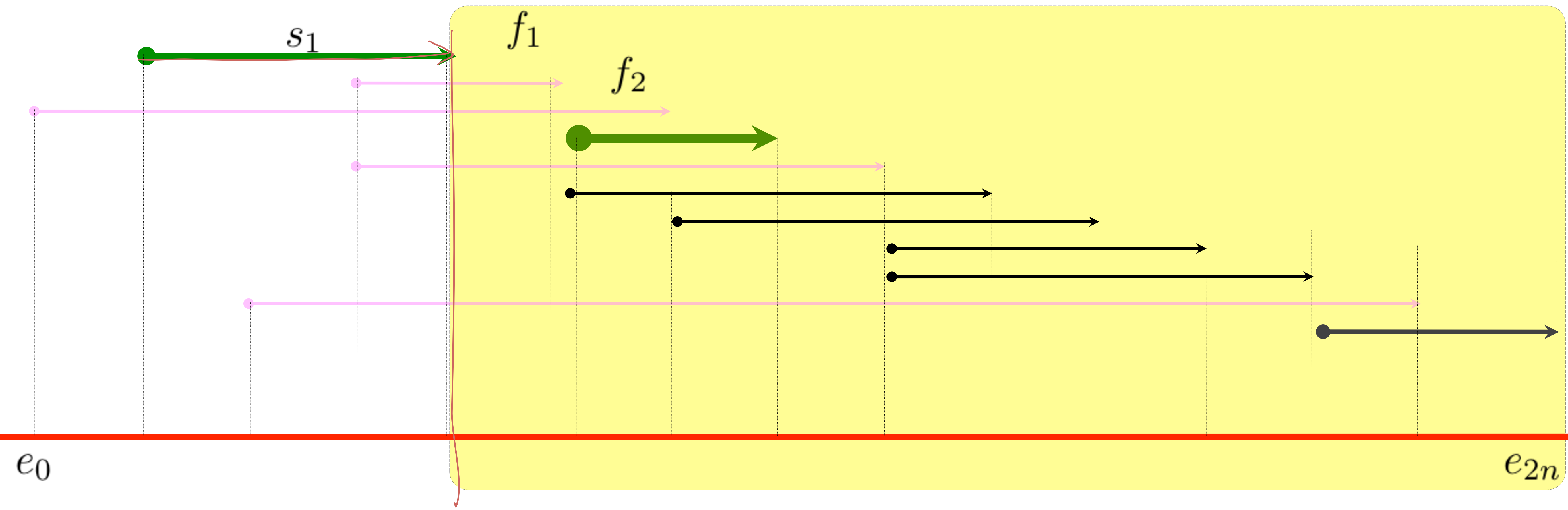
algorithm: find first event to finish. add to solution.
remove conflicting events.
continue.

greedy solution:



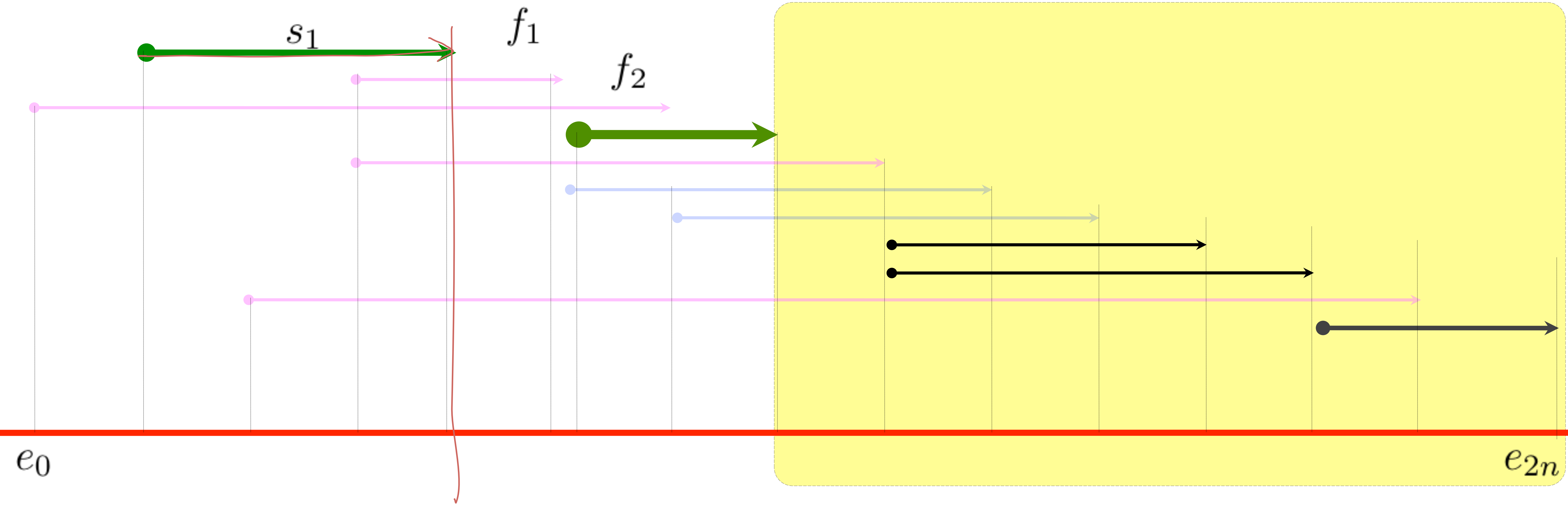
algorithm: find first event to finish. add to solution.
remove conflicting events.
continue.

greedy solution:



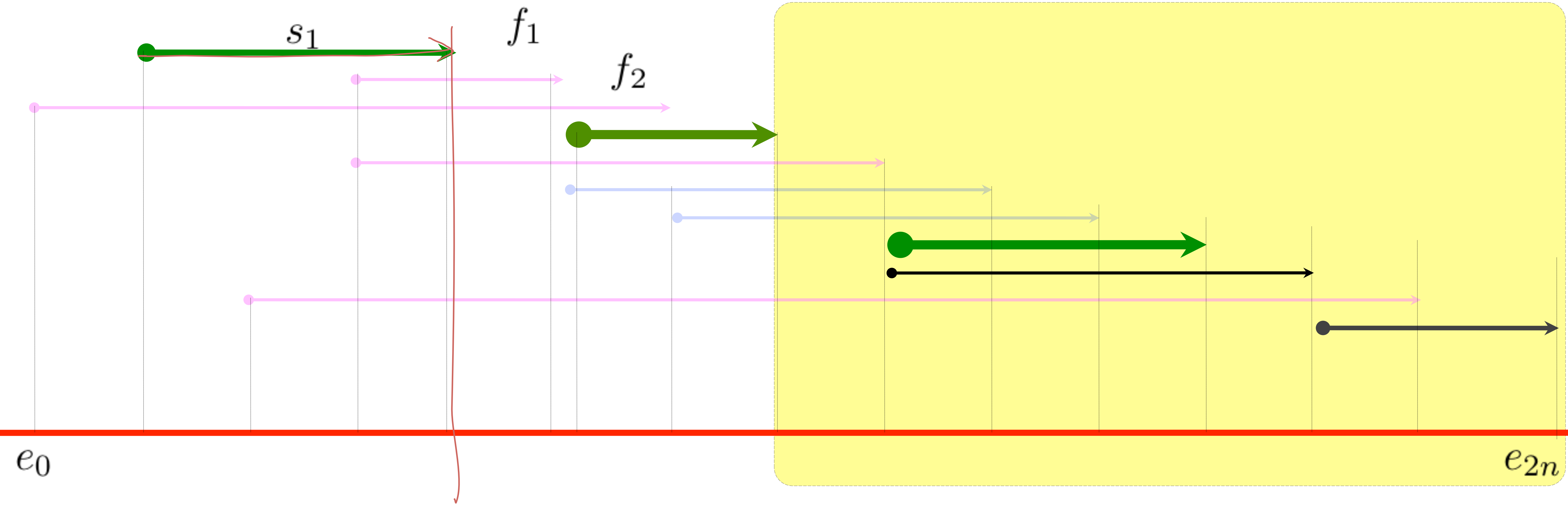
algorithm: find first event to finish. add to solution.
remove conflicting events.
continue.

greedy solution:



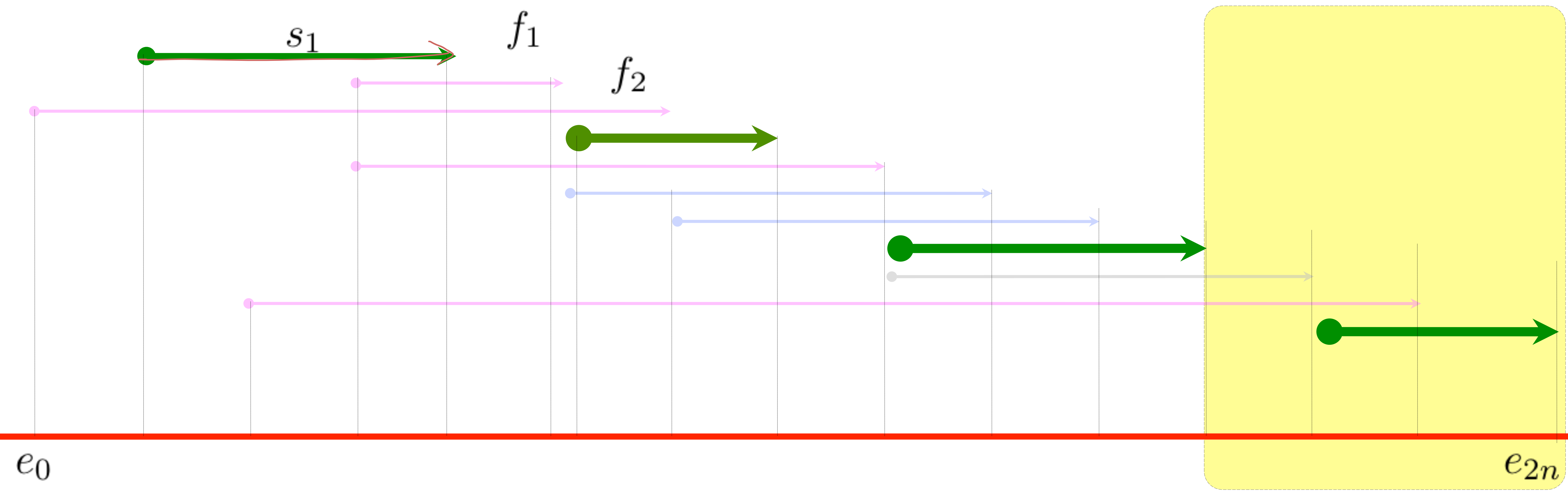
algorithm: find first event to finish. add to solution.
remove conflicting events.
continue.

greedy solution:



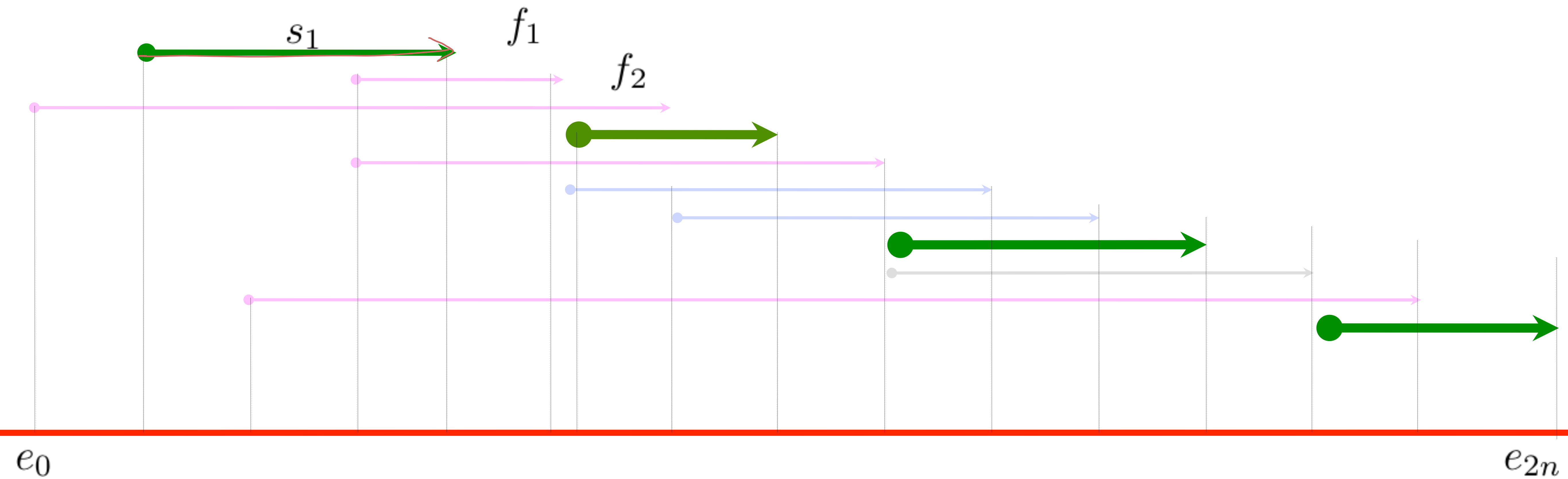
algorithm: find first event to finish. add to solution.
remove conflicting events.
continue.

greedy solution:



algorithm: find first event to finish. add to solution.
remove conflicting events.
continue.

greedy solution:



algorithm: find first event to finish. add to solution.
remove conflicting events.
continue.

running time

algorithm: find first event to finish. add to solution.
remove conflicting events.
continue.

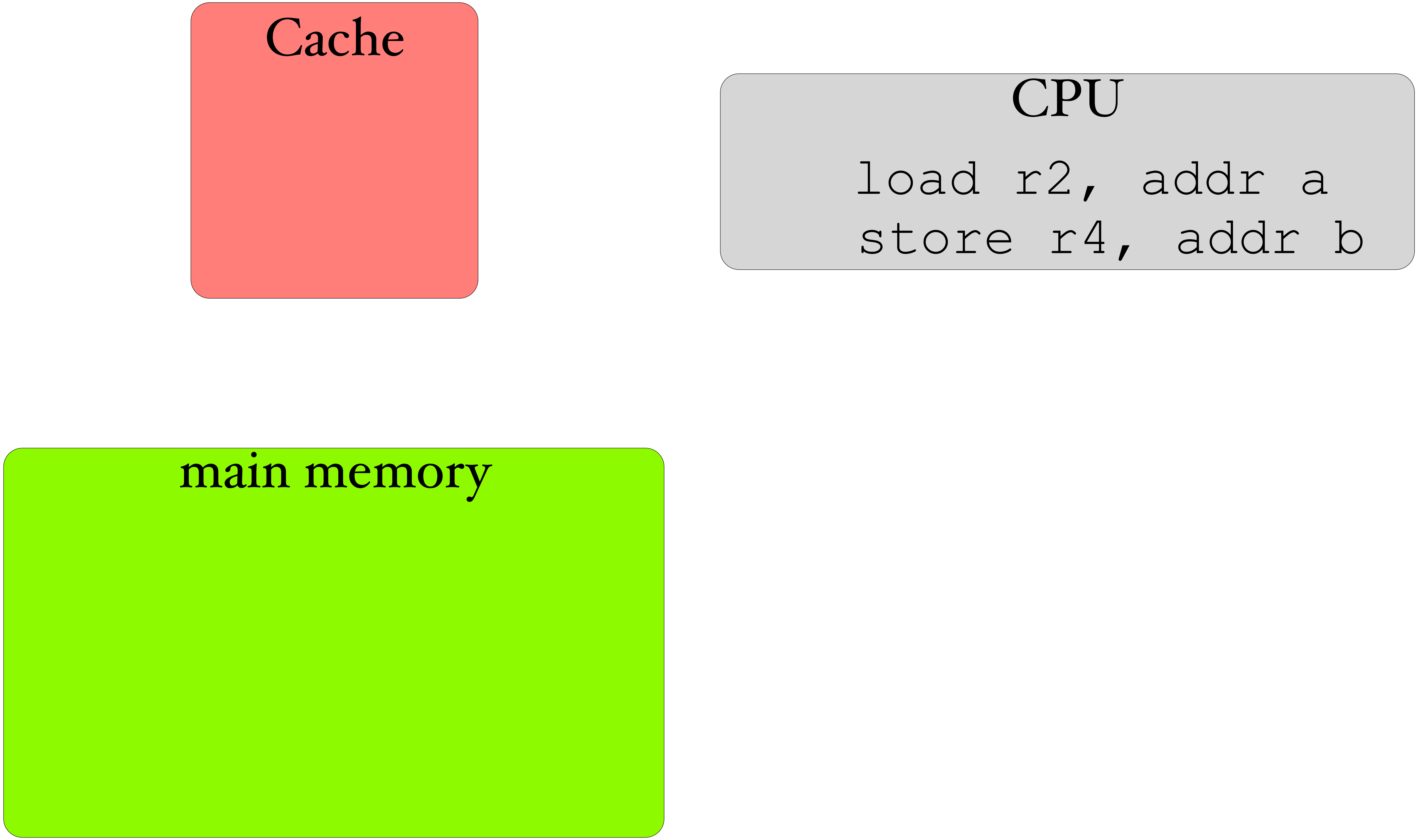
$(f_1, f_2, \ldots, f_n)$ (sorted) $s_i < f_i$

