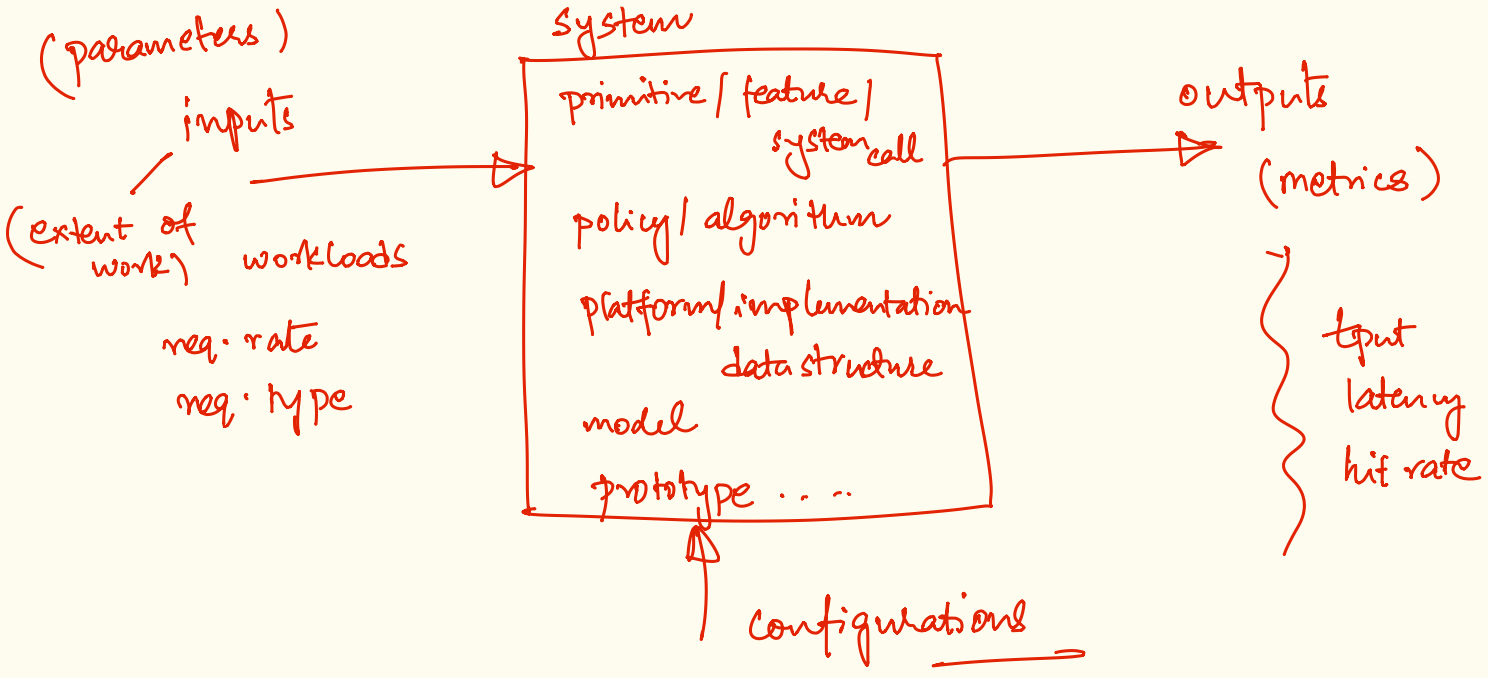
CS 695 the experiments story

motivation. formulation. design. build. deploy. measure. infer. report.
observe reason repeat

empirical action



meta-point / (guiding principles for an experiment)

