

Load Balancing

(1) Traditional techniques : RANDOM
(Static) ROUND-ROBIN
FIFO

(2) Size-based approaches

a) - High variation of task size (Pareto Distribution

where α is the task variation

$0 < \alpha < 2$

eg

OS if α near 2 $\alpha \geq 1.5$
The variation is not

high.

if $0.8 < \alpha \leq 1.1$ very high

REALISTIC CONDITIONS FOR variation of task size

eg. http traffic

OF TRAFFIC eg. http, ftp

if $\alpha < 0.8$ extreme

variation

NOT REALLY

of task size

REALISTIC

2 b) task size known / un-known

b-1.)

size known by the dispatcher

e.g

SITK-V: computes

the "best" cutoff, say

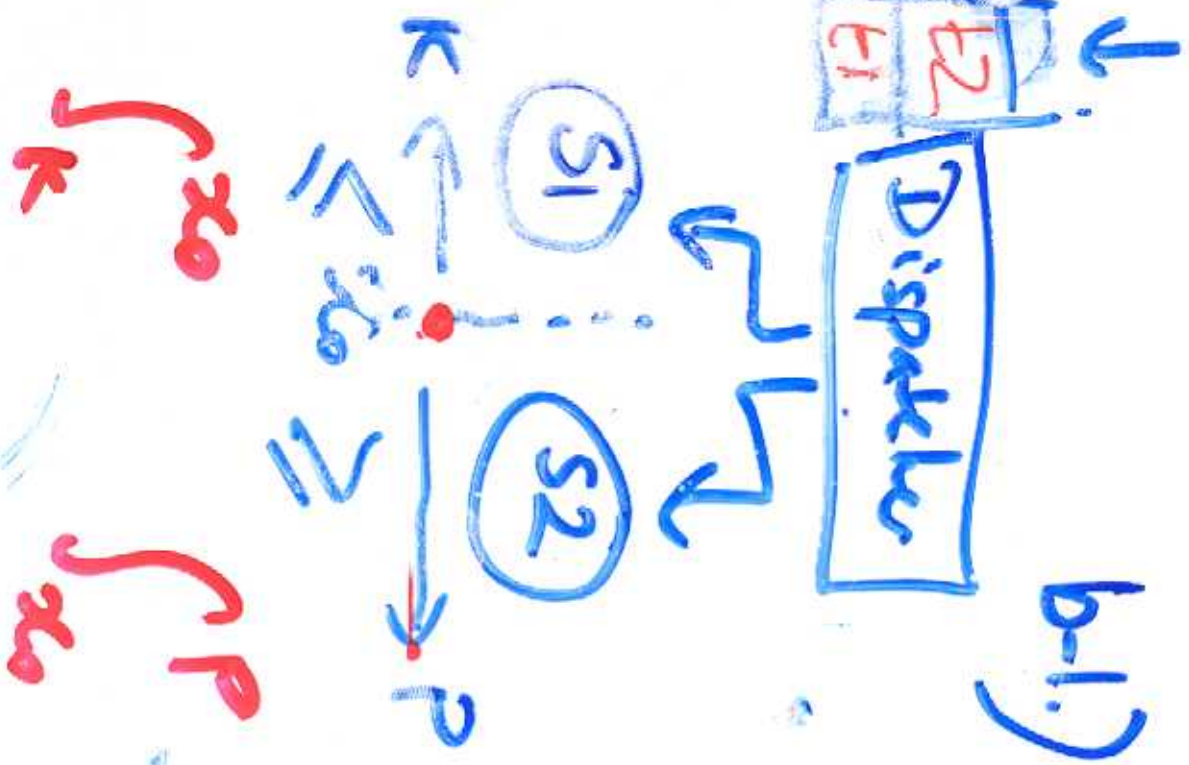
x_0 , such that

$t_1 \leq x_0 \rightarrow$ Dispatcher

sends ~~t_1~~ t_1