caching

L10

CS4800

cache hit



question:

problem statement

input:

output:

cache is

problem statement

input: K, the size of the cache
$d_1, d_2, \ldots, d_m$ memory accesses

output: schedule for that cache that minimizes # of cache misses while satisfying requests

cache is fully associative, line size is 1

contrast with reality

Belady evict rule

example

cache

a

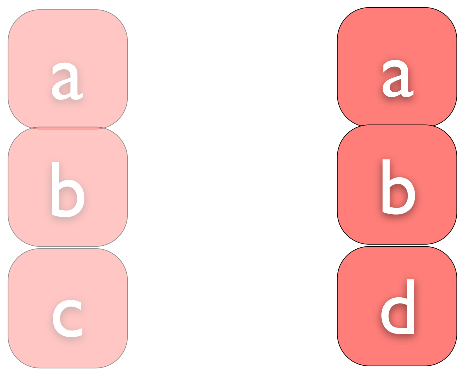
b

c

a b c d a d e a d b a e c e a

example

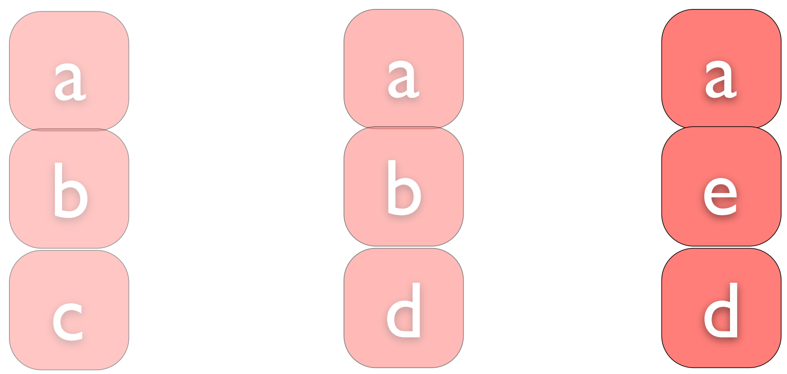
cache



a b c d a d e a d b a e c e a

example

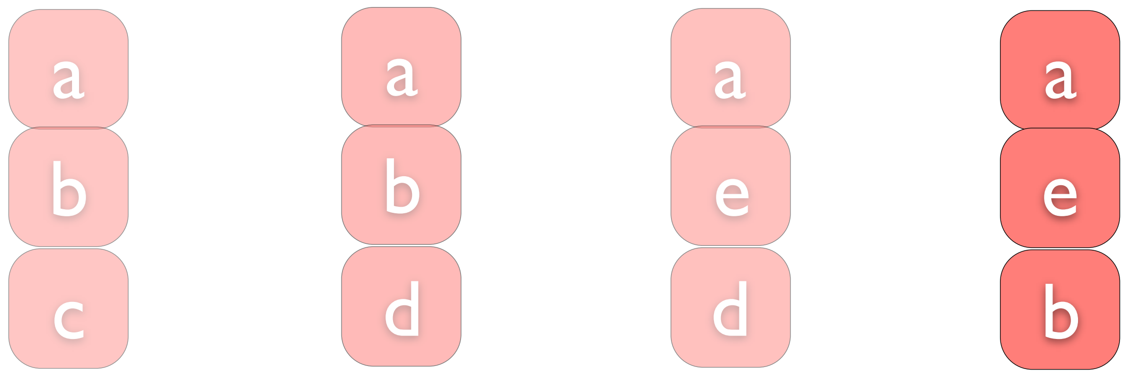
cache



a b c d a d e a d b a e c e a

example

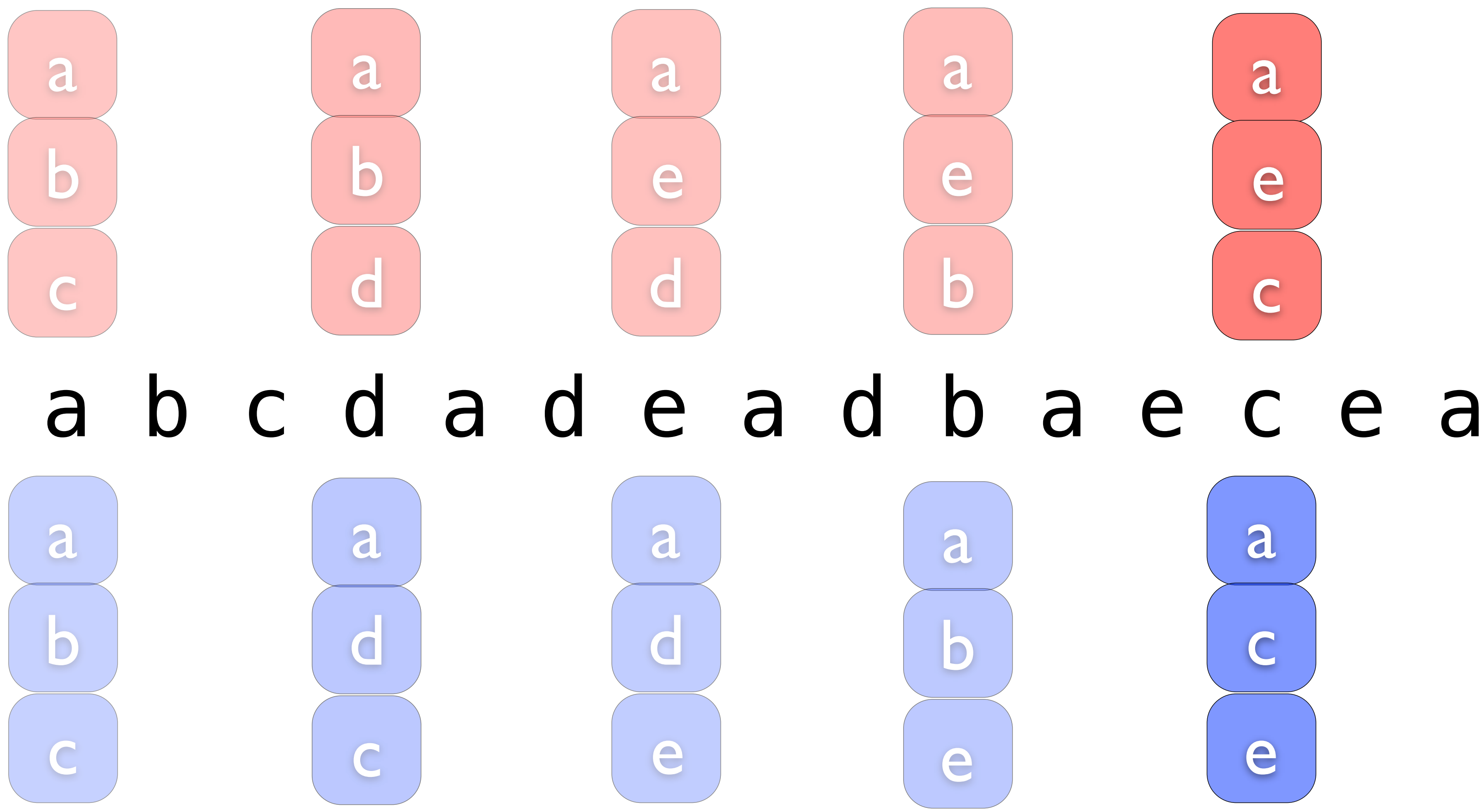
cache



a b c d a d e a d b a e c e a

example

cache



Surprising theorem

schedule

Schedule for access pattern $d_1, d_2, \ldots, d_n$:

Reduced schedule:

Exchange lemma

Exchange Lemma:

Let S be a reduced schedule that agrees with S_{ff} on the first j items. There exists a reduced schedule S’ that agrees with S_{ff} on the first j+1 items and has the same or fewer #misses as S.

S^* S_{ff}

Proof of Lemma

Let S be a reduced sched that agrees with S_{ff} on the first j items.
There exists a reduced sched **S'** that agrees with S_{ff} on the first j+1 items and has the same or fewer #misses as S.

Proof of lemma

State of the cache after J operations under the two schedules.

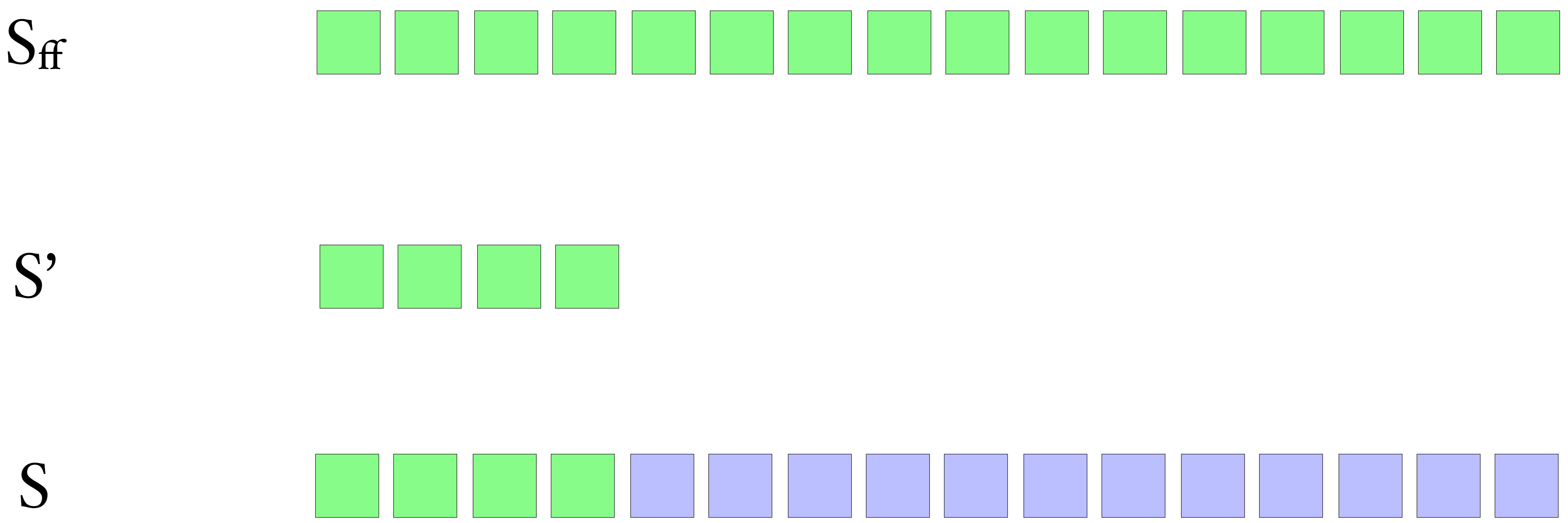
easy case 1

easy case 2

Proof of lemma

case 3

Timeline

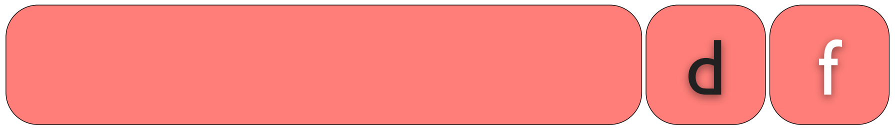


Proof of lemma

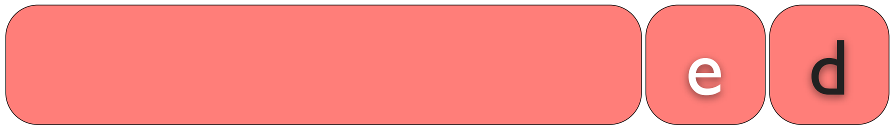
Let access t

Proof of lemma

S



S’



what if t=e ?

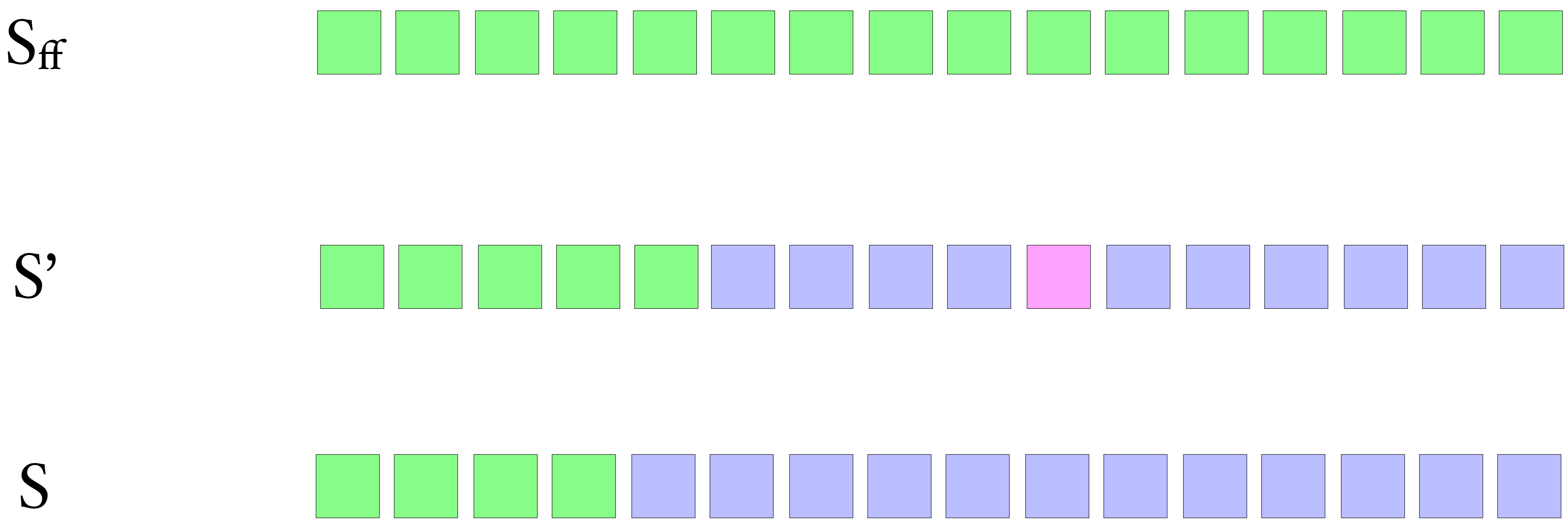
Proof of lemma

what if t=f ?

Proof of lemma

what if t is neither e nor f ?

What have we shown



Let S be a reduced sched that agrees with S_{ff} on the first j items.
There exists a reduced sched **S'** that agrees with S_{ff} on the first j+1 items and has the same or fewer #misses as S.

Let S be a reduced sched that agrees with S_{ff} on the first j items.
There exists a reduced sched **S'** that agrees with S_{ff} on the first j+1 items and has the same or fewer #misses as S.

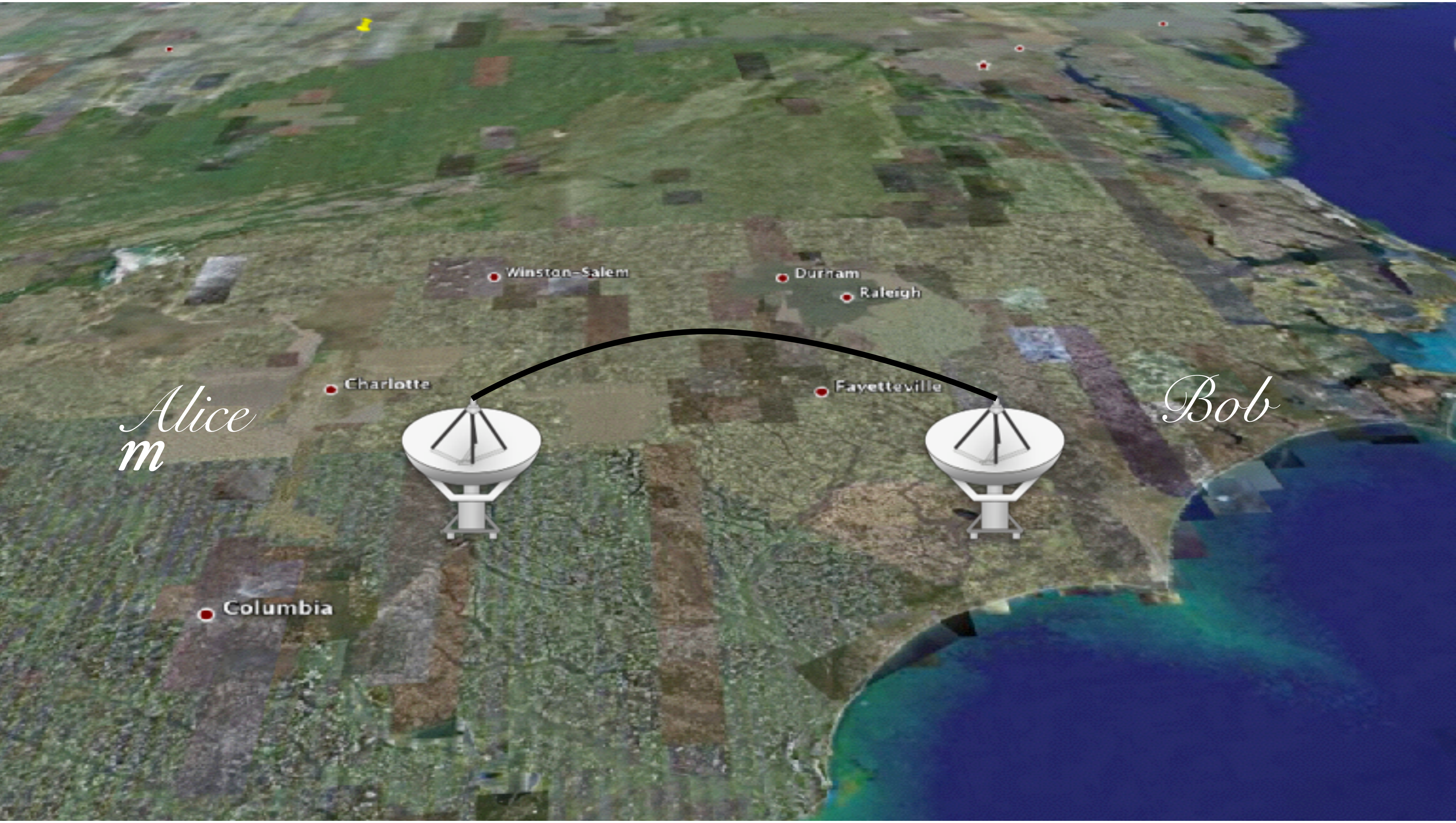
$$S^* \qquad\qquad\qquad\qquad S_{\text{ff}}$$

