State-dependent and Energy-aware Control of Server Farm

Esa Hyytiä, Rhonda Righter and Samuli Aalto

Aalto University, Finland
UC Berkeley, USA

First European Conference on Queueing Theory
ECQT 2014

Aalto University
School of Electrical Engineering

Outline

1. Model
   - Queueing system with job dispatching
   - Running costs and setup delay
2. Optimal static operation
   - Numerical example
3. Dynamic operation
   - Value functions for M/G/1
   - Dynamic dispatching and switching off policies
4. Conclusions

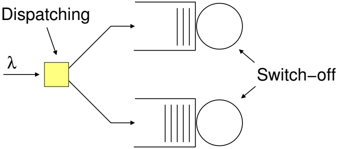
Aalto University
School of Electrical Engineering

I. Model

Aalto University
School of Electrical Engineering

Model

System model:



- n identical FCFS parallel servers
- Jobs dispatched upon arrival
- Running costs at rate e (energy)
- Idle servers can be switched off
- Setup delay of s when switched on
- Objective:

$$\boxed{\min \quad \mathrm{E}[N] + e \cdot \mathrm{E}[A]} \qquad \text{or} \qquad \boxed{\min \quad r_T + r_R}$$

where

- $\mathrm{E}[N]$ is the mean number in the system ($\mathrm{E}[N] = \lambda\,\mathrm{E}[T]$)
- $\mathrm{E}[A]$ is the mean number of running servers

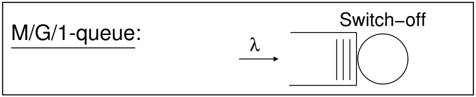
Dispatching and Switch-off decisions!

Aalto University
School of Electrical Engineering

II. Static operation

Aalto University
School of Electrical Engineering

Single M/G/1



- **Static switch-off policy**:
  1. `NeverOff`: keep the server always ON
  2. `InstantOff`: switch off immediately when idle

- **Mean running cost**:

$$r_R = \begin{cases} \dfrac{\lambda(\mathrm{E}[X] + s)}{1 + \lambda s} e, & \text{if } \texttt{InstantOff} \\ e, & \text{if } \texttt{NeverOff} \end{cases}$$

- **Mean delay cost**:

$$r_T = \begin{cases} \dfrac{\lambda^2\,\mathrm{E}[X^2]}{2(1-\rho)} + \dfrac{\lambda s(2+\lambda s)}{2(1+\lambda s)} + \lambda\,\mathrm{E}[X], & \text{if } \texttt{InstantOff} \\ \dfrac{\lambda^2\,\mathrm{E}[X^2]}{2(1-\rho)} + \lambda\,\mathrm{E}[X], & \text{if } \texttt{NeverOff} \end{cases}$$

Aalto University
School of Electrical Engineering

The total cost rate under **InstantOff**

$$r_{\mathrm{IO}} = \overbrace{\frac{\lambda^2\,\mathrm{E}[X^2]}{2(1-\rho)} + \frac{\lambda s(2+\lambda s)}{2(1+\lambda s)} + \lambda\,\mathrm{E}[X]}^{\text{Sojourn time}} + \overbrace{\frac{\lambda(\mathrm{E}[X]+s)}{1+\lambda s}e}^{\text{Running cost}}$$

and under **NeverOff**,

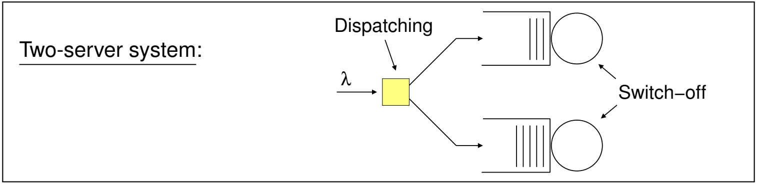
$$r_{\mathrm{NO}} = \frac{\lambda^2\,\mathrm{E}[X^2]}{2(1-\rho)} + \lambda\,\mathrm{E}[X] + e$$

Studying $r_{\mathrm{IO}} - r_{\mathrm{NO}}$ $\Rightarrow$ **InstantOff** better if $\boxed{e > \frac{\lambda s(2+\lambda s)}{2(1-\rho)}}$

Note: $\boxed{\text{Threshold depends only on } \lambda\,\mathrm{E}[X] \text{ and } \lambda s}$