to S_1

(3)-7



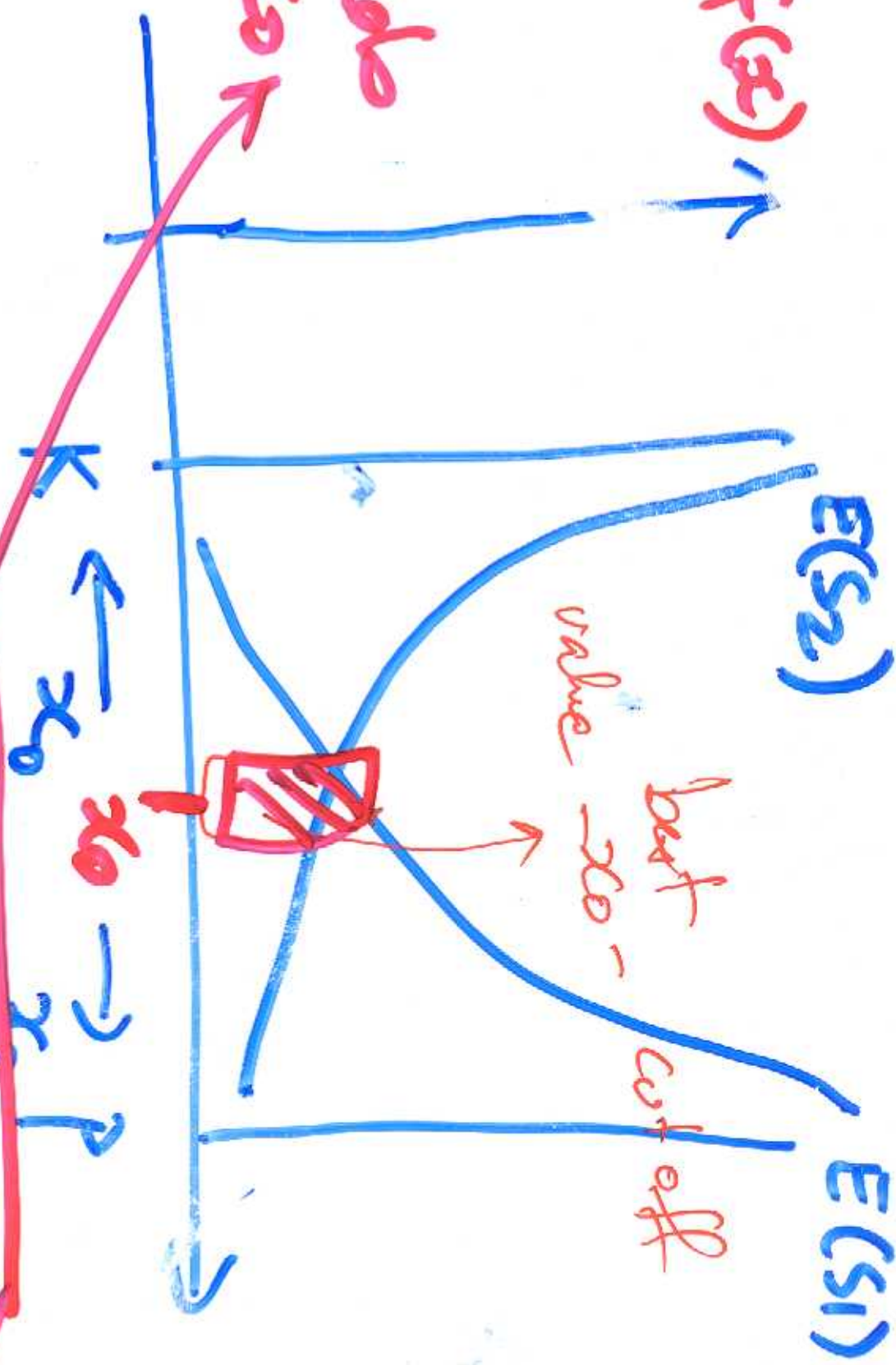
• $t_1 > t_0 \rightarrow$ Dispatches
will send it to

S2

The computation of x_0
is done in such a way
you have the same
slow done in S1 and S2

$\int_{x_0}^{p_0} x f(x) dx$

depends on x_0



$$E(S_2) = \sum_{j \in S_2} T_j$$

Waiting Time (T_j)

Size (T_j)

⑤ - 2

Dispatchable

SITA-E: similar to

SITA-V



COMPUTE THE "BEST" VALUES

FOR THE CUT-OFF

x_1, x_2, \dots, x_{n-1}

E: task

Size

$0 < t \leq x_1 \rightarrow S_1$

$x_1 < t \leq x_2 \rightarrow S_2$

Q2

$$x_2 \leq x_3 \rightarrow S_3$$

"BEST" \equiv MEANS THAT

THEY HAVE THE
SAME SLOWDOWN

$$E(S_1) \sim E(S_2) = E(S_3) = \dots = E(S_n)$$

one dependent on x_1

dependent on x_2

⑦-⑦

b)