Huffman

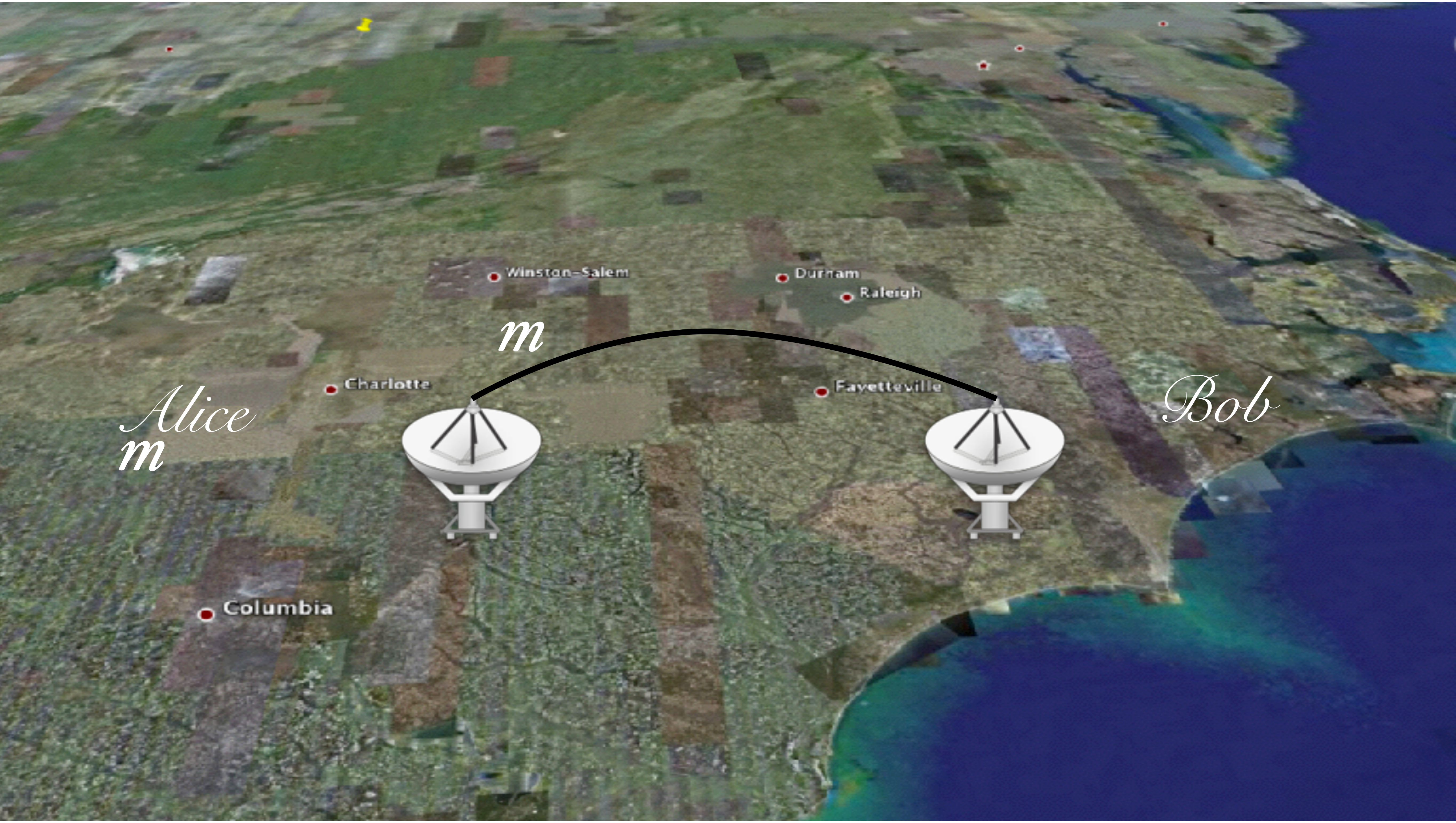
L10

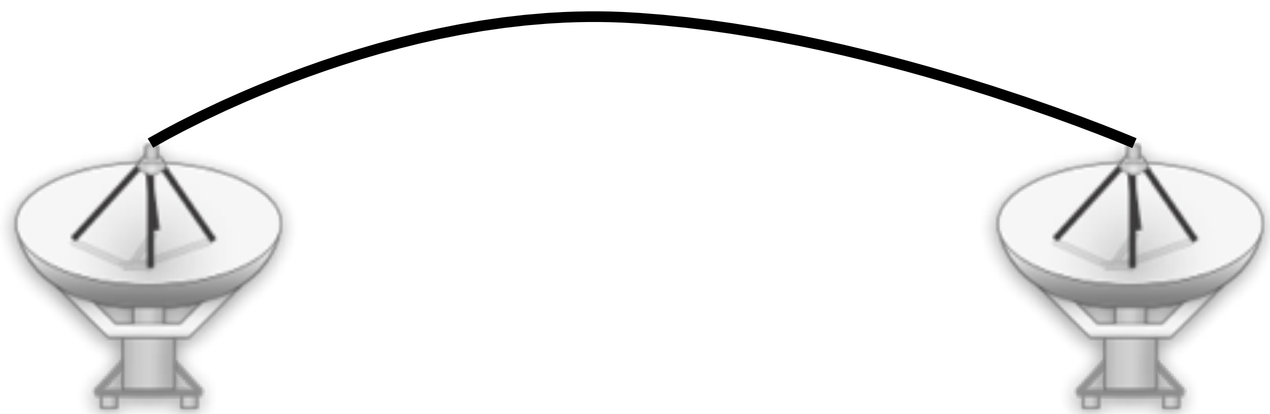
CS4800


image: wikimedia



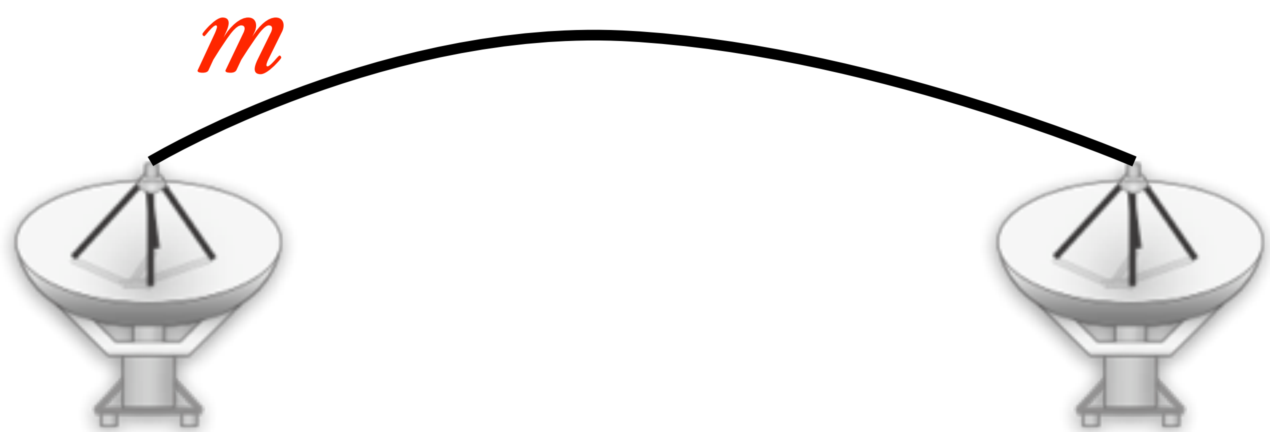
Winston-Salem
Durham
Raleigh
Charlotte
Fayetteville
Columbia
Alice
m
Bob


Winston-Salem
Durham
Raleigh
m
Charlotte
Fayetteville
Alice
m
Bob
Columbia



MOSCOW — President Vladimir V. Putin's typically theatrical order to withdraw the bulk of Russian forces from Syria, a process that the Defense Ministry said it began on Tuesday, seemingly caught Washington, Damascus and everybody in between off guard — just the way the Russian leader likes it.

By all accounts, Mr. Putin delights at creating surprises, reinforcing Russia's newfound image as a sovereign, global heavyweight and keeping him at the center of world events.



MOSCOW — President Vladimir V. Putin's typically theatrical order to withdraw the bulk of Russian forces from Syria, a process that the Defense Ministry said it began on Tuesday, seemingly caught Washington, Damascus and everybody in between off guard — just the way the Russian leader likes it.

By all accounts, Mr. Putin delights at creating surprises, reinforcing Russia's newfound image as a sovereign, global heavyweight and keeping him at the center of world events

$c \in C$ f_c T

e:	235
i:	200
o:	170
u:	87
p:	78
g:	47
b:	40
f:	24

881

$c \in C$	f_c	T	ℓ_c
e:	235	000	3
i:	200	001	3
o:	170	010	3
u:	87	011	3
p:	78	100	3
g:	47	101	3
b:	40	110	3
f:	24	111	3

881

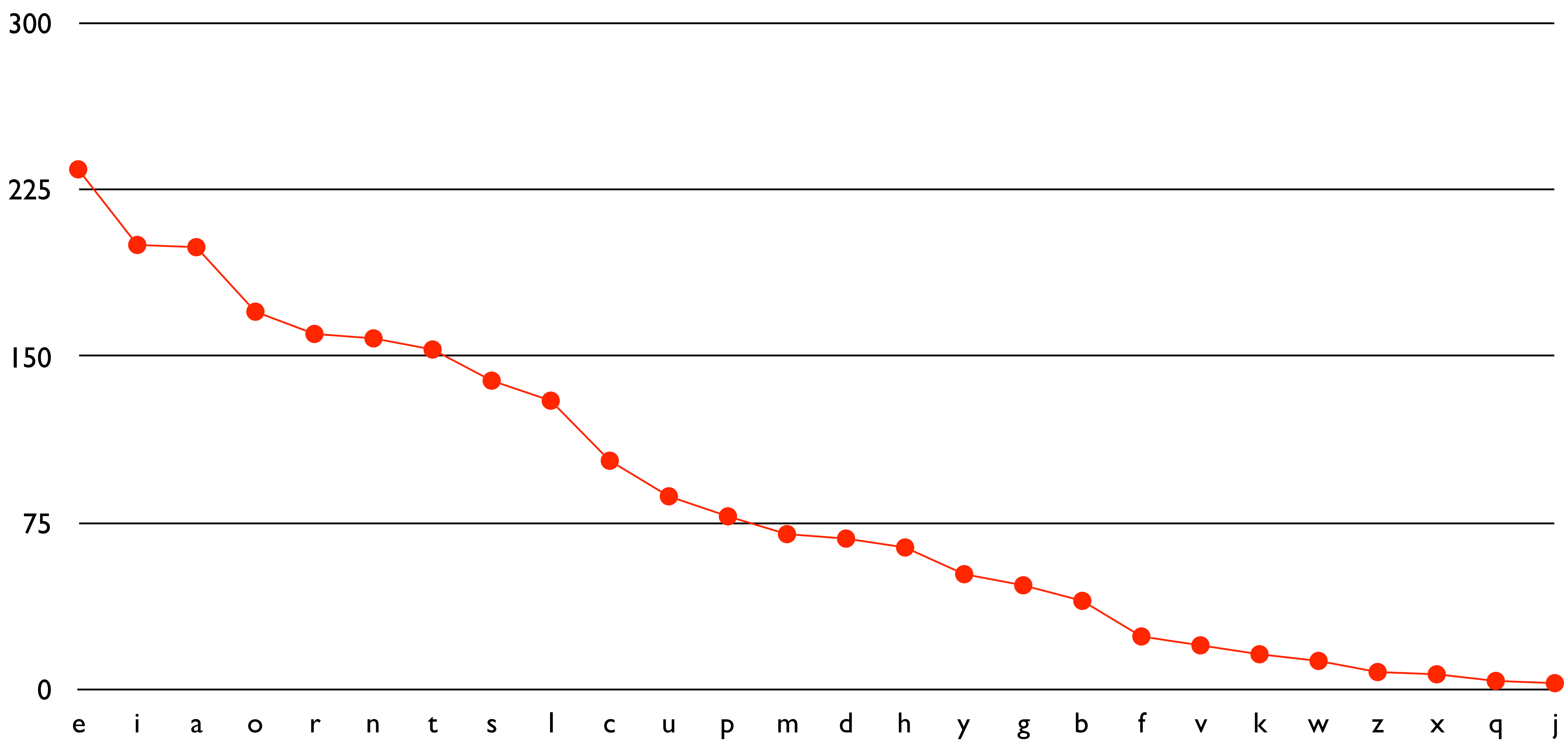
def: cost of an encoding

$$B(T, \{f_c\}) = \sum_{c \in C} f_c \cdot \ell_c$$

$c \in C$	f_c	T	ℓ_c
e:	235	000	3
i:	200	001	3
o:	170	010	3
u:	87	011	3
p:	78	100	3
g:	47	101	3
b:	40	110	3
f:	24	111	3
	881		

character frequency

```
e: 234803
i: 200613
a: 198938
o: 170392
r: 160491
n: 158281
t: 152570
s: 139238
l: 130172
c: 103307
 u: 87211
 p: 78077
 m: 70504
 d: 68007
 h: 64165
 y: 51527
 g: 47011
 b: 40351
 f: 24110
 v: 20103
 k: 16012
 w: 13825
  z: 8439
  x: 6926
  q: 3729
  j: 3075
```

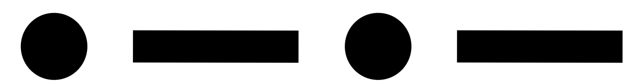


morse code

image http://en.wikipedia.org/wiki/Morse_code

morse code


International Morse Code
- 1 dash = 3 dots.
- The space between parts of the same letter = 1 dot.
- The space between letters = 3 dots.
- The space between words = 7 dots.



def: prefix-free code

def: prefix-free code

$$\forall x, y \in C, x \neq y \implies \textsc{code}(x) \text{ not a prefix of } \textsc{code}(y)$$

def: prefix code

$$\forall x, y \in C, x \neq y \implies \textsc{code}(x) \text{ not a prefix of } \textsc{code}(y)$$

e:	235	0
i:	200	10
o:	170	110
u:	87	1110
p:	78	11110
g:	47	111110
b:	40	1111110
f:	24	11111110

decoding a prefix code

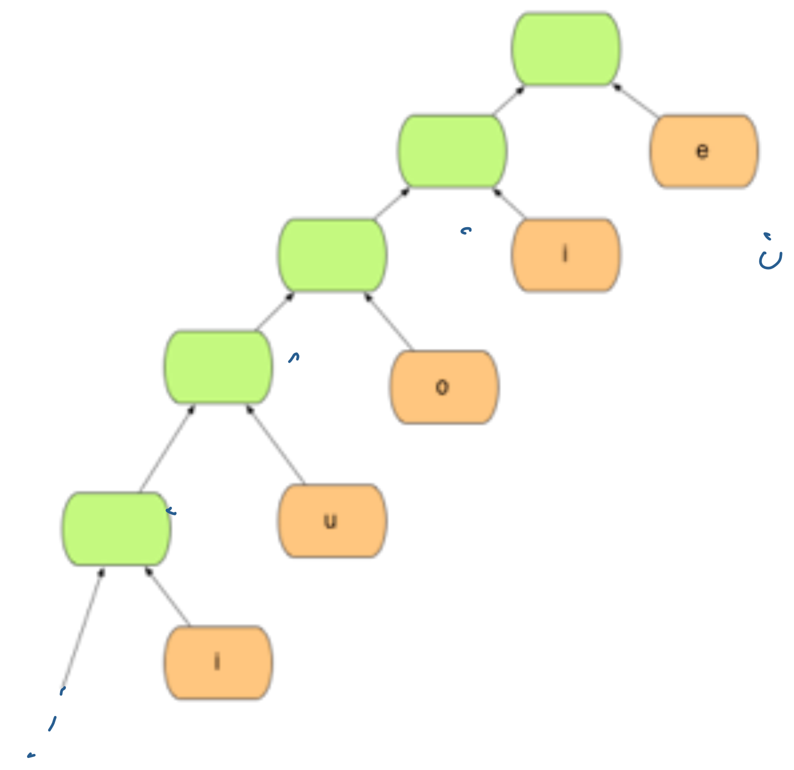
e:	235	0
i:	200	10
o:	170	110
u:	87	1110
p:	78	11110
g:	47	111110
b:	40	1111110
f:	24	11111110