Aalto University
School of Electrical Engineering

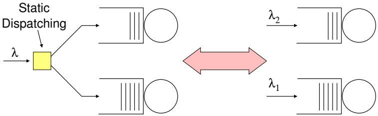
Static dispatching: Decomposition



- Static job dispatching
  - *Independent of the queue states*
  - E.g., random split (RND) and SITA[1]

- Decomposition:

  Static dispatching:
  $\Rightarrow$ *n* independent M/G/1 queues

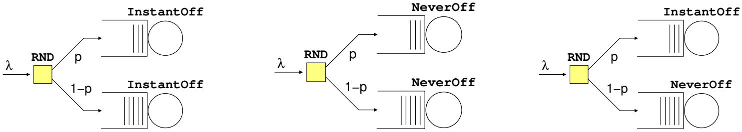


- Mean results available for M/G/1
  - The total cost rate can be computed

[1] Size-Interval-Task-Assignment: *"short to queue 1, other to queue 2.*

Example

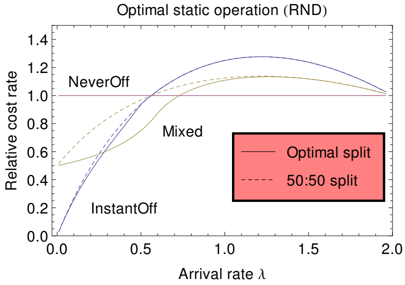
- Two identical servers:
  - Setup time $s = 2$
  - Running cost rate $e = 1$
- Service times $X \sim \mathrm{Exp}(1)$
- Poisson arrival process with rate λ
- RND dispatching (Bernoulli split) w.p. p
- Switch-off policies: **`NeverOff`** and **`InstantOff`**



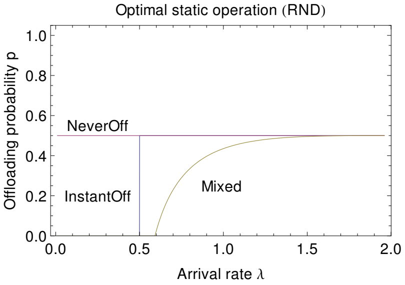
(1) **InstantOff** (2) **NeverOff** (3) **Mixed**

Aalto University
School of Electrical Engineering

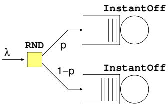
Results



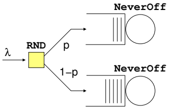
(a) Relative performance



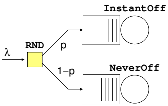
(b) Optimal split



(1) **InstantOff**



(2) **NeverOff**



(3) **Mixed**

Results

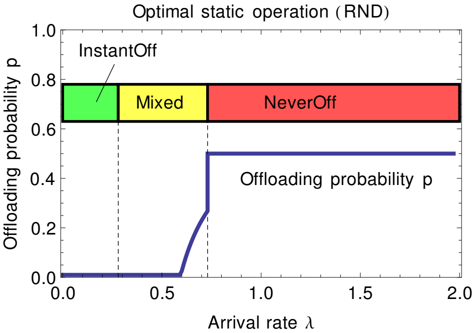


Figure: Optimal operation with RND.

Static dispatching

Observations

- Optimal switch-off policy changes as the load increases

  `InstantOff` $\rightarrow$ **`Mixed`** $\rightarrow$ **`NeverOff`**

- **`NeverOff`** always splits the jobs uniformly
  - Running costs are fixed, $2 \times e$
  - Uniform split minimizes the mean sojourn time

- **`InstantOff`** and **`Mixed`** use
  - Only one server under a very low load
  - Uniform split under a very high load

Aalto University
School of Electrical Engineering

III. Dynamic operation

Aalto University
School of Electrical Engineering

Dynamic operation

- Dynamic dispatching & switch-off decisions
  - Require state information
  - Can improve the performance
- cf. JSQ vs. RND
- Option to switch-off makes the situation more complicated
- We consider **size- and state-aware** setting

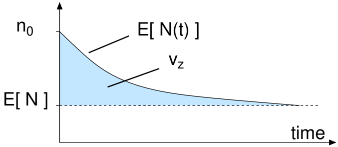
How to capitalize the state information?

