$\frac{TP}{TP+FN}$ **recall** — how many true positives you find

alarm all the time = 100% recall

precision — how many of things you report are real

$\frac{TP}{TP+FP}$

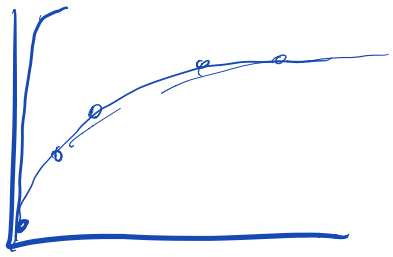
alarm never goes off? 0% recall 100% precision NO FALSE POSITIVES

classifiers

anomaly detector

F1-score

combines recall & precision

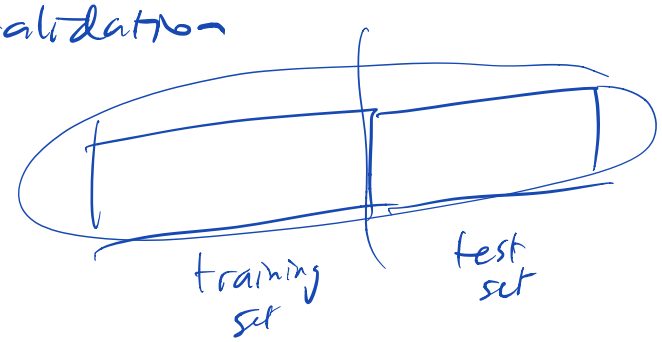


ROC curve

parameter

# of login attempts

Cross-validation



$E[\text{heads}] = 0.5$

HH
HT
TH
TT

HHHHHHTTT

HyperLogLog

approx counter

8 16

$\frac{1}{8}\ \frac{1}{8}\ \frac{1}{8}\ \frac{1}{8}\ \frac{1}{8}\ \frac{1}{8}\ \frac{1}{8}\ \frac{1}{8}$

Bloom Filters

approx set membership

if $(x \notin S)$ NO

if $(x \stackrel{?}{\in} S)$ MAYBE

1	0	1	1

0 0 0 1 MAYBE

0 1 1 1 NO

set cardinality questions

- how many users have Spotify Premium?
- how many people saw post/tweet?
- how many diff. words in El Cid?
- how many users are from Spain & non-Android?

1,000,000
1,000,001

One-pass:

```
words = ["En", "el", "tiempo", "el", "hombre"...]

dict = {}
count = 0
for w in words:
    if not w in dict:
        dict[w] = True
        count += 1
```

Two pass

```
s = sort(words)

curr = s[0]
ptr = 0
while (
    while curr = s[ptr]
        ptr += 1
    count += 1
```

["el", "el", "hombre", "hombre", "tiempo"...]