111111010111110

code to binary tree

```
e: 235     0
i: 200     10
o: 170     110
u: 87      1110
p: 78      11110
g: 47      111110
b: 40      1111110
f: 24      11111110
```

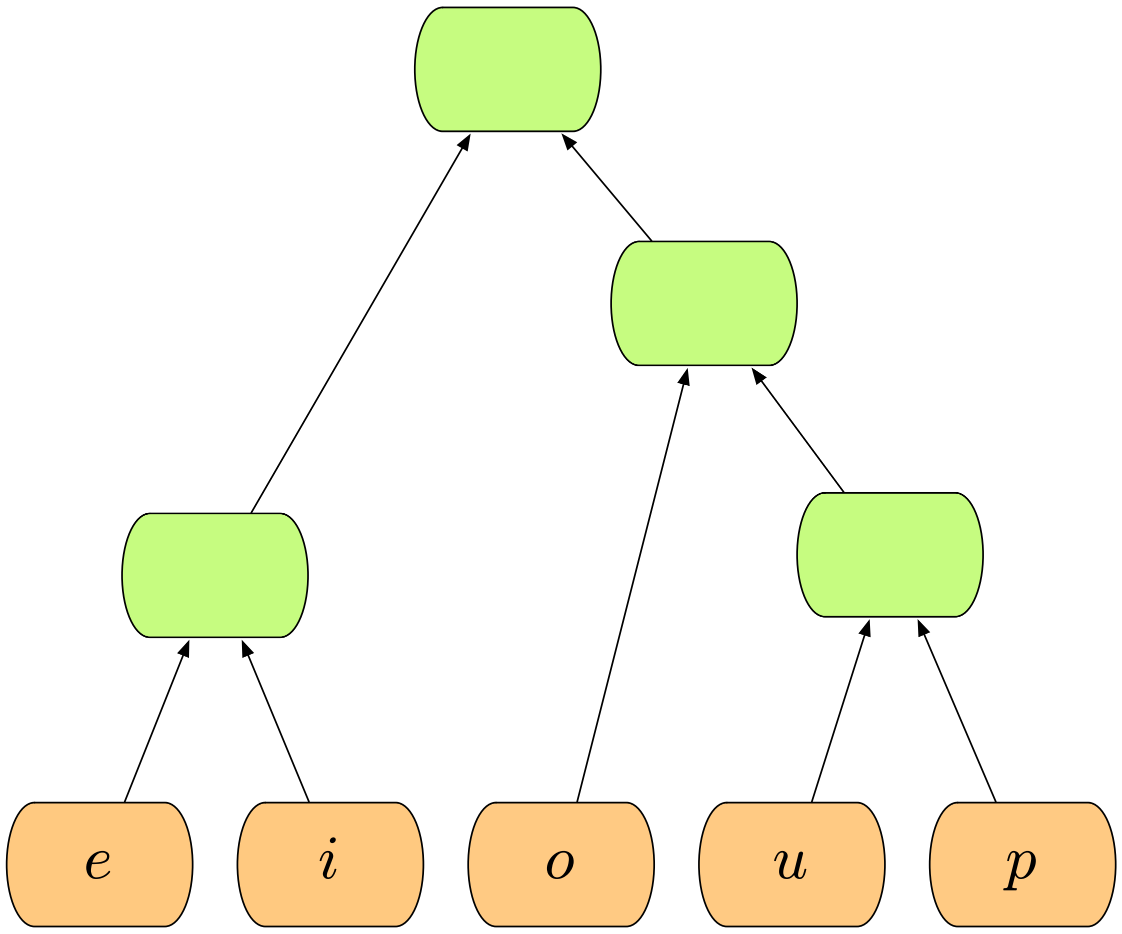
11111101011110



prefix code

binary tree

use tree to encode



$c \in C$	f_c	T	ℓ_c
e:	235	00	2
i:	200	01	2
o:	170	10	2
u:	87	110	3
p:	78	111	3

goal

given the

goal

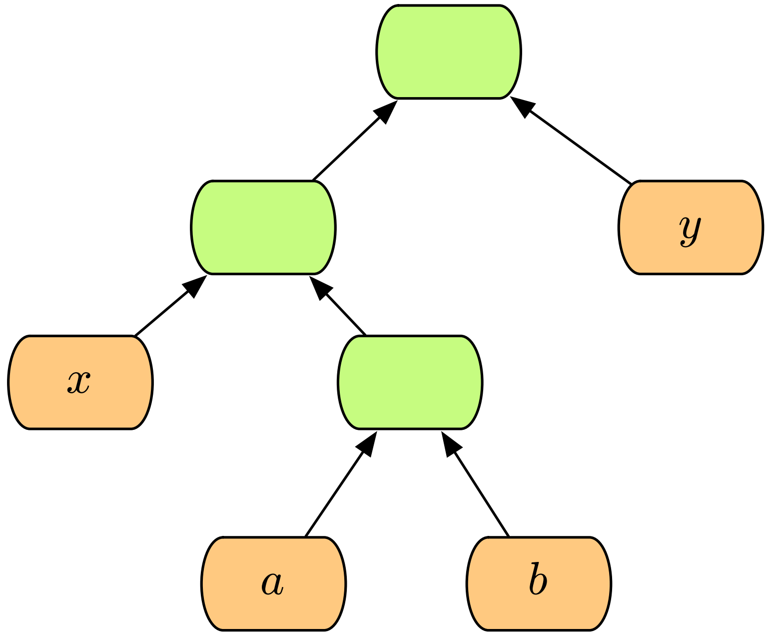
(all frequencies are > 0)

given the character frequencies $\{f_c\}_{c \in C}$

produce a prefix code T with smallest cost

$$\min_T B(T, \{f_c\})$$

property

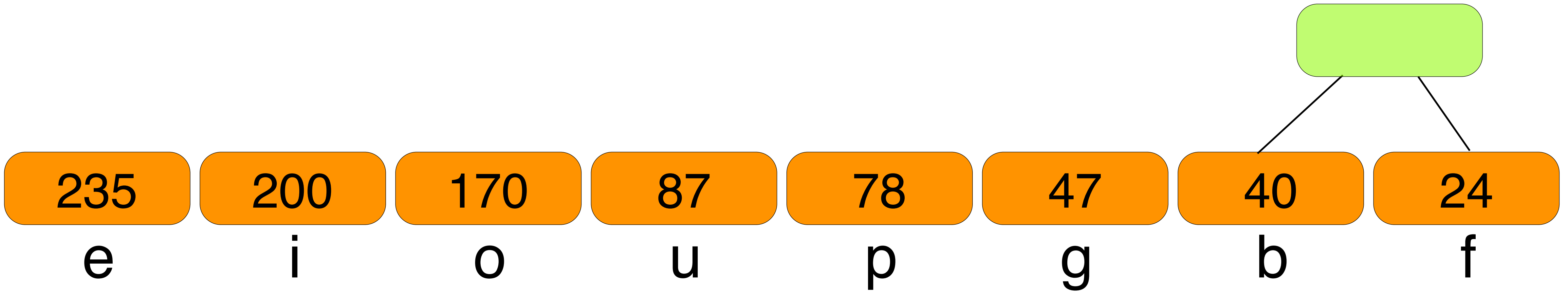


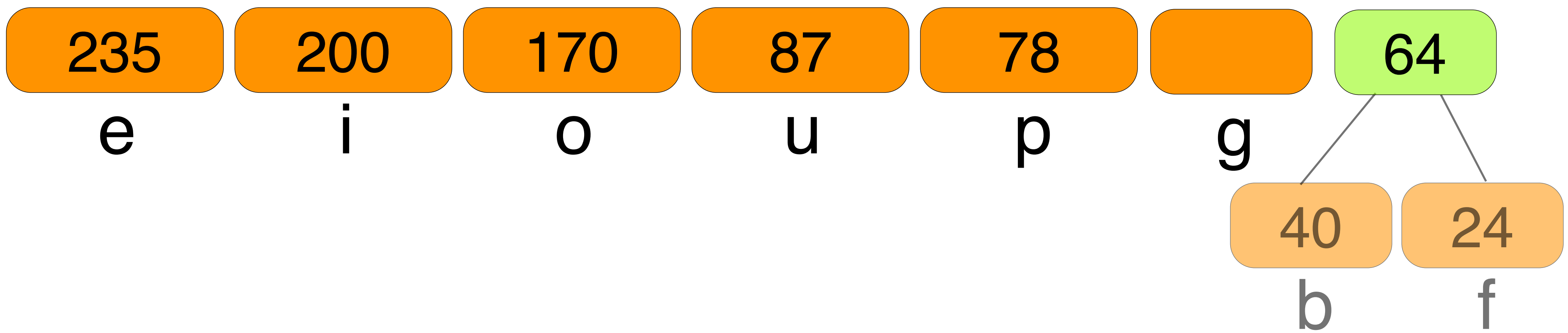
lemma:optimal tree must be full.

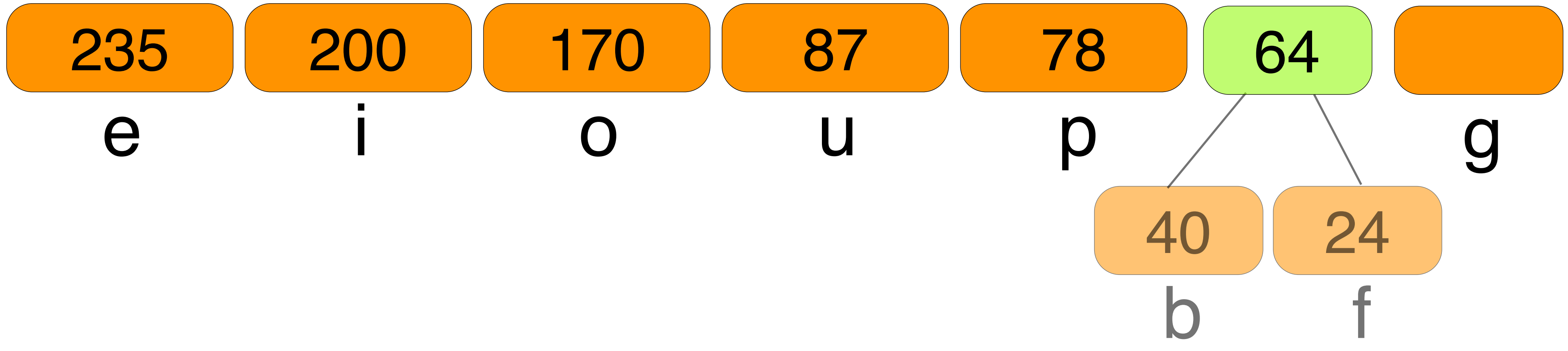
divide & conquer?