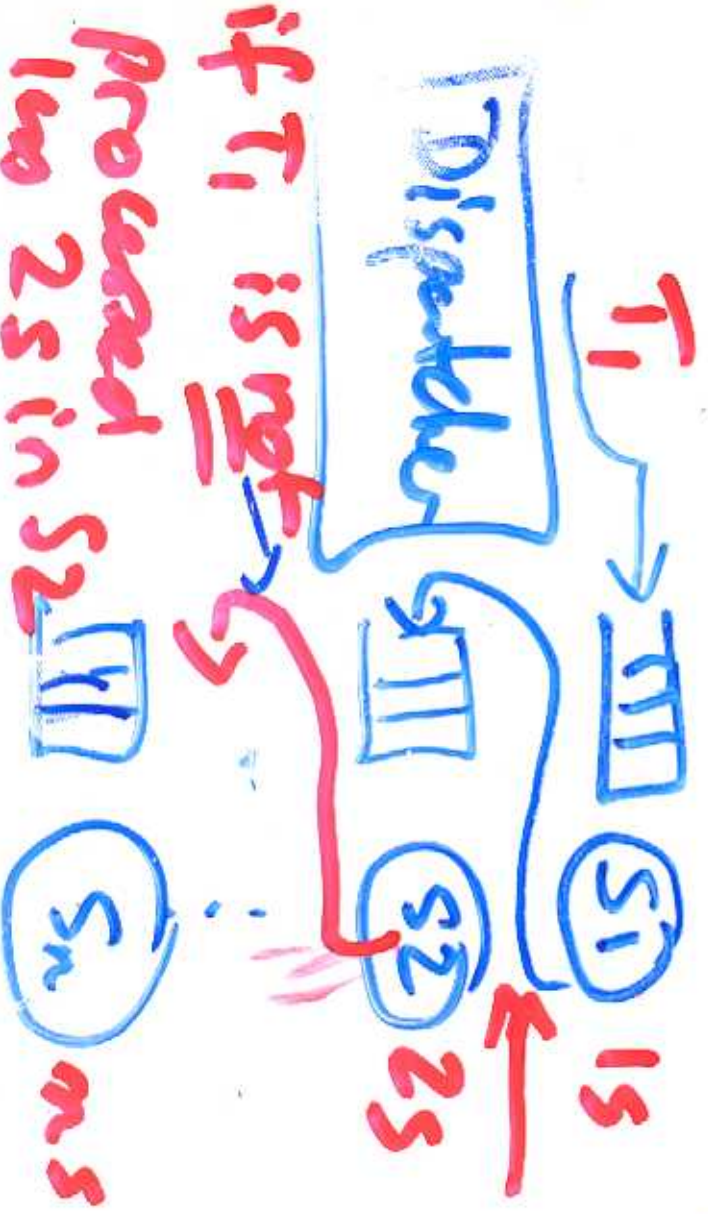
b-1) size is known

SIT_{A-V}, SIT_{A-E}

b-2) size is un-known

TACS (Task Assignment)

by Guessing Size)



if T_1 is not processed in S2 then

↳ forwards T_1 to S2

8-2

TACS is

good - not need to know
the task size.

x_1 • tasks are "grouped"

← based on their size

$$0 < t < 1s \rightarrow s_1$$

$$1 \leq t < 2s \rightarrow s_2$$

$$\vdots$$

$$(n-1)s < t < ns \rightarrow s_n$$

↳ Similar to Sigma-E 9-7

but

- SITA-E @ know

TASK

size)

(NOT

know TASK

size)

07

THESE

limitations TASKS

PRIORITY ← Handoffs (too many
QUEUE
+ ASSIGNED
TASKS AT
ALL SERVERS ←

re-start of tasks
at different servers)

~~both~~ bottleneck
at "top" servers (e.g. S1,

S2, ...)