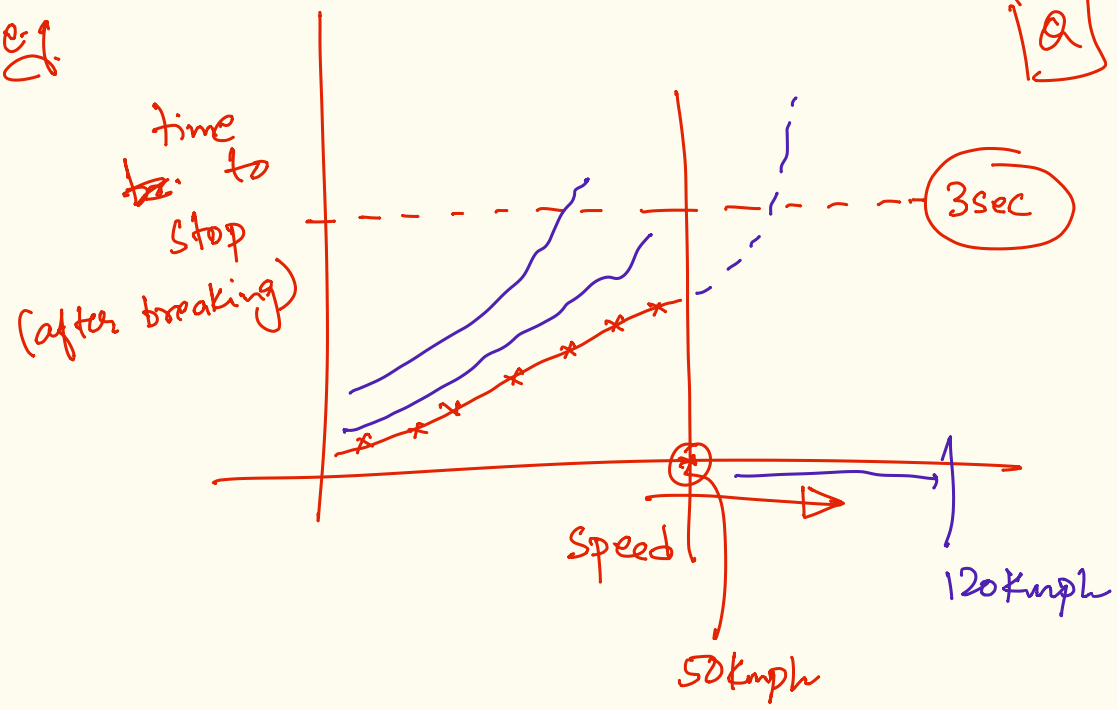
Q ~ question ~ no question, no experiment! why is an experiment needed?

A ~ accuracy ~ infer correct findings

R ~ repeatability ~ can redo experiments for validations & comparison.

C ~ completeness ~ coverage of the parameter space

e.g.



Q does a vehicle stop in 3 seconds after breaking?

---

① types of experiments / questions?

- establish motivation
- hypothesis testing / correctness
- performance - cost tradeoff study
- sensitivity analysis / performance characterization
- comparison
- best case - worst case scenarios.

---

② setup ~ hardware, software, workloads } soln. setup, config & inputs

~ system: "realistic" / ~~realistic~~ representative

setup

~ workloads ~ how to generate work for the system?

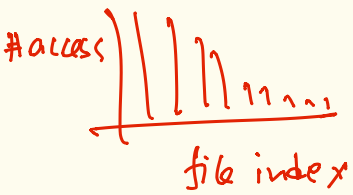
(i) synthetic benchmark

- { custom program to generate work based on ~~imp~~ configs. } probability distribution of the system.

- e.g: read : write ratio

object size distribution

- iozone, dbench, rubis, fio, ...



(ii) trace driven logging (record) & replay

(iii) real workloads on production systems.

---

(2) Execution of experiments

(i) modality — how will the system be implemented?

- analytical model
- simulation
- emulation
- prototype
- production

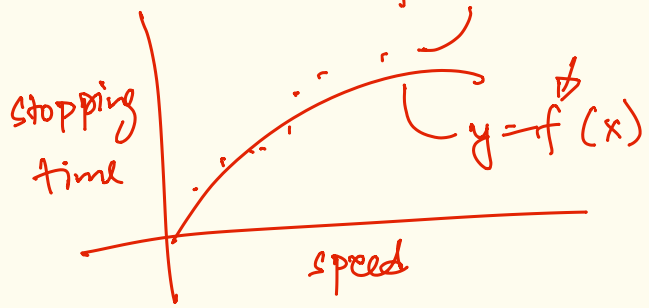
~ analytical model

- mathematical representation of the main components of the soln.

- eg: queuing model

$y = f(x)$

↑ f^n encodes behavior of system



(ii) simulation ~ all s/w soln.