Aalto University
School of Electrical Engineering

Preliminaries

- Consider an arbitrary (stable queueing) system
  - $C_z(t)$ = incurred costs in $(0, t)$ when initially in state z
  - r = mean cost rate
- **Value function** characterizes the expected long-term deviation from the mean cost rate r

$$\boxed{v_z \triangleq \lim_{t\to\infty} \mathrm{E}[C_z(t) - rt]}$$



- For two initial states z_1 and z_2,

$$v_{z_2} - v_{z_1}$$

gives the expected difference in the cumulative costs.

Enables the comparison of initial states!

Size-aware value functions for M/G/1

- Virtual backlog u includes the remaining setup time δ,

$$u = \delta + x_1 + \ldots, x_n.$$

- Value function w.r.t. running costs is[2]

$$v_R(u) - v_R(0) = \begin{cases} \frac{u}{1+\lambda s} e, & \text{if } \mathbf{\texttt{InstantOff}} \\ 0, & \text{if } \mathbf{\texttt{NeverOff}} \end{cases}$$

- Value function w.r.t. sojourn time is[2]

$$v_S(u) - v_S(0) = \begin{cases} \frac{\lambda}{2(1-\rho)} \left(u^2 - \frac{s(2+\lambda s)u}{1+\lambda s} \right) & \text{if } \mathbf{\texttt{InstantOff}} \\ \frac{\lambda\, u^2}{2(1-\rho)} & \text{if } \mathbf{\texttt{NeverOff}} \end{cases}$$

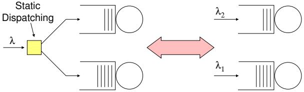
The immediate cost is equal to the resulting backlog u.

[2] Hyytiä, Righter, Aalto: Performance Evaluation (2014).

Aalto University
School of Electrical Engineering

First policy iteration

- Consider a dispatching system with a static policy α_0
- System decomposes to n parallel queues



- Value function is the sum of M/G/1 value functions

$$v_z = \sum_{i=1}^{n} v^{(i)}(z_i).$$

- Policy iteration step gives a new **dynamic policy**,

$$\boxed{\alpha(z,x) = \underset{i}{\operatorname{argmin}} \quad \overbrace{c(z_i,x) + v^{(i)}(z_i \oplus x) - v^{(i)}(z_i)}^{\text{Admission cost}},}$$

where $c(z_i, x)$ is the immediate cost of server i

Aalto University
School of Electrical Engineering

1. **First policy iteration**

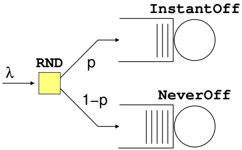
   (static policy) + (value function) $\overset{FPI}{\Rightarrow}$ new policy

   - Queues are evaluated assuming future jobs according to α_0

2. **Lookahead**
   - Evaluate decisions such as
     - This job to server *i*
     - Next job to server *j* (tentatively)
     - Later arriving jobs according to a static α_0
   - More accurate evaluation of each possible action
   - Yields typically a better policy than FPI

Aalto University
School of Electrical Engineering

Numerical example:



- Two servers
  - Server 1: **`NeverOff`**
    Server 2: **`InstantOff`**
  - Setup delay: $s = 2$
  - Running cost: $e = 1$
- Objective: Minimize $r_W + r_R$ (waiting time + running costs)
- Reference dispatching policies
  - RND: random 50:50 split
  - SITA-E: short jobs to server 1, long to server 2
  - Myopic: socially optimal if no later arrivals
  - Greedy: individually optimal choice (only delay)
- Value function based policies
  - FPI: policy iteration based on SITA-E
  - Lookahead: "advanced FPI", considers also the next job (tentatively)

Numerical example: $X \sim \mathrm{Exp}(1)$

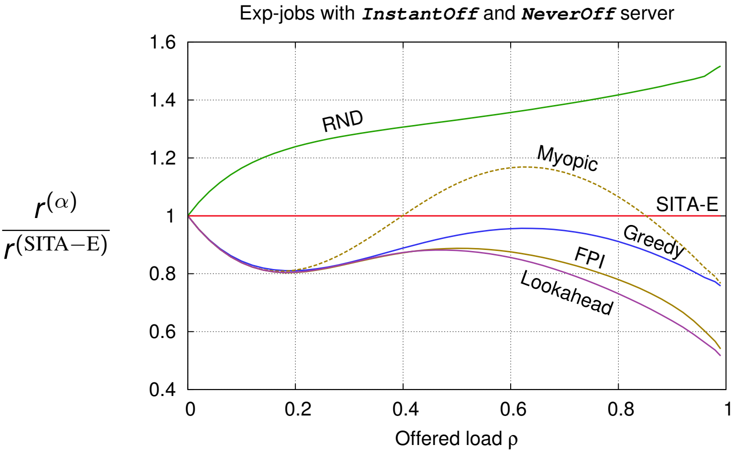


Figure: Relative mean cost rate with the objective of $r_W + r_R$.