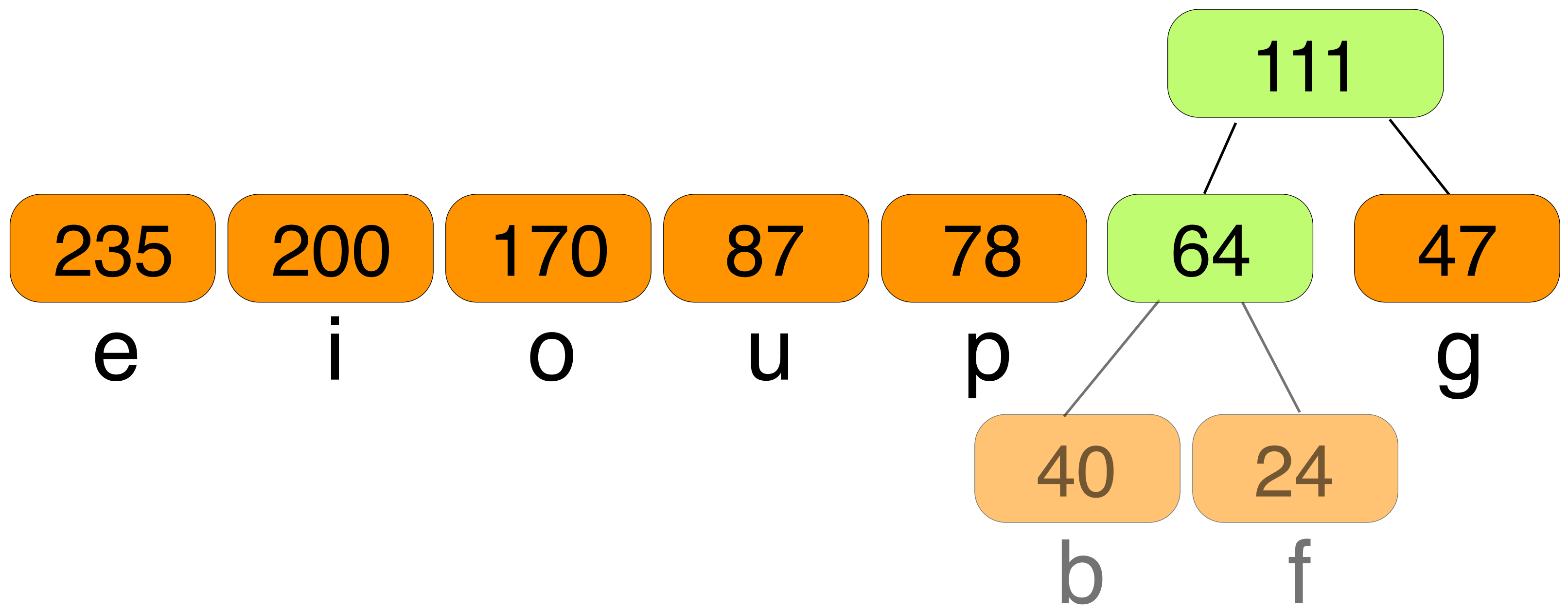
counter-example

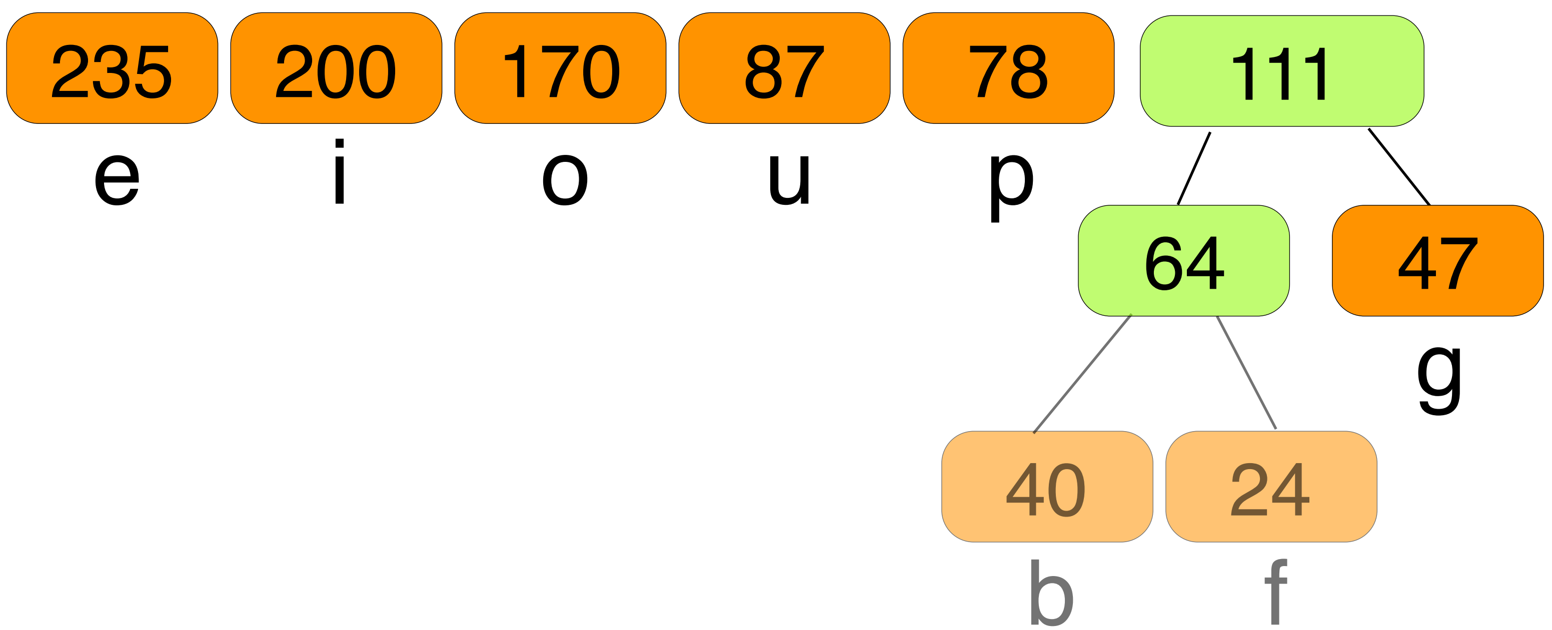
```
e: 32
i: 25
o: 20
u: 18
p: 5
```


235
200
170
87
78
47
40
24
e
i
o
u
p
g
b
f

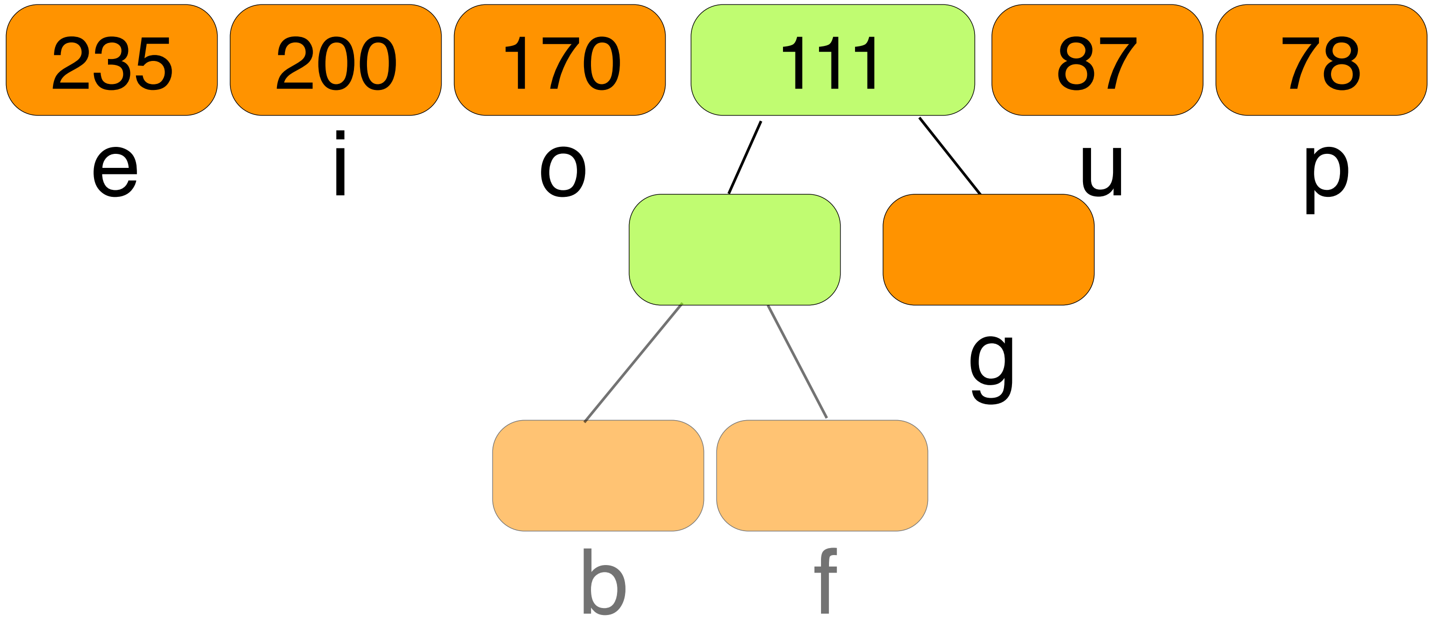

235
200
170
87
78
64
e
i
o
u
p
g
40
24
b
f


235
e
200
i
170
o
87
u
78
p
64
g
40
b
24
f

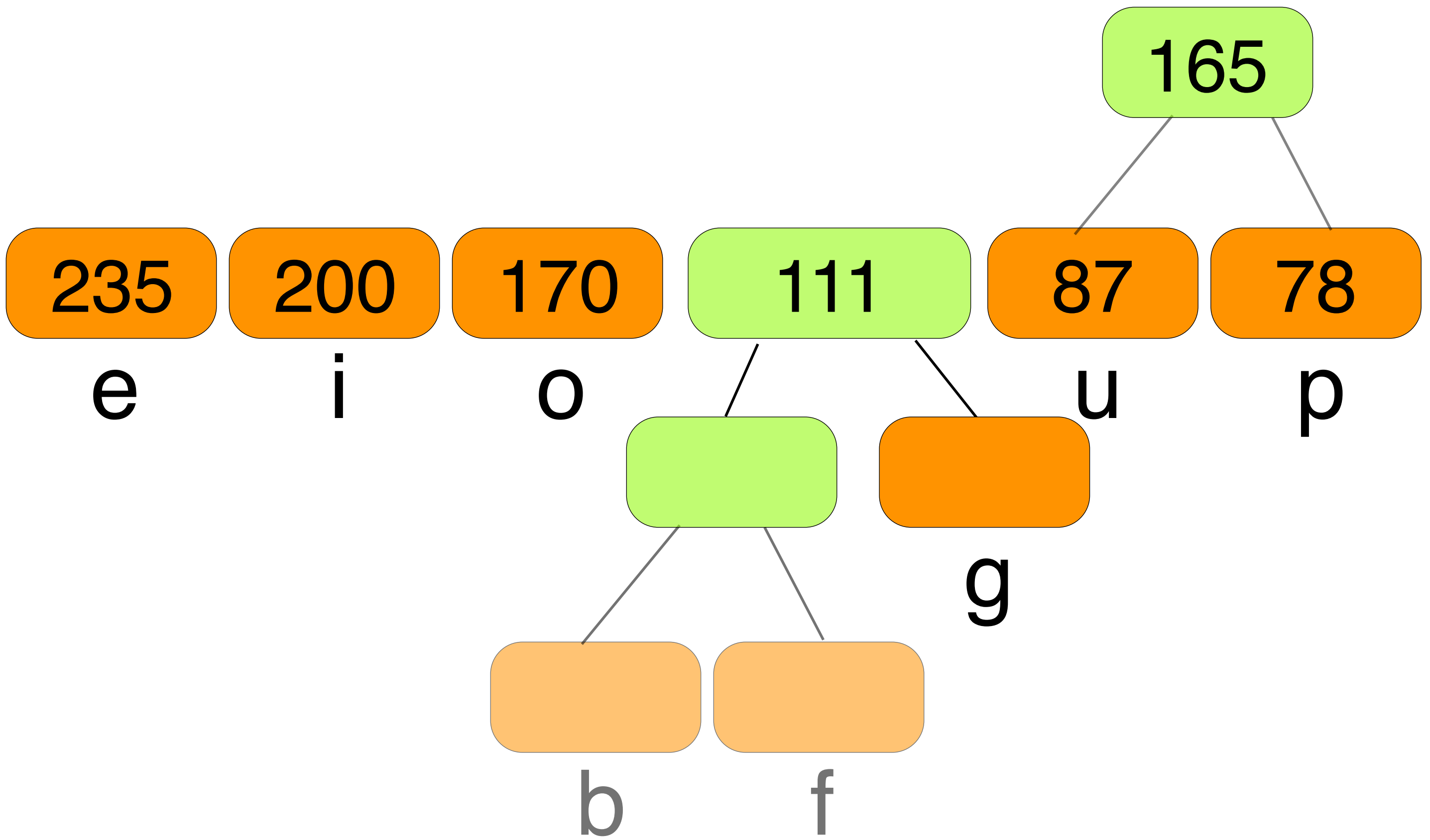

111
235
200
170
87
78
64
47
e
i
o
u
p
g
40
24
b
f

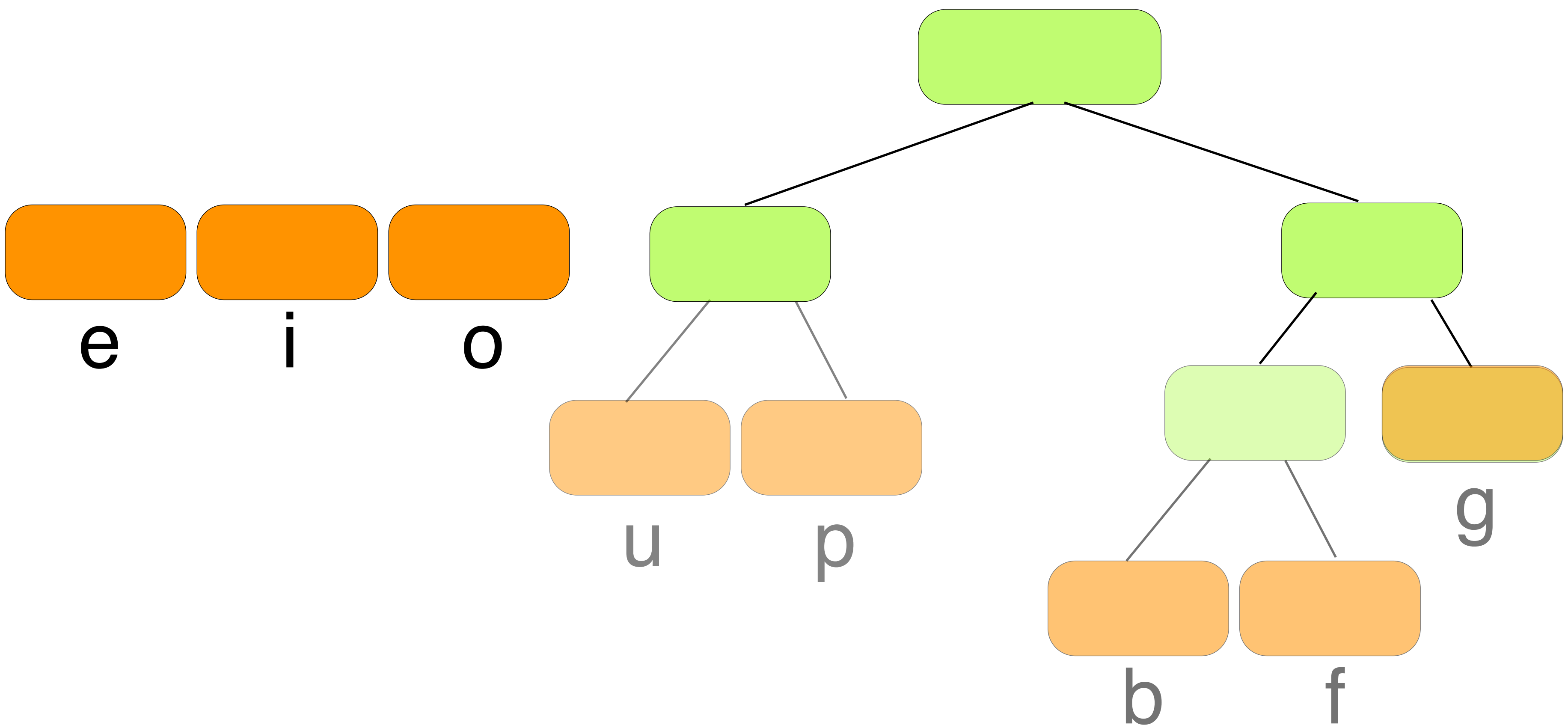


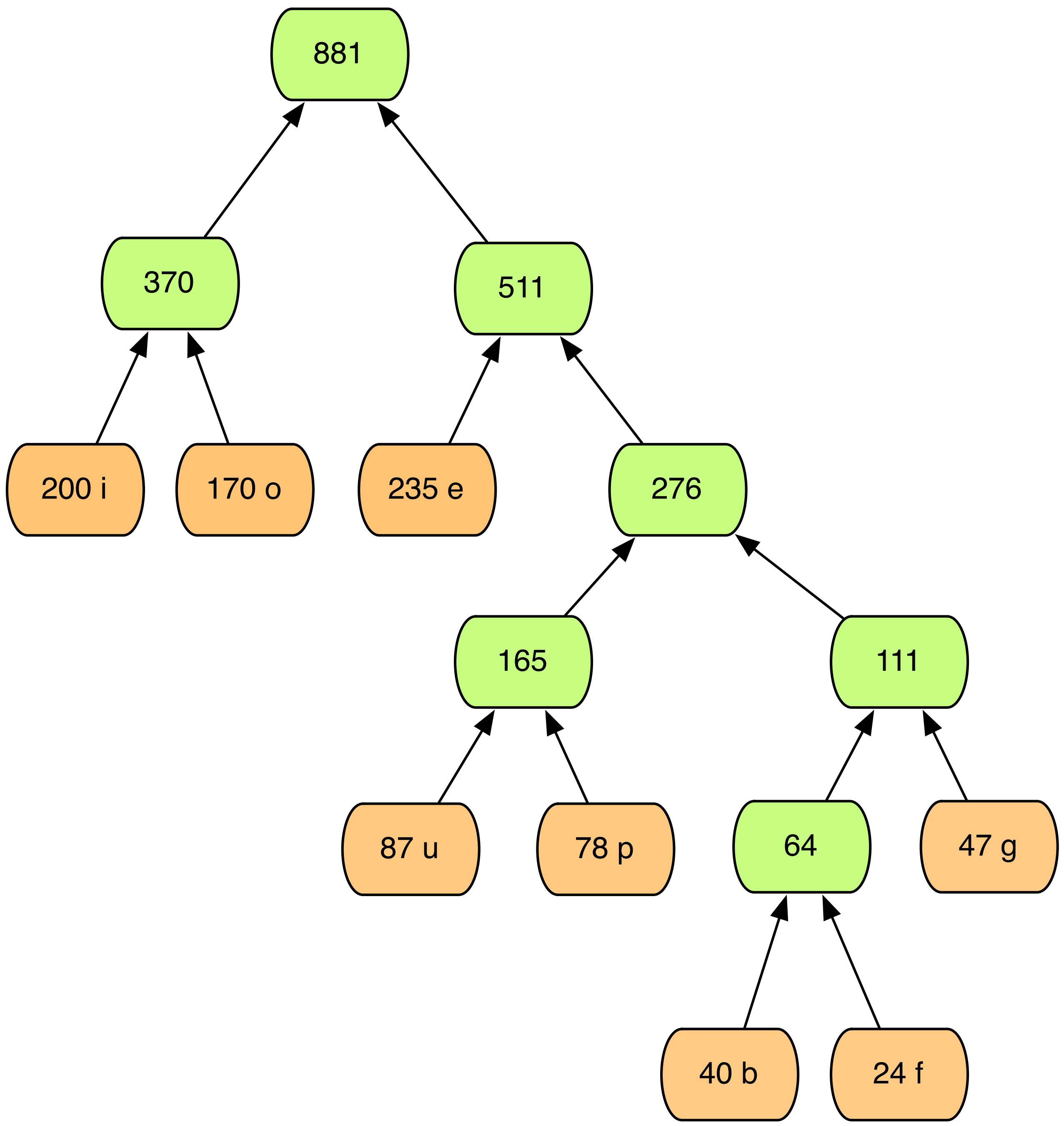
235
e
200
i
170
o
87
u
78
p
111
64
47
g
40
b
24
f