~ objects / interactions are not "real" components.

eg: cache simulator

- no objects created/stored for caching.
- only manipulate "state" of cache (size, object list)
- no malloc / write to disk etc.
- process / procedure is true eg: eviction policy

~ do not account for real interactions

- impln. ovhds
- ~ non-deterministic parallelism

pros:

- quick / main-compute analysis
- ~ system is complicated / large

basic loop of a simulator

~ priority queue ~ prioritized by the time an event occurs

```
while (1) {
    e = getNextEvent (Q);
    processEvent (e, Q);
}

addEvent (e, Q) {
    t = timeEvent (e);
    addQueue (e, t, Q);
}
```

priority queue

on cache miss

- read object from disk
- simulate disk access latency

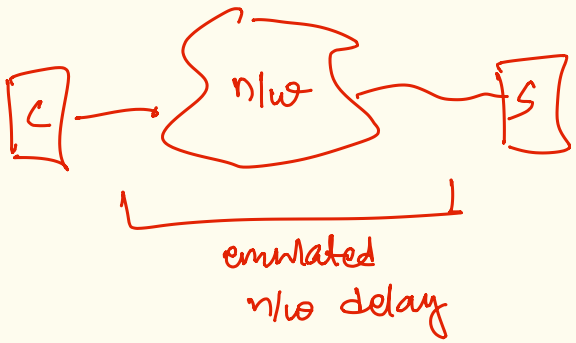
~ e: disk read completion

t: time from disk

from a model $y = f(x)$

(ii) emulation

~ emulating behaviour of some components of a system.



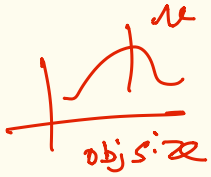
(iii) prototype

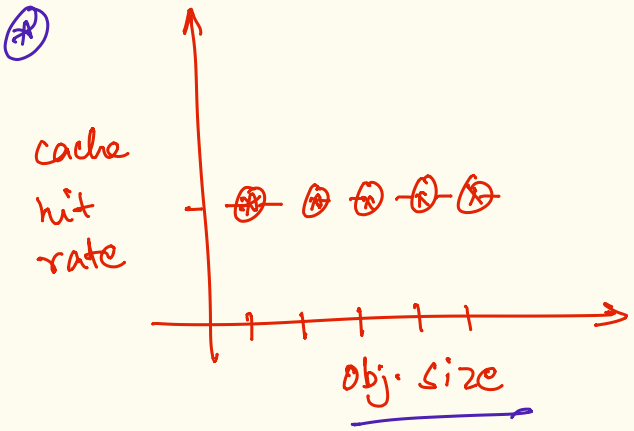
(iv) production

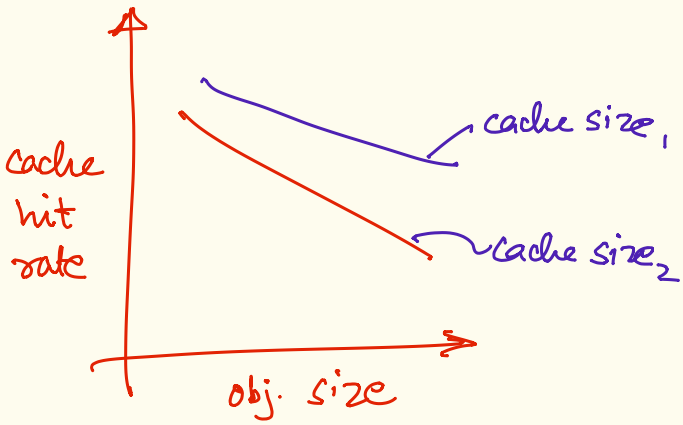
(b) process of experimentation

independent (parameters) varied vs dependent (metrics) measured

~ across experiments only (single) the parameter under study should be changed.







~ cache size was also changed per exp^r.

~ object size

cache size is constant

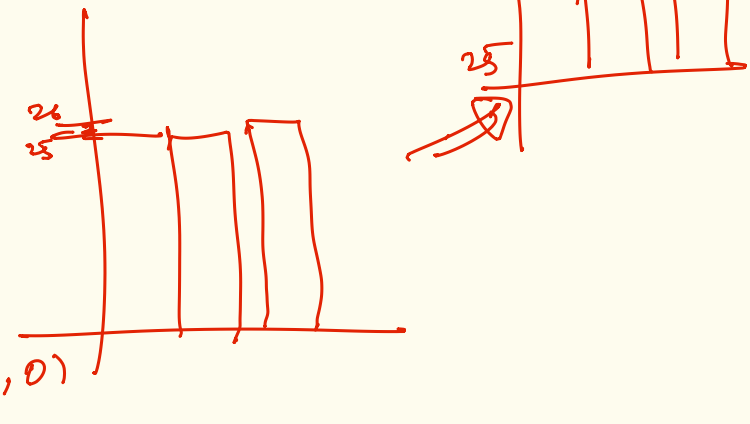
(*) workload pattern/sequence has to be identical / expectation.

(*) instrumentation overheads.

~ does measurement effect the system behaviour

(4) Observations & inferences via representations

- tables
- graphs



revisit QARC & repeat.