Aalto University
School of Electrical Engineering

Numerical example: $X \sim \mathrm{Pareto}(1)$ (truncated)

Figure: Relative mean cost rate with the objective of $r_W + r_R$.

Aalto University
School of Electrical Engineering

App #2: Improved Switching-off policies

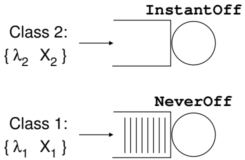


Figure: System according to the static basic policy α_0

Scenario

- Server 1 is `NeverOff` and server 2 `InstantOff`
- Dispatching according to a static α_0
- Server 1 is busy ($u_1 \gg 0$) when server 2 becomes empty

"Should we keep server 2 still running?"

Aalto University
School of Electrical Engineering

1 Change of roles (renaming)

Idea: swap the roles of the servers?

- Server 1 will receive class 2 jobs, and vice versa
- Server 1 becomes `InstantOff`, and server 2 `NeverOff` (and is thus kept running)

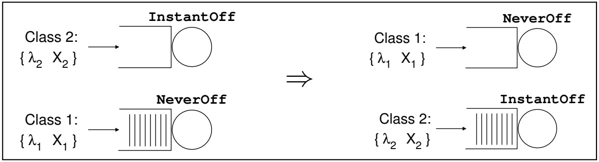


Figure: Swap the roles?

Aalto University
School of Electrical Engineering

Expected gain can be evaluated with the value functions!

$$\Delta = \frac{u}{2}\left(\left[\frac{\lambda_1}{1-\rho_1} - \frac{\lambda_2}{1-\rho_2}\right] u + \frac{\lambda_2 s(2+\lambda_2 s)}{(1-\rho_2)(1+\lambda_2 s)} - \frac{2e}{1+\lambda_2 s}\right)$$

If $\Delta > 0$, then keep server 2 running

Equivalently,

$$e < \frac{1+\lambda_2 s}{2}\left(\frac{\lambda_1}{1-\rho_1} - \frac{\lambda_2}{1-\rho_2}\right) u + \frac{\lambda_2 s(2+\lambda_2 s)}{2(1-\rho_2)}$$

- Uniform RND as α_0 gives

$$e < \frac{\lambda_2 s(2+\lambda_2 s)}{2(1-\rho_2)}$$

(same threshold as with a single M/G/1...)

- SITA-E as α_0 gives

$$e < \frac{1}{2(1-\rho_i)}\left((1+\lambda_2 s)(\lambda_1 - \lambda_2)u + \lambda_2 s(2+\lambda_2 s)\right)$$

Aalto University
School of Electrical Engineering

2 Lookahead approach

Similarly as with the dispatching we can ask

A: "Keep server 2 running and assign the next job there?"

For comparison:

B: "Switch off server 2 and assign the next job still there"
C: "Switch off server 2 and assign the next job to server 1"

For each action, we can compute

- The expected costs incurred until the next job arrives
- The expected future costs afterwards (w/ value functions)

While the expected gain is positive, keep server 2 running!

(details omitted)

Aalto University
School of Electrical Engineering

IV. Conclusions

Aalto University
School of Electrical Engineering

Conclusions

- Server farm modelled as a queueing system
  - Job dispatching decisions
  - Server switch-off decisions to save energy
  - Setup delay included
- Static control straightforward
  - Mean results available
- Dynamic control is harder
  - Value functions and FPI/Lookahead approaches
  - Can be applied to both dispatching and switching off
- Omitted:
  - Other cost functions (e.g., W^2, holding costs)
  - Other scheduling disciplines (e.g., LCFS and PS)

Aalto University
School of Electrical Engineering

Thanks!

1. Hyytiä, Righter and Aalto, *Task Assignment in a Heterogeneous Server Farm with Switching Delays and General Energy-Aware Cost Structure*, Performance Evaluation (2014).
2. Hyytiä, Righter and Aalto, *Energy-aware Job Assignment in Server Farms with Setup Delays under LCFS and PS*, ITC 2014.

Aalto University
School of Electrical Engineering