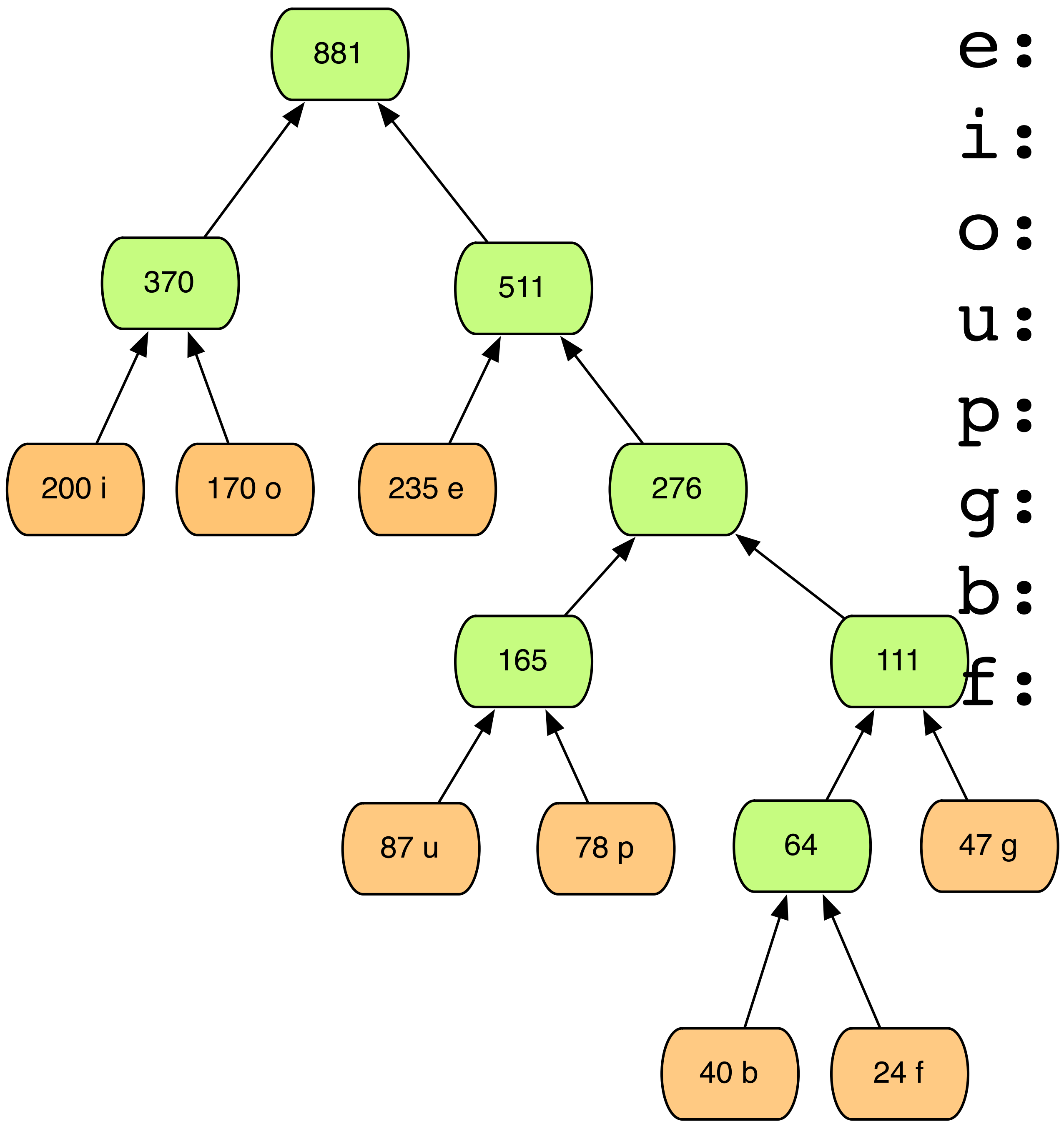


235
200
170
111
87
78
e
i
o
u
p
g
b
f

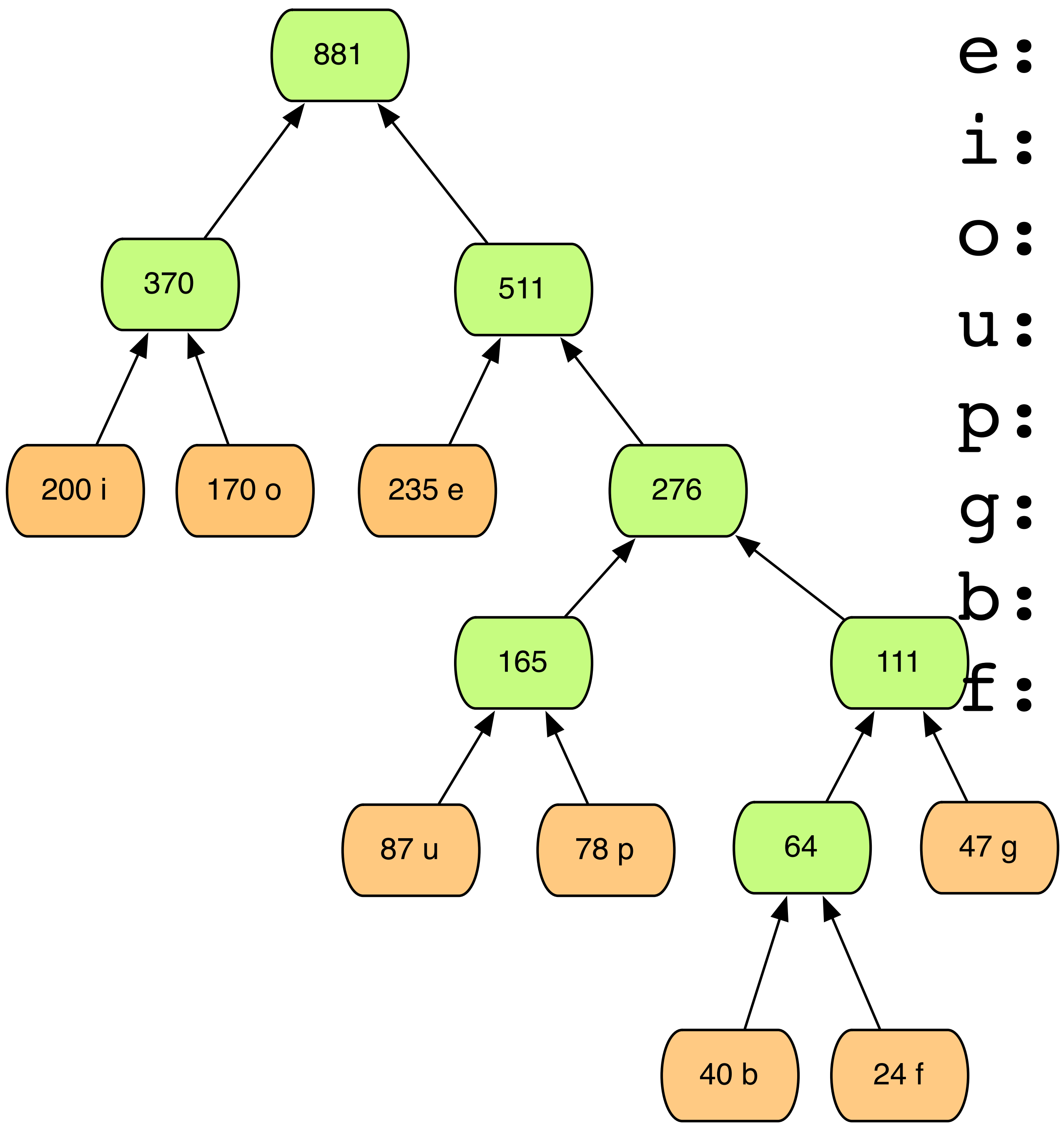

165
235
200
170
111
87
78
e
i
o
u
p
g
b
f


e
i
o
u
p
g
b
f


881
370
511
200 i
170 o
235 e
276
165
111
87 u
78 p
64
47 g
40 b
24 f



e:	235	01
i:	200	11
o:	170	10
u:	87	0011
p:	78	0010
g:	47	0000
b:	40	00011
f:	24	00010



e:	235	01	470
i:	200	11	400
o:	170	10	340
u:	87	0011	348
p:	78	0010	312
g:	47	0000	188
b:	40	00011	200
f:	24	00010	120
			2378

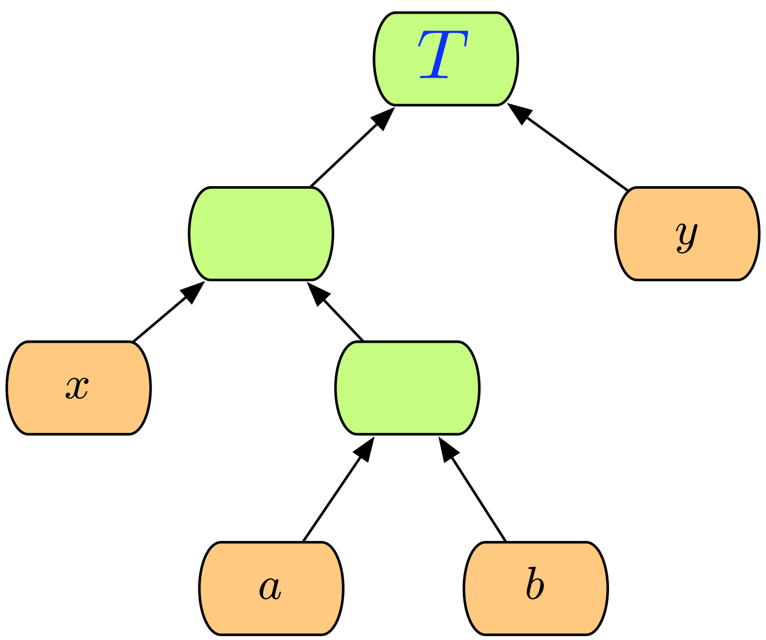
objective

exchange argument

lemma:

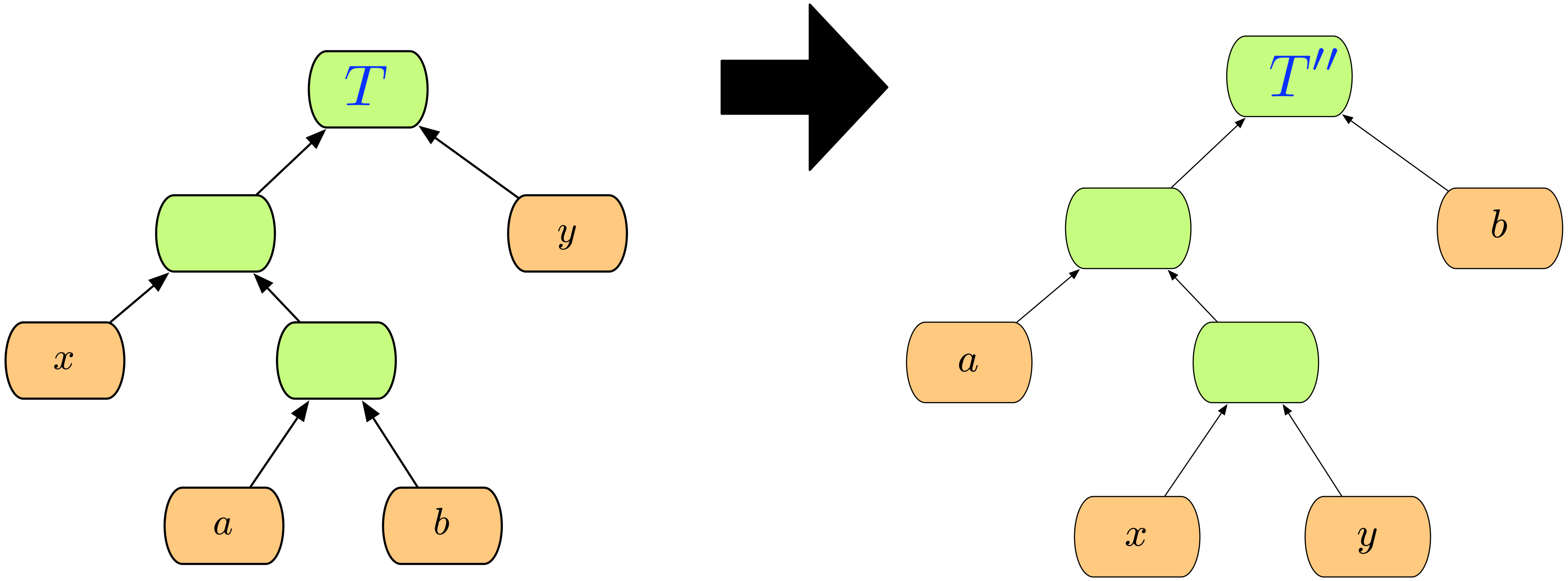
exchange argument

lemma: Let $x, y \in C$ be characters with smallest frequencies f_x, f_y. There exists an optimal prefix code T'' for C in which x, y are siblings. That is, the codes for x, y have the same length and only differ in the last bit.



exchange argument

lemma: Let $x, y \in C$ be characters with smallest frequencies f_x, f_y. There exists an optimal prefix code T'' for C in which x, y are siblings. That is, the codes for x, y have the same length and only differ in the last bit.



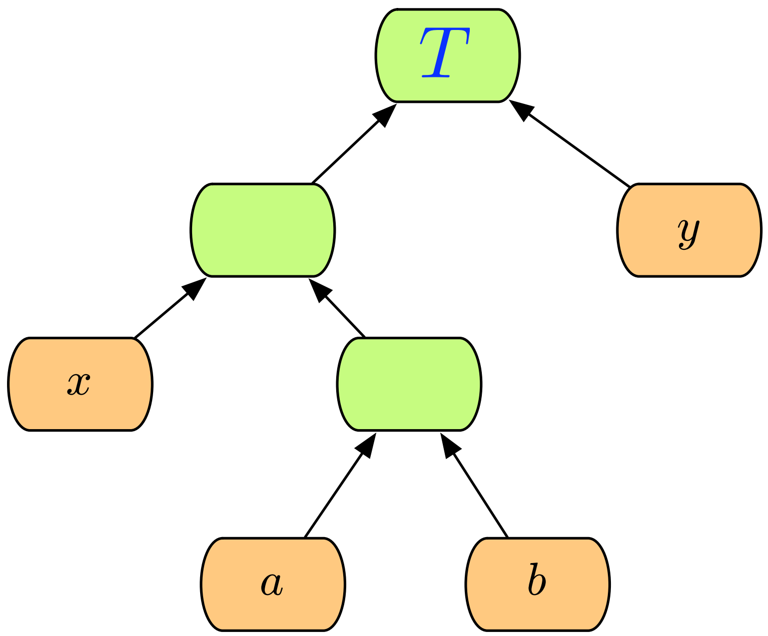
exchange argument

lemma: Let $x, y \in C$ be characters with smallest frequencies f_x, f_y. There exists an optimal prefix code T'' for C in which x, y are siblings. That is, the codes for x, y have the same length and only differ in the last bit.

proof:

exchange argument

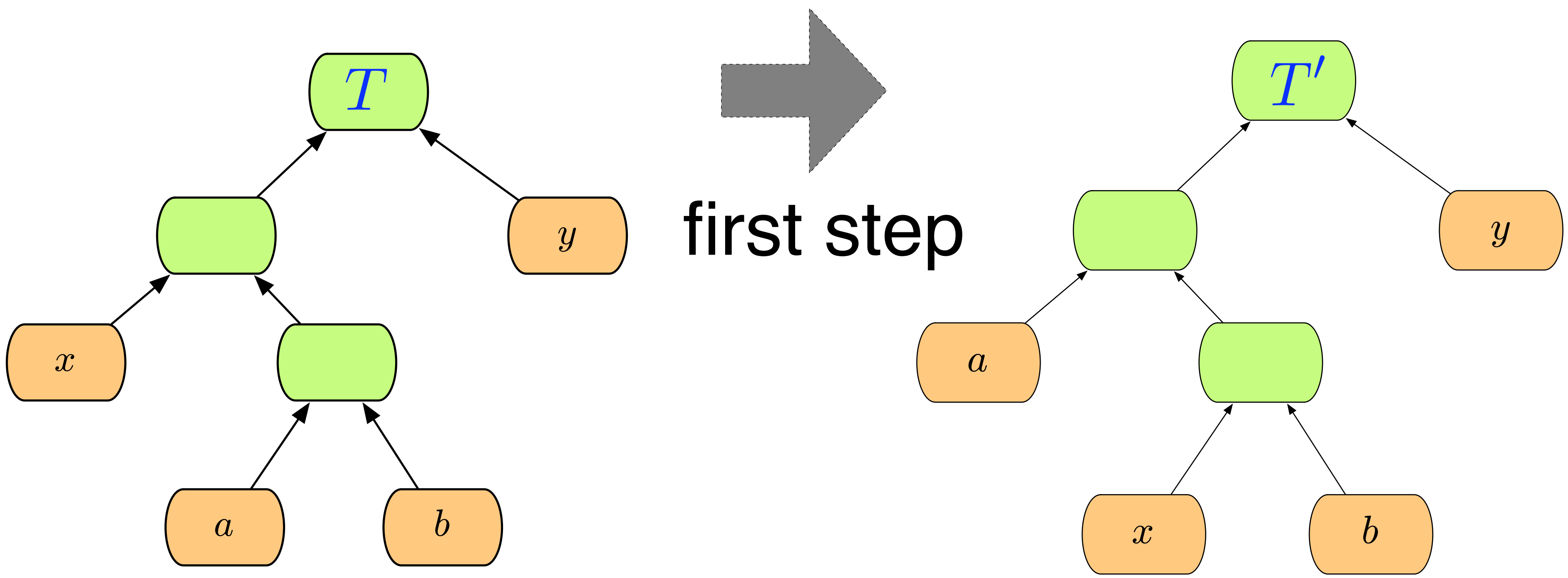
lemma: Let $x, y \in C$ be characters with smallest frequencies f_x, f_y. There exists an optimal prefix code T'' for C in which x, y are siblings. That is, the codes for x, y have the same length and only differ in the last bit.



first step

exchange argument

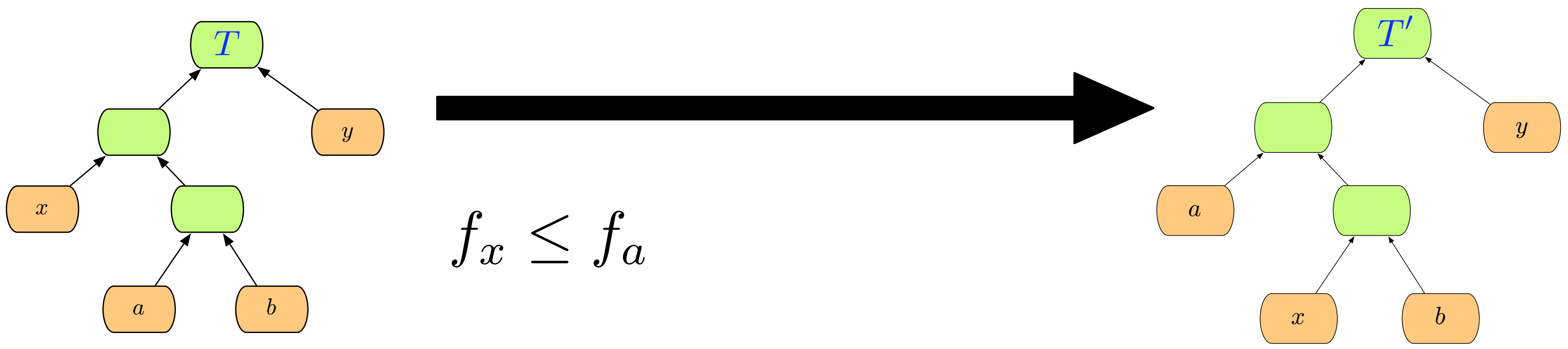
lemma: Let $x, y \in C$ be characters with smallest frequencies f_x, f_y. There exists an optimal prefix code T'' for C in which x, y are siblings. That is, the codes for x, y have the same length and only differ in the last bit.