↳ All requests go

through S1, S2,

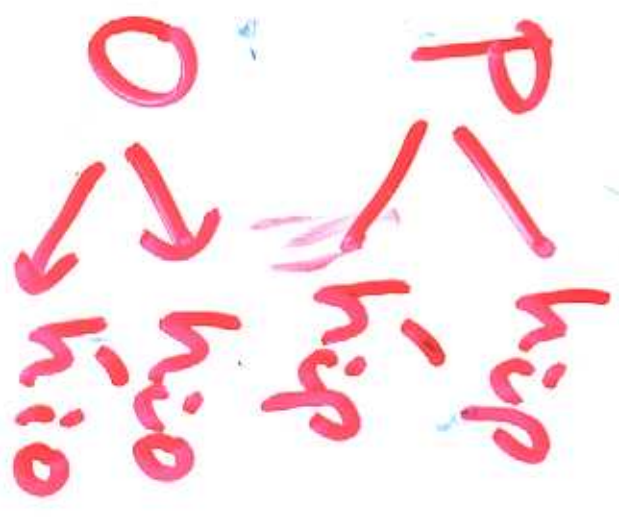
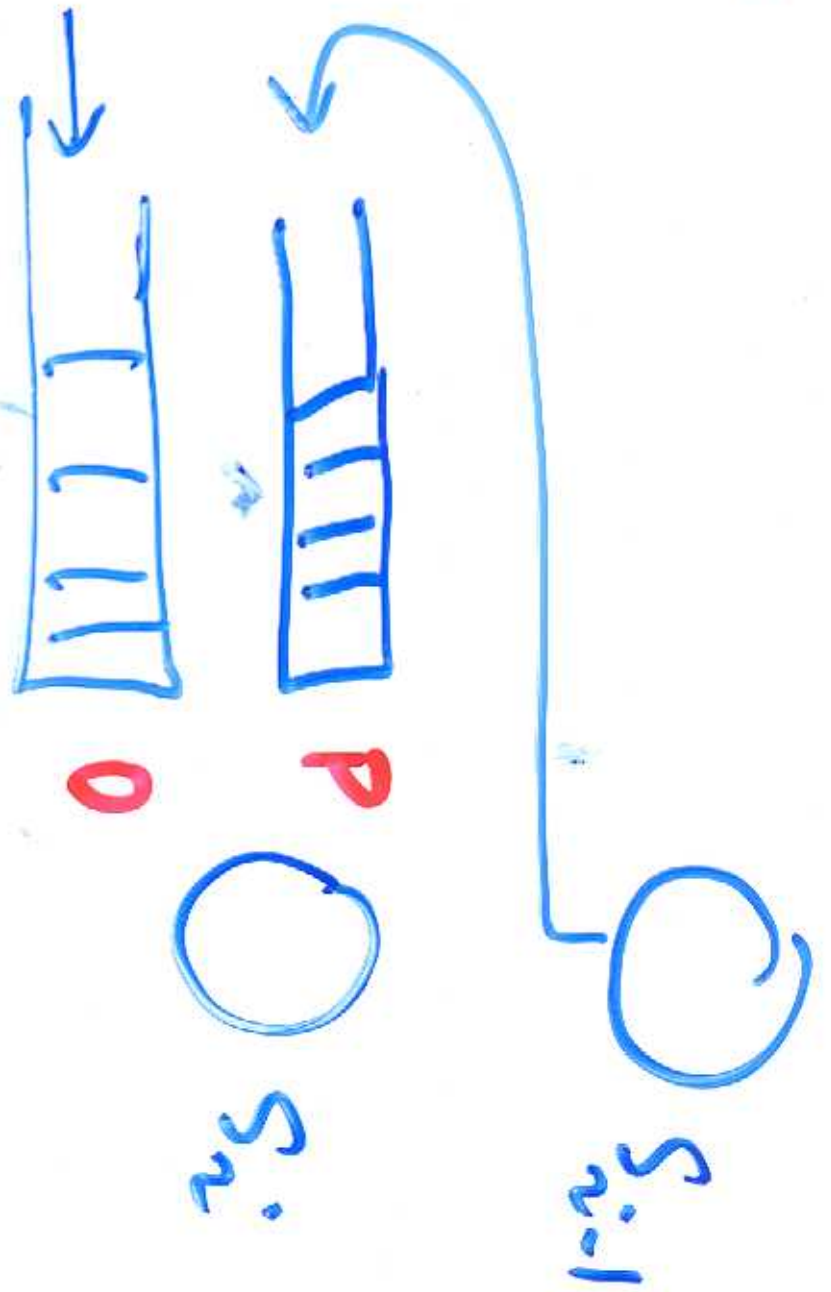
PRIORITY \leftarrow \leftarrow
*
ORDINARY
QUEUES

High variation in
task size in
the queues of
intermediate servers

TAQR IMPROVES TACS

TAESR

Priority Queue
 ordinary queue
 Dispatcher



$$P_i = P(X \leq s_i) = \int_0^{s_i} f(x) dx$$

↳ densite
of Pareto
Distrib.

$h_{i0} = q_i$ ~~_____~~

h_{i1}



heap = # of tasks that visit
priority queue of S2

↳ tasks have been
restarted a S1

$$h_{2P} = q_1 (1 - P_A)$$

heap

S

S2 (ordinary queue of S2)

S1 → S2 → S3

↑
ordinary priority

(S) →

$$h_{30} = q_2(1-p_2) + q_1(1-p_1)(1-p_2)$$

⑩-7

$$h_{i,p} = q_{i-1} (1 - p_{i-1}) + q_{i-2} (1 - p_{i-2}) (1 - p_{i-1}) + q_{i-3} (1 - p_{i-3}) (1 - p_{i-2}) (1 - p_{i-1}) + \dots + q_{i-1} (1 - p_{i-1})$$

tasks of ordinary \rightarrow q_{i-1}

$i-3$ $i-2$ $i-1$

$(1 - p_{i-2}) (1 - p_{i-1})$ $(1 - p_{i-1})$

• TASKS → TAESSR

↑ included priority

Queue @ server

• LLF

(Least Load
First)

→

FFF

(Least Flow time
First)

assigning to the

~~task priority~~

fill server

18.7

LF_{FF}

→

LF_{FF}-Priority

↑

compute priorities
for tasks

19-7