1 2

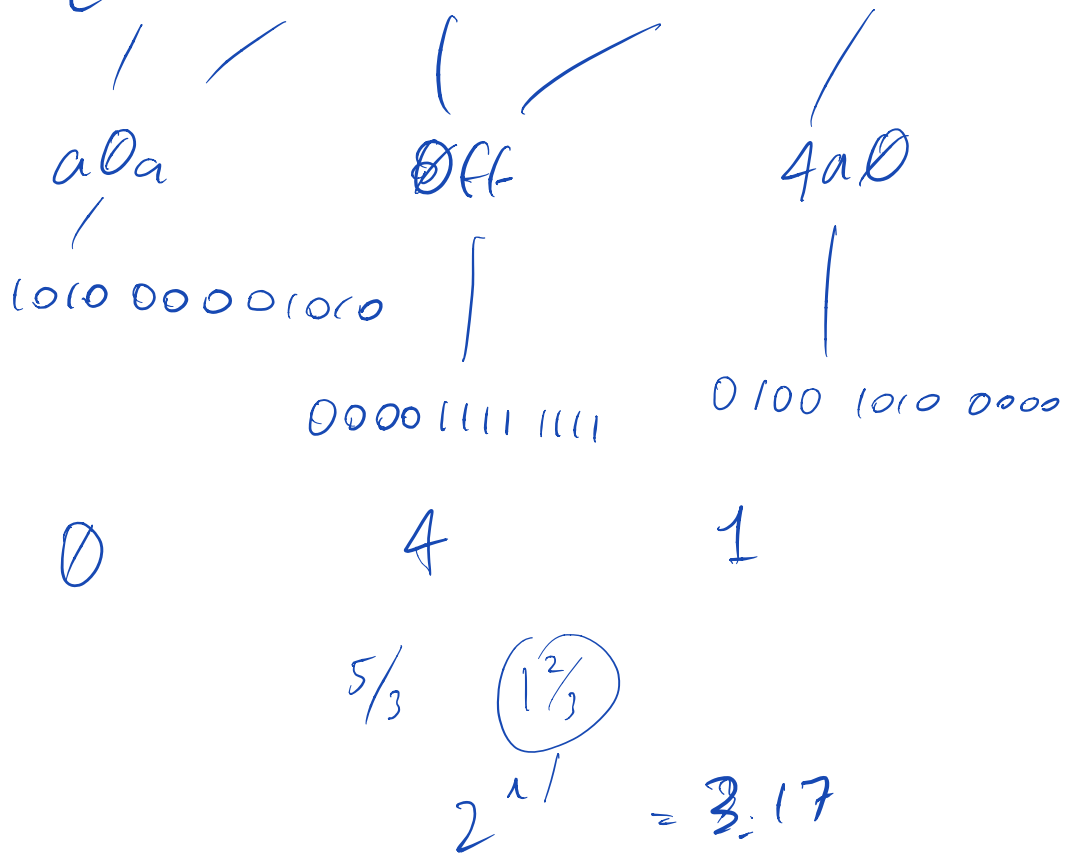
1

Click fraud!
"bot"

how many distinct IP addr clicked on link?

HyperLogLog

["el", "el", "hombre", "hombre", "tiempo"]



1 ∅

H T

a HHTTHTH

b THTHHTT

HHHHHH HHHHH

HLL:

one-pass

~~words~~

```
for w in words:
  h = hash(w)
  for chunk in h:
      c = countLZ(chunk)
      max[chunk] = max(max[chunk], c)
for c in chunk
    s += max[c]
return s / #chunks
```

4 3 1 2 0

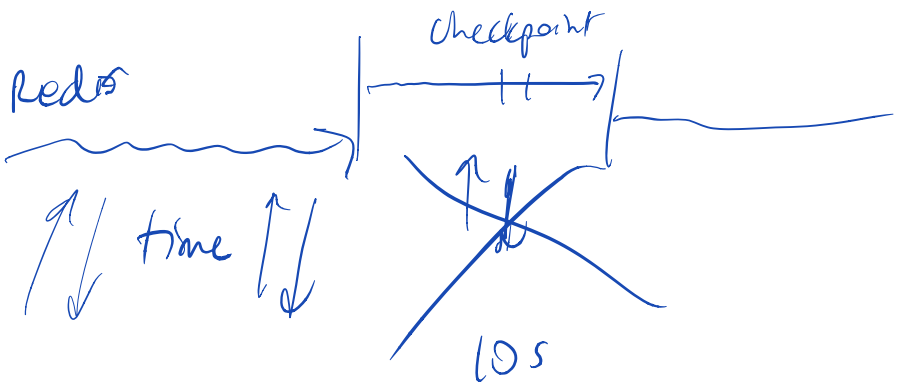
avg

Redis in-memory durability?

