CS101: Intro to Computing
Fall 2015

Lecture 21

Administrivia

- Homework 13 due Wednesday
  - Lost? Simulate at 45 degree angle first.
- Midterm 2: November 16th
  - Next Monday!
  - Practice exam coming Wednesday
- Homework 14 released ***after*** the exam

REVIEW

```
x=np.arange(1,4)
np.random.shuffle(x)
```

A

1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

B

1	2
4	3

C

1	2	3

```
x=np.arange(1,4)
np.random.choice(x,2)
```

A

1	3	2

B

2	2

C

3

```
z=13
n=np.array((1,0))
s=np.array((-1,0))
e=np.array((0,1))
w=np.array((0,-1))
d=[n,s,e,w]
x=np.array((z/2,z/2))
x+=d[np.random.randint(0,4)]
```

What does x represent in this code?

a) The current location

b) The direction of travel

c) A step in a random direction

OPTIMIZATION

Optimization

- Given a function f(x), find the x such that f(x) is maximized/minimized
- Goal: search through the domain for the optimum x
- Many clever ways to do this, but let's start with something naïve

Example

On vacation, you found n items of varying weight and value.

Your bag has a weight limit of 50 pounds.

What is the best set of items to take on the flight?

Set up problem

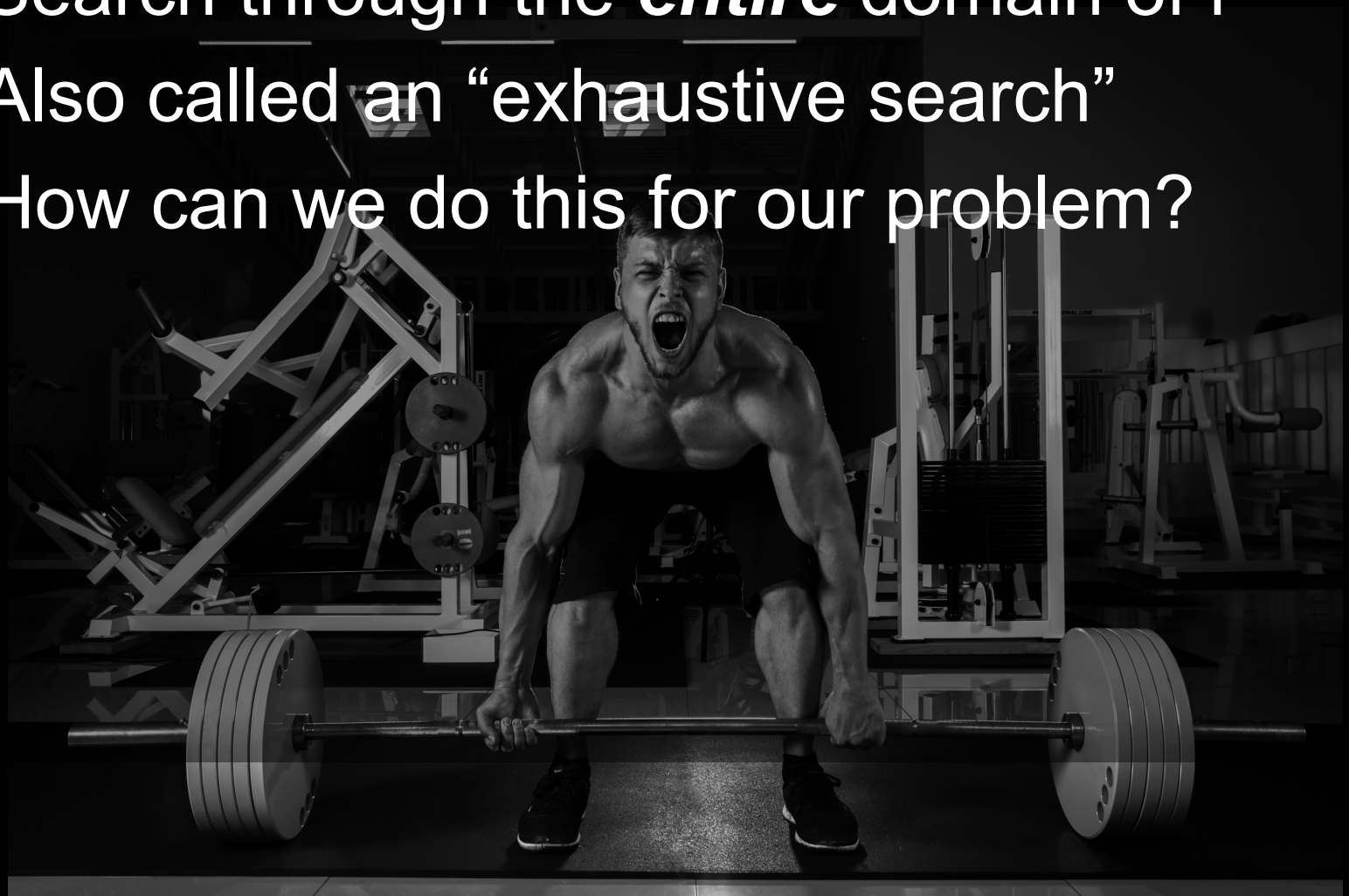
```
import numpy as np
n=10
items=range(n)
weights=np.random.rand(n)*50
values=np.random.rand(n)*100
```

Set up function

```
def f(x,w,v):
        total_weight=0
        total_value=0
        for i in x:
                total_weight+=w[i]
                total_value+=v[i]
        if total_weight>=50:
                return 0
        else:
                return total_value
```

Brute-force search

- Search through the ***entire*** domain of f
- Also called an “exhaustive search”
- How can we do this for our problem?



itertools

- Python library for manipulating iterators
- Loop over iterators “succinctly and efficiently”

```
import itertools
a=“ABCD”
b=“12345”
for x,y in itertools.product(p,q):
  print x,y
```

itertools.product

- Can replace nested for loops

```
import itertools
p=[1,2,3]
q=[5,6]
for a,b in itertools.product(p,q):
   print a+b
```

- Equal to:

```
for a in p:
   for b in q:
      print a+b
```

```
x="ABCD"
y="XYZ"
for a in itertools.product(x,y):
  print ' '.join(a)
```

Which of these are ***not*** printed?

a) "A X"

b) "B D"

c) "C X"

d) "D Z"

itertools.combinations

- Iterate through all subsets of size n
- Subset items are sorted

```
import itertools
a=[1,2,3,4]
for x in itertools.combinations(a,2):
   print x
```

```
x=“ABCD”
for a in itertools.combinations(x,3):
  print ‘ ‘.join(a)
```

Which of these are printed?

a) “A B C”

b) “B D C”

c) “A A D”

d) “A B”