$f_a \leq f_b$ $\qquad$ $f_x \leq f_a$

$f_x \leq f_y$ $\qquad$ $f_y \leq f_b$

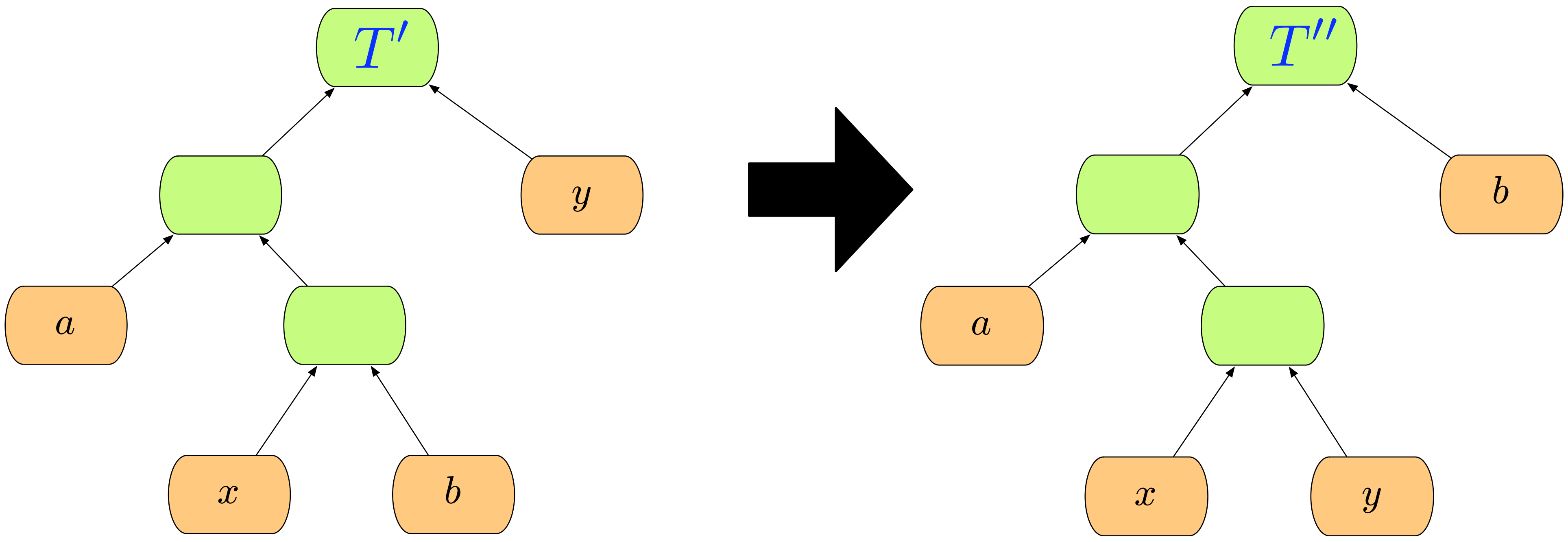

T
x
y
a
b
T′
a
y
x
b



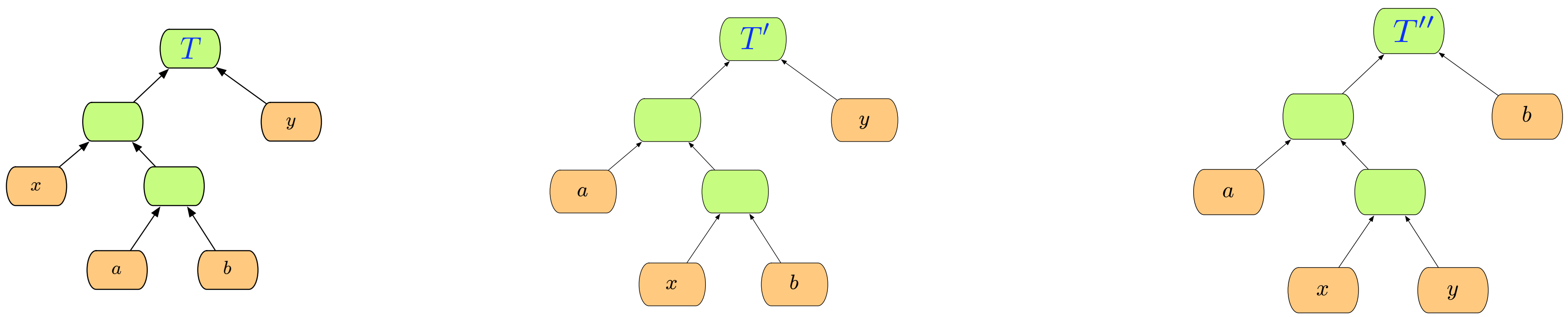
$$B(T) = \sum_c f_c \ell_c + f_x \ell_x + f_a \ell_a \qquad B(T') = \sum_c f_c \ell'_c + f_x \ell'_x + f_a \ell'_a$$

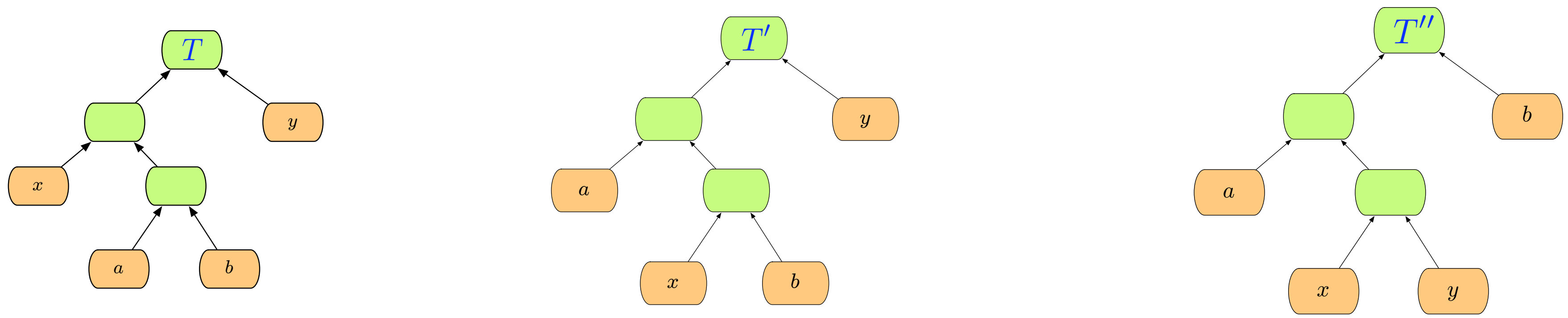
$$B(T) - B(T') \geq 0$$

exchange argument



$$B(T') - B(T'') \geq 0$$


T
x
a
b
y
T'
a
x
b
y
T''
a
x
y
b

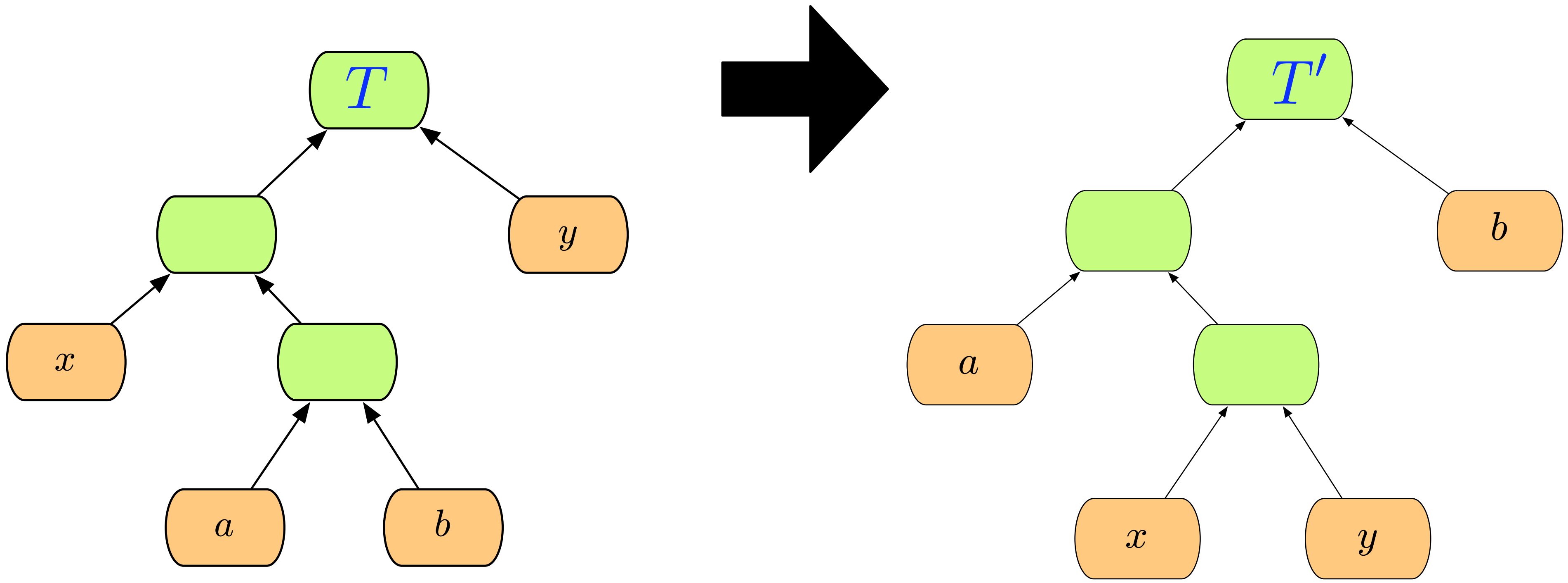


$$B(T) - B(T') \geq 0 \qquad B(T') - B(T'') \geq 0$$

T'' is also optimal

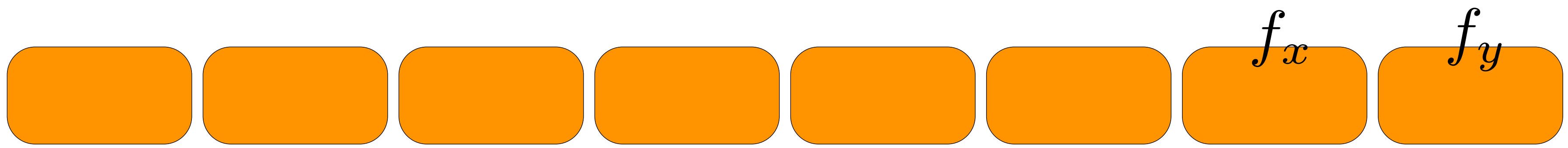
exchange argument

lemma: Let $x, y \in C$ be characters with smallest frequencies f_x, f_y. There exists an optimal prefix code T'' for C in which x, y are siblings. That is, the codes for x, y have the same length and only differ in the last bit.