C : 1 ... 10

losing ⇒ 10 los writes

Checkpointing



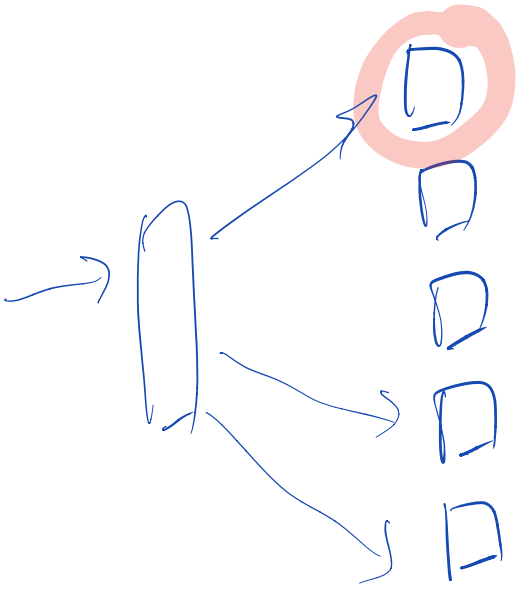
10m 10s

600s

590s / 600s = 98.3

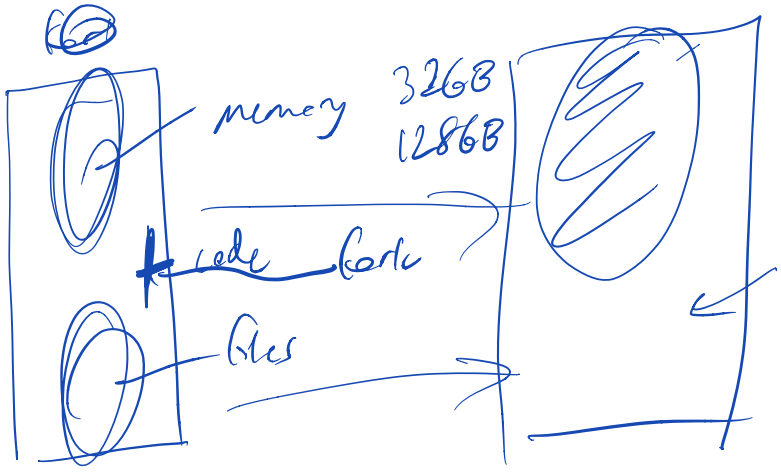
99.9

2.76 hours

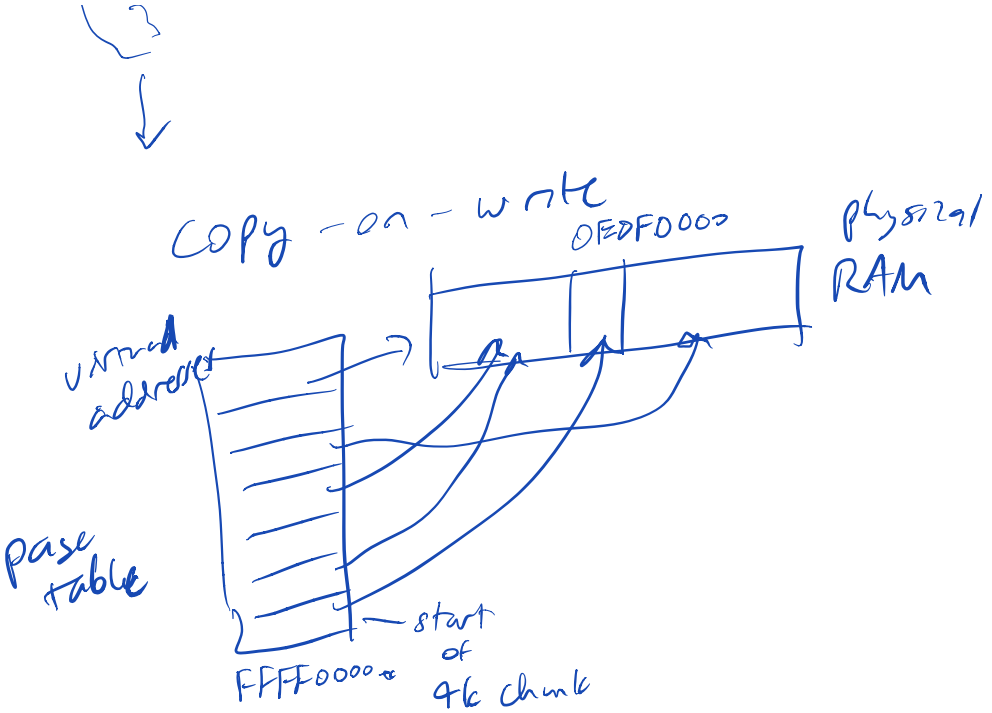


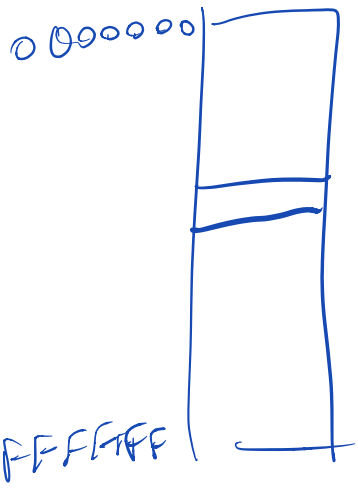
fork()

pthread_create(..., f)




```
if (fork()) {
  // parent
} // "done" child — openDoors();
  exit();
```





mem[0] ... mem[____]

mem[123] = 4

x = mem[1497...]

User 1:

mem[123] += 1

User 2

while true:

mem[123] = 0

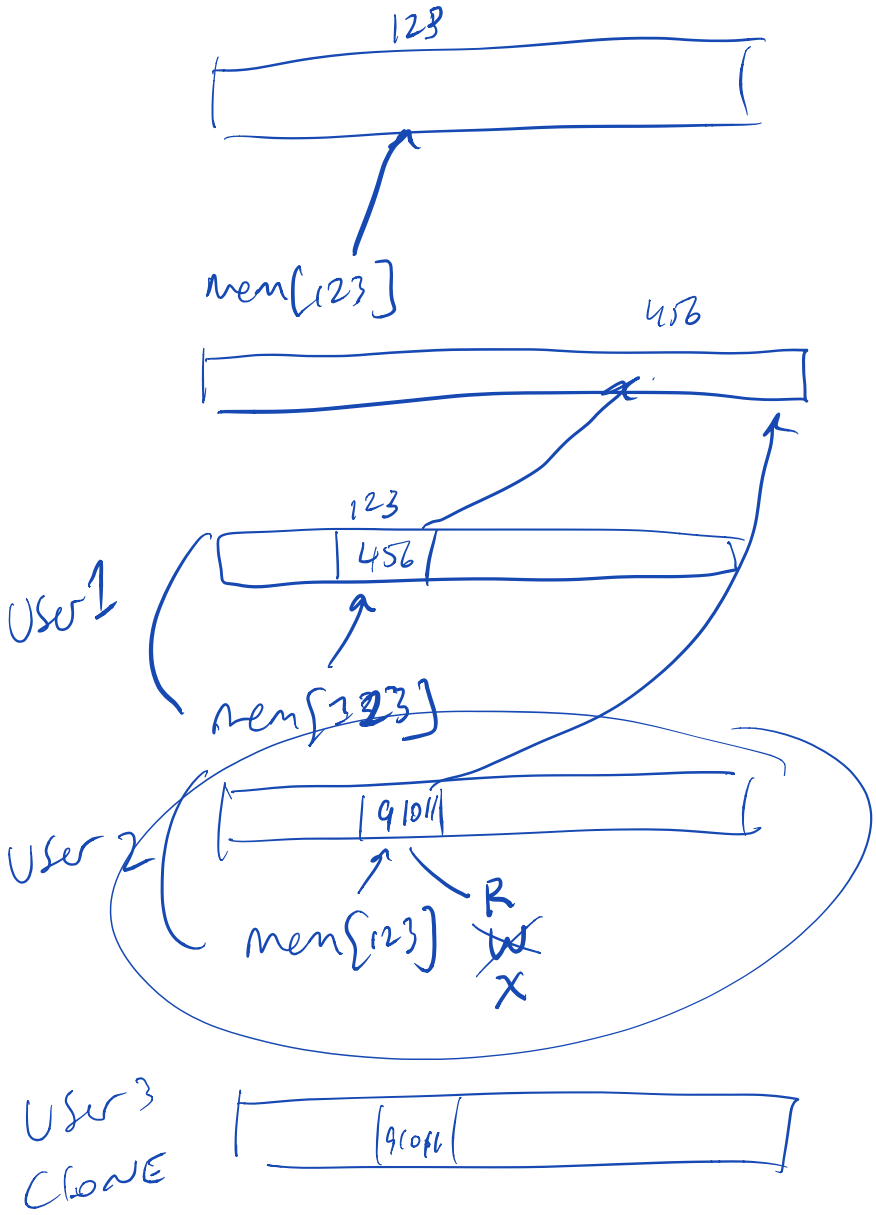
physical
memory

NO ISOLATION OF PROCESSES

mem[123]
+= 1

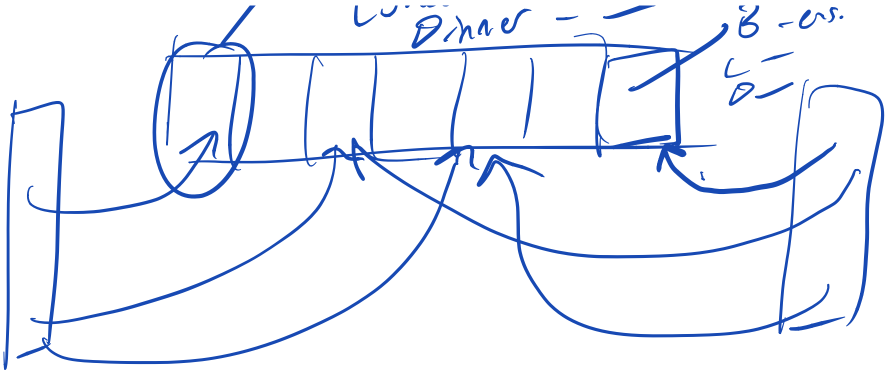
Peter Denning

"virtual memory"