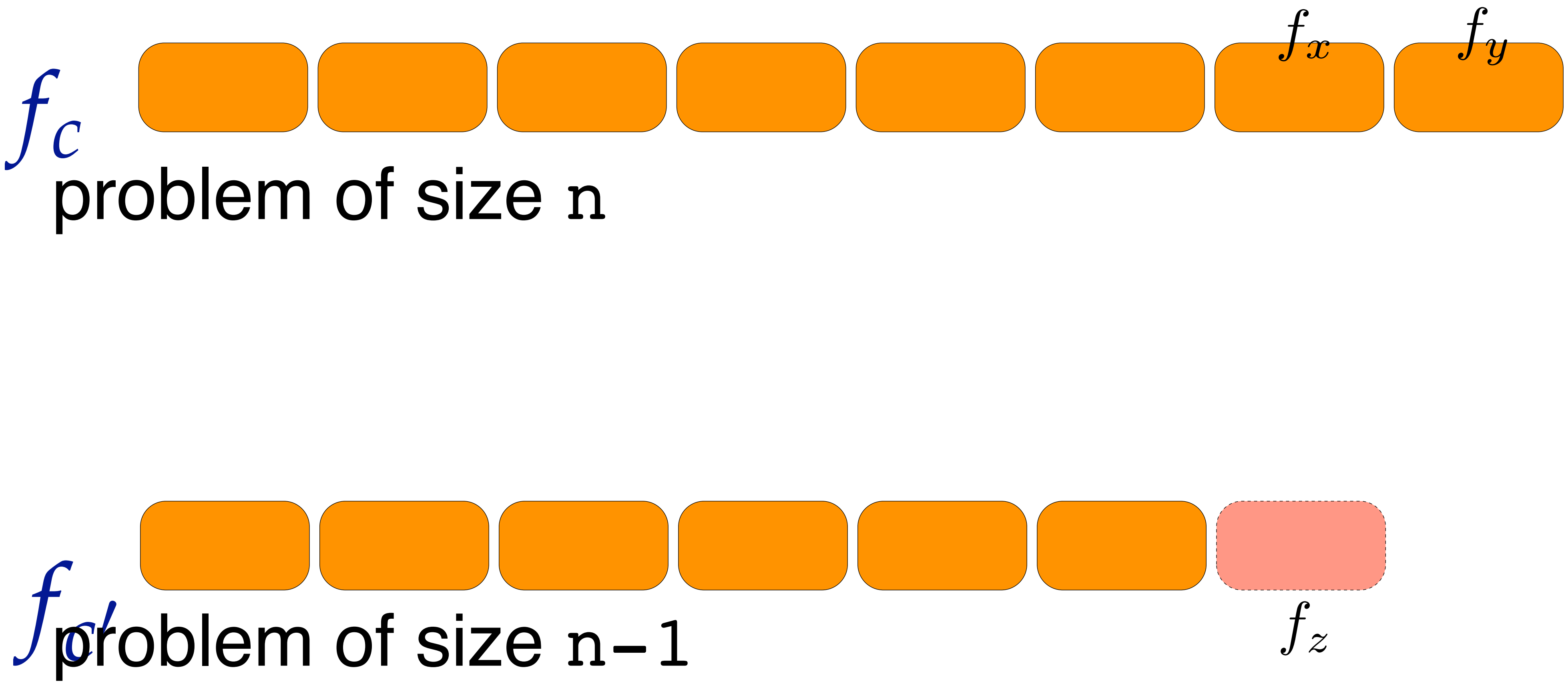


optimal sub-structure

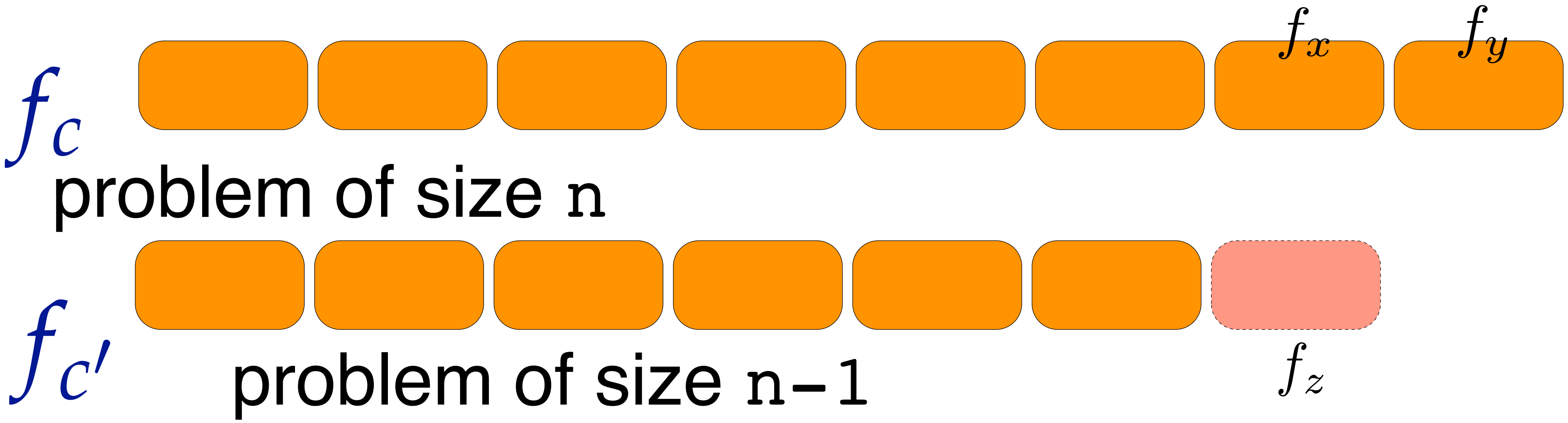
f_c



optimal sub-structure

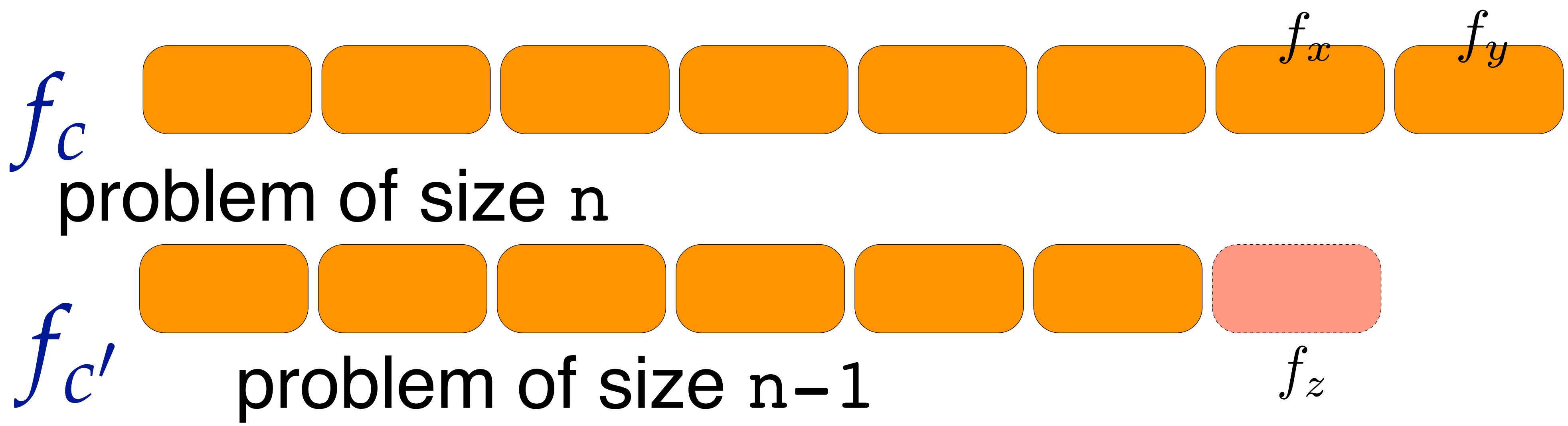


optimal sub-structure

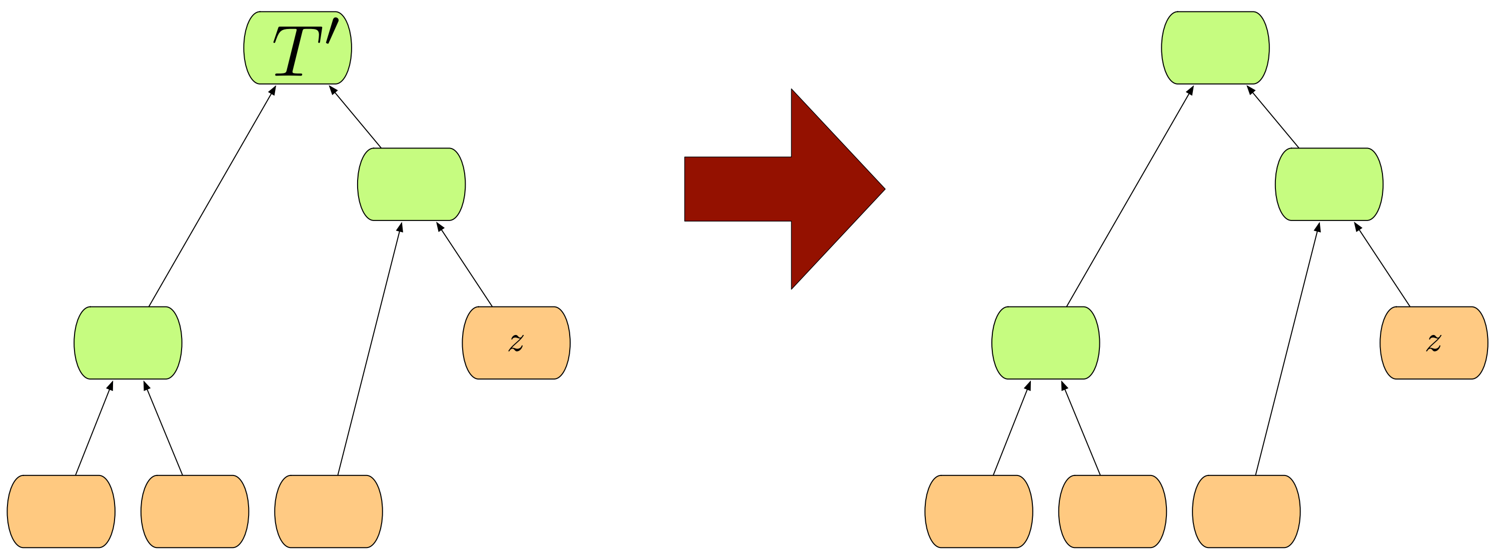


Lemma:

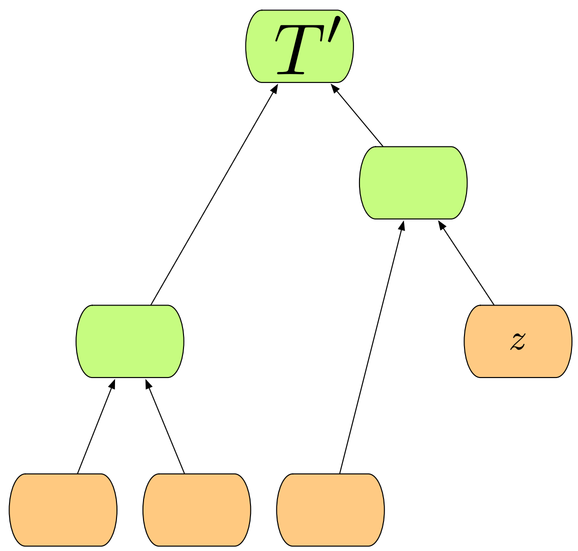
optimal sub-structure



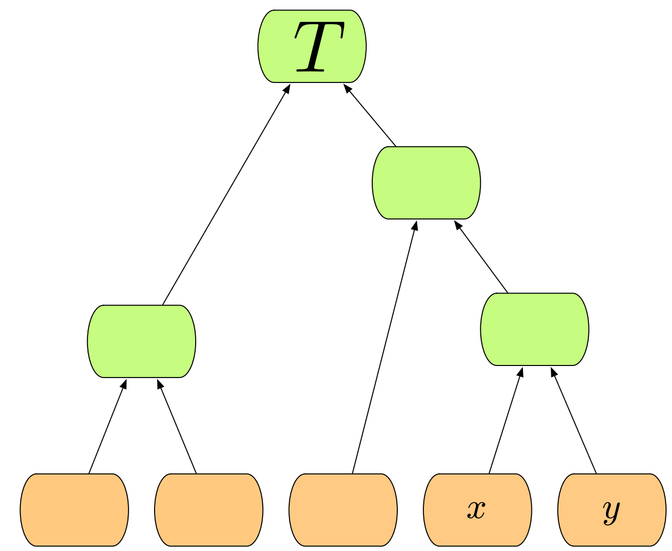
Lemma: The optimal solution for T consists of computing an optimal solution for T' and replacing the left z with a node having children x, y.



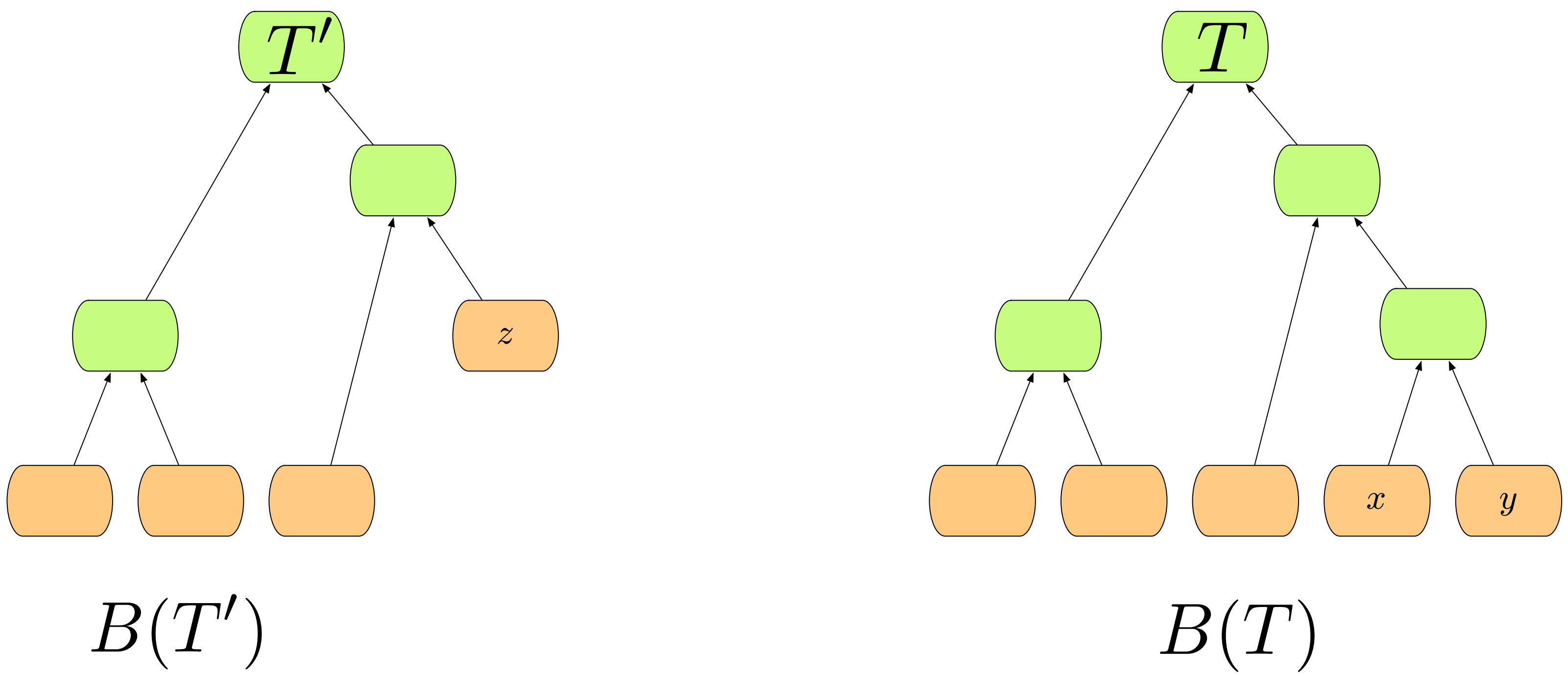
T'
z
z



$B(T')$



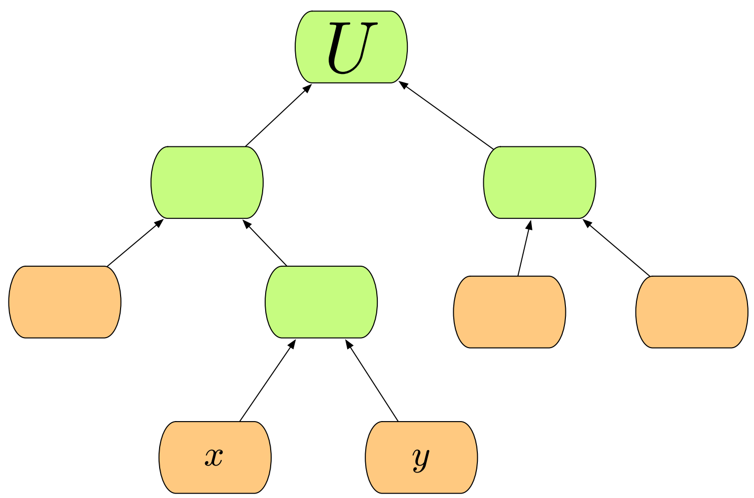
$B(T)$



$$B(T') = B(T) - f_x - f_y$$

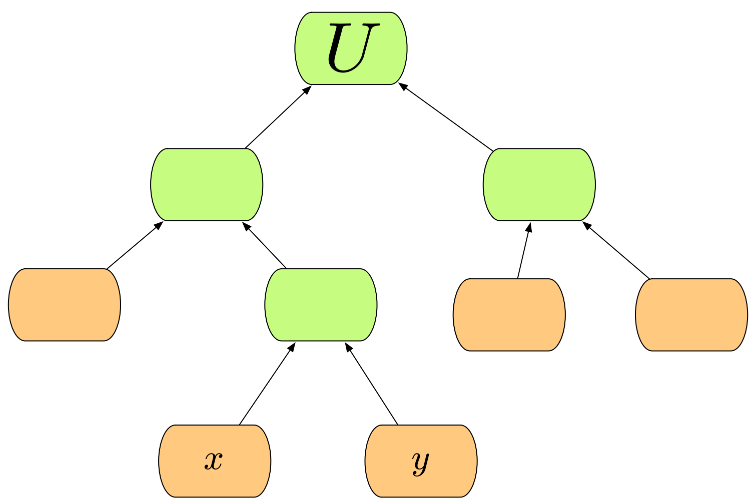
Suppose T is not optimal

Suppose T is not optimal

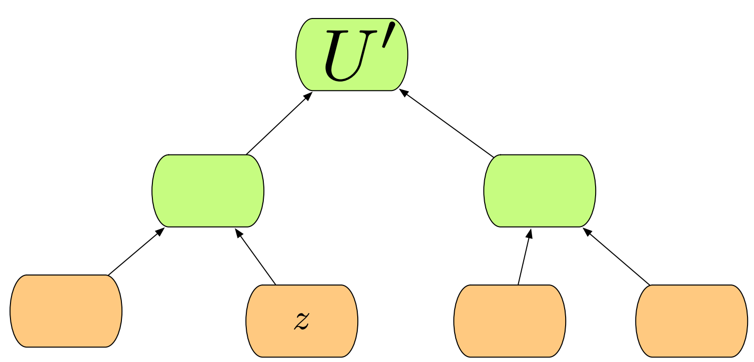


$$B(U) < B(T)$$

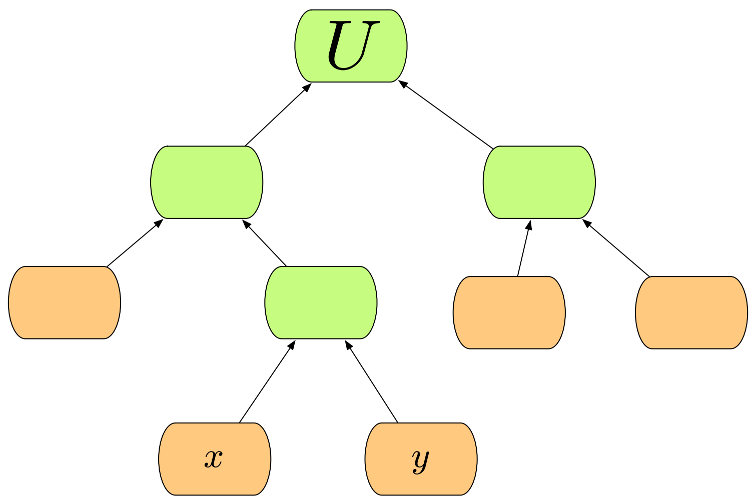
Suppose T is not optimal



$$B(U) < B(T)$$



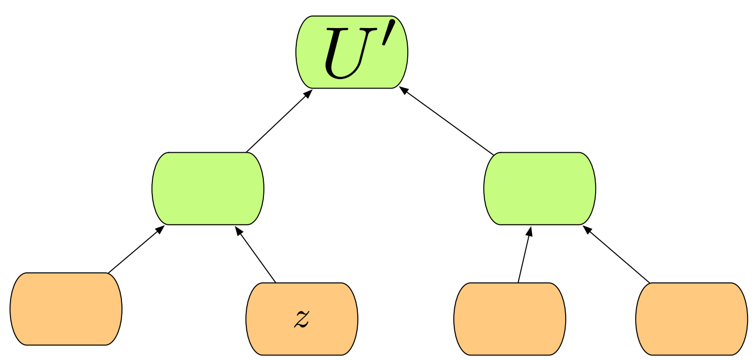
Suppose T is not optimal



$$B(U) < B(T)$$

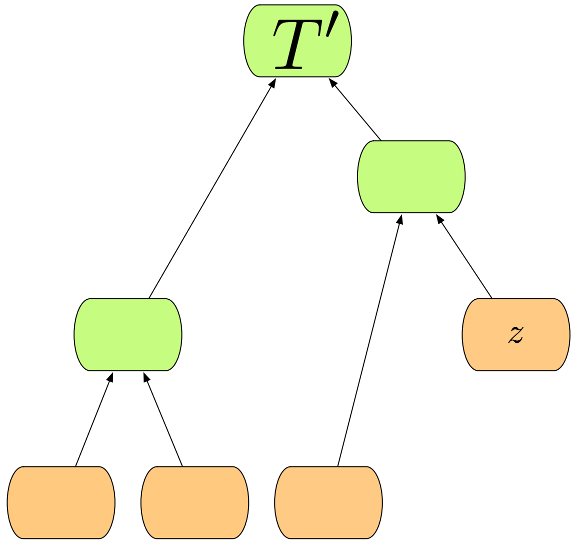
$$B(U') = B(U) - f_x - f_y$$

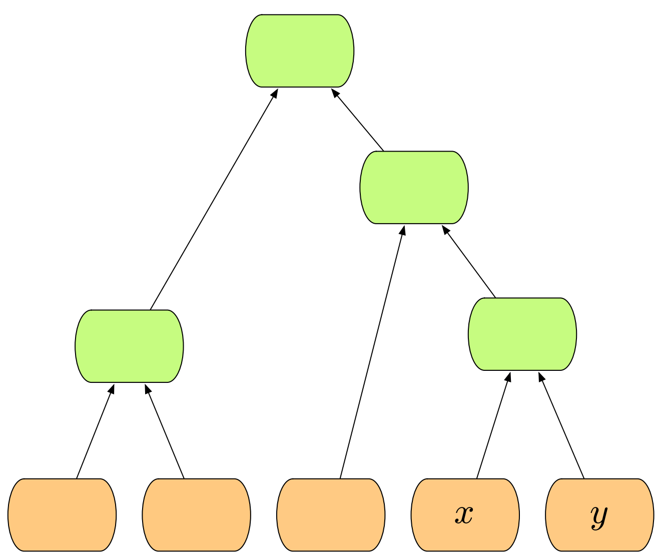
< B(t) - fx - fy



But this implies that B(T') was not o

therefore





summary of argument