Breakfast — ensaimada

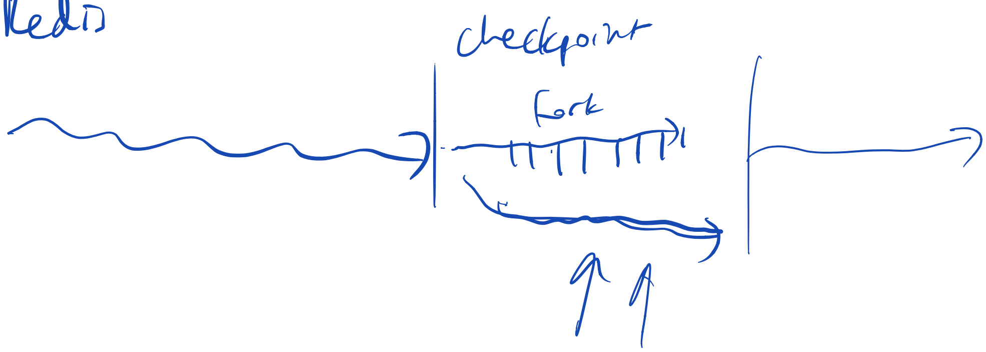
lunch —



lazy copy

"copy-on-write"

Redis



MongoDB "document store"

JSON files

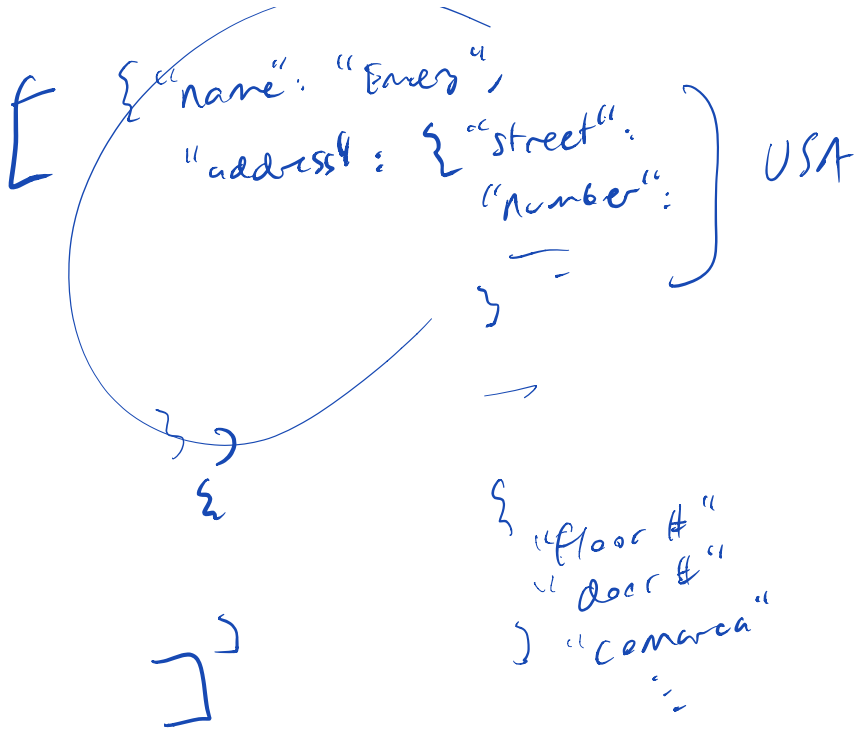
JavaScript Object Notation

{ fieldname$_1$: value$_1$

⋮

}

{ "name": "Emez",
"address": "MA",
"eyecolor": "brown"
}



ID	name	addrID

addrID	street	co...

DBMS - nesting ⇒ DB wins

homogeneity — everything is the same

- CREATE DB (ID, type), ( ), ...

⇒ SCHEMA

"name"

"age" "OLD"

DBMS: Java

X=12
y = "banana"
print X + 1

MongoDB: PyZon
JavaScript

SEGUE

R:

[1,2] + [0,0] = [1,2]

[1,2] + [10,100] = [11,102]

[1,2,3] + [10,100] = [11,102,]

"BSON"

Mongo DB

collections, databases

c.insert("JSON")
"{ name"

."

c.find('{ name: "Emily" }')

"_id